

# MASAKARI-The\_Holy\_Axe\_Textual\_Sidekick



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,001 ' Masakari.bas, written in QB64 by Kaze (sanmayce@sanmayce.com), 2022-Jan-26
00,002 ' Revised 2021-Jul-28, fixed double-clicks, fixes in combo handling; fixed report line length bug when (ExpandTabs was not dealing with the full size).
00,003 ' Many thanks go to Galleon, also thanks go to the www.qb64.org/forum members for sharing useful excerpts/etudes
00,004
00,005 ' For F10 is needed: Schizandrafield_Corpus_revision_D_(45-corpora_-unique-words).sorted.ind
00,006 ' For F11 is needed: Dictionary_Specification_Language_(ABEYY_Software_House)_Oxford_English_Dictionary_2nd_Edition_Version_4_(En-En)_ANSI.dsl
00,007 ' For F12 is needed: dict2txt_Online_Etymology_Dictionary1.1.txt
00,008
00,009 ' Note #1: For wrapping set the flag 'WrapFlag = 1' or 'WrapFlag = 0' for vanilla.
00,010 ' 'MASAKARI_General-Purpose_Grade_English_Wordlist.wrd' is the unigram i.e. wordlist to be used here.
00,011
00,012 Const RSHIFTkey& = 100303
00,013 Const LSHIFTkey& = 100304
00,014 Const RCTRLkey& = 100305
00,015 Const LCTRLkey& = 100306
00,016 Const RALTkey& = 100307
00,017 Const LALTkey& = 100308
00,018
00,019 Const BACKSPkey& = 8
00,020 Const TABkey& = 9
00,021 Const SPACEkey& = 32
00,022 Const ESCkey& = 27
00,023 Const ENTERkey& = 13
00,024
00,025 Const HOMEkey& = 18176
00,026 Const ENDkey& = 20224
00,027
00,028 Const INSkey& = 20992
00,029 Const DELkey& = 21248
00,030
00,031 Const PGUPkey& = 18688
00,032 Const PGDNkey& = 20736
00,033
00,034 Const LEFTkey& = 19200
00,035 Const RIGHTkey& = 19712
00,036 Const UPkey& = 18432
00,037 Const DOWNkey& = 20480
00,038
00,039 Dim Shared ToLoadOrNotFlag As Integer ' Global flag, if 1 then load the entire file - use it to speed up parsing. Set to 0 if you want to load bigger files, slow it is.
00,040 Dim Shared WrapFlag As Integer
00,041
00,042 Dim Shared CursorS As Integer
00,043 Dim Shared CursorE As Integer
00,044
00,045 Dim Shared Schizandra$(1 To 45)
00,046 Schizandra$(1) = "AHD, American_Heritage_Dictionary_4_(En-En)_WHOLEWORDS.dsl"
00,047 Schizandra$(2) = "BNC, Machine-Learning_British-National-Corpus_XML-edition.tar"
00,048 Schizandra$(3) = "BRE, Britannica_Encyclopedia_2010_1.563_miled_(En-En)_ANSI.dsl"
00,049 Schizandra$(4) = "CAL, Cambridge_Advanced_Learner's_Dictionary_4th_Ed_(En-En).dsl"

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,050 Schizandra$(5) = "CCA, Collins_COBUILD_Advanced_Learner's_English_Dictionary_(5th_ed)_(En-En)_WHOLEWORDS.dsl"
00,051 Schizandra$(6) = "DMC, DeepMind_Q_and_A_Dataset_cnn_downloads_(92579_files).tar"
00,052 Schizandra$(7) = "DMD, DeepMind_Q_and_A_Dataset_dailymail_downloads_(219506_files).tar"
00,053 Schizandra$(8) = "EDR, encyclopediadramaticase-20150628-current.tar"
00,054 Schizandra$(9) = "EJD, Encyclopaedia_Judaica_(in_22_volumes)_TXT.tar"
00,055 Schizandra$(10) = "EJN, ENAMDICT_Japanese_names"
00,056 Schizandra$(11) = "FDU, For_Dummies_978-ebooks_Collection.tar"
00,057 Schizandra$(12) = "GGB, Google_Books_corpus_version_20130501_English_All_Nodes.txt"
00,058 Schizandra$(13) = "GNS, Genesis_Library_English / Genesis_Library_English_fiction (2,290,923 .TXT files)"
00,059 Schizandra$(14) = "HCN, Hacker_News_2006_to_2017-jul.json"
00,060 Schizandra$(15) = "IST, INTERNET_SACRED_TEXT_ARCHIVE_DVD-ROM_9_(English_140479_htm_files).tar"
00,061 Schizandra$(16) = "LDC, Longman_Dictionary_of_Contemporary_English_5th_Ed_(En-En)_WHOLEWORDS.dsl"
00,062 Schizandra$(17) = "MCD, Macmillan_English_Dictionary_(En-En).dsl"
00,063 Schizandra$(18) = "MCT, Macmillan_English_Thesaurus_(En-En).dsl"
00,064 Schizandra$(19) = "NSO, New_Shorter_Oxford_English_Dictionary_fifth_edition.tar"
00,065 Schizandra$(20) = "OED, Oxford_English_Dictionary_2nd_Edition_Version_4_(En-En)_WHOLEWORDS.dsl.txt"
00,066 Schizandra$(21) = "OSH, OSHO.TXT"
00,067 Schizandra$(22) = "PGT, Project_Gutenberg_DVD-2010_(29180_files).tar"
00,068 Schizandra$(23) = "RDD, Reddit_Comments_(JSON_objects)_from_(2005-12_to_2018-01).json"
00,069 Schizandra$(24) = "RHW, Random_House_Webster's_Unabridged_Dictionary_(En-En).dsl"
00,070 Schizandra$(25) = "SNT, Machine-Learning_WestburyLab.NonRedundant.UsenetCorpus_(47860_English_language_non-binary-file_news_groups).tar"
00,071 Schizandra$(26) = "STX, archive.org_stackexchange_(346_corpora_2017-Oct-12).tar"
00,072 Schizandra$(27) = "TAL, the-anarchist-library-2016-01-18-en.html.tar"
00,073 Schizandra$(28) = "TXF, TEXTFILES.COM_(58096_files).tar"
00,074 Schizandra$(29) = "URB, Machine-Learning_Urban_Dictionary_Definitions_Corpus_(1999_-_May-2016).words.json"
00,075 Schizandra$(30) = "WKD, dumps.wikimedia.org_Germany_dewiki-20180220-pages-articles.xml"
00,076 Schizandra$(31) = "WKE, dumps.wikimedia.org_English_enwiki-20180220-pages-articles.xml"
00,077 Schizandra$(32) = "WKF, dumps.wikimedia.org_France_frwiki-20180220-pages-articles.xml"
00,078 Schizandra$(33) = "WKI, dumps.wikimedia.org_Italy_itwiki-20180220-pages-articles.xml"
00,079 Schizandra$(34) = "WKN, dumps.wikimedia.org_Netherlands_nlwiki-20180220-pages-articles.xml"
00,080 Schizandra$(35) = "WKP, dumps.wikimedia.org_Portugal_ptwiki-20180220-pages-articles.xml"
00,081 Schizandra$(36) = "WKS, dumps.wikimedia.org_Spain_eswiki-20180220-pages-articles.xml"
00,082 Schizandra$(37) = "WMB, dumps.wikimedia.org_English_enwikibooks-20180220-pages-articles.xml"
00,083 Schizandra$(38) = "WMN, dumps.wikimedia.org_English_enwikinews-20180220-pages-articles.xml"
00,084 Schizandra$(39) = "WMP, dumps.wikimedia.org_English_specieswiki-20180220-pages-articles.xml"
00,085 Schizandra$(40) = "WMQ, dumps.wikimedia.org_English_enwikiquote-20180220-pages-articles.xml"
00,086 Schizandra$(41) = "WMS, dumps.wikimedia.org_English_enwikisource-20180220-pages-articles.xml"
00,087 Schizandra$(42) = "WMT, dumps.wikimedia.org_English_enwikiversity-20180220-pages-articles.xml"
00,088 Schizandra$(43) = "WMV, dumps.wikimedia.org_English_enwikivoyage-20180220-pages-articles.xml"
00,089 Schizandra$(44) = "WMW, dumps.wikimedia.org_English_enwiktionary-20180220-pages-articles.xml"
00,090 Schizandra$(45) = "WUD, Webster's_Unabridged_3_(En-En)_WHOLEWORDS_ANSI.dsl"
00,091
00,092 Dim Shared DCTNS$(1 To 23)
00,093 DCTNS$(1) = " 42,257,313 American_Heritage_Dictionary_4_(En-En)_UTF-8.dsl"
00,094 DCTNS$(2) = " 299,225,953 Britannica_Encyclopedia_2010_1.563_miled_(En-En)_ANSI.dsl"
00,095 DCTNS$(3) = " 75,122,799 Cambridge_Advanced_Learner's_Dictionary_4th_Ed_(En-En).dsl"
00,096 DCTNS$(4) = " 23,413,283 Collins_COBUILD_Advanced_Learner's_English_Dictionary_(5th_ed)_(En-En).dsl"
00,097 DCTNS$(5) = " 37,870,978 Longman_Activator_2nd_Ed_(En-En).dsl"
00,098 DCTNS$(6) = " 52,743,002 Longman_Dictionary_of_Contemporary_English_5th_Ed_(En-En).dsl"

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

Listing: MASAKARI_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus_ToshibaTxl2_8x16.ttf; Downloadable at: www.sanmayce.com/Masakari.zip
00,099 DCTNS$(7) = " 83,412,925 Macmillan_English_Dictionary_(En-En)_UTF-8.dsl"
00,100 DCTNS$(8) = " 29,772,872 Macmillan_English_Thesaurus_(En-En)_UTF-8.dsl"
00,101 DCTNS$(9) = " 101,528,058 Merriam-Webster's_Collegiate_Dictionary_11th_Edition_(En-En).dsl"
00,102 DCTNS$(10) = " 57,854,694 Oxford_Advanced_Learner's_Dictionary_8th_Edition_(En-En).dsl"
00,103 DCTNS$(11) = " 44,362,425 Oxford_American_Dictionary_2nd_Edition.dsl"
00,104 DCTNS$(12) = " 11,747,238 Oxford_American_Thesaurus_(En-En).dsl"
00,105 DCTNS$(13) = " 559,408,265 Oxford_English_Dictionary_2nd_Edition_Version_4_(En-En)_ANSI.dsl"
00,106 DCTNS$(14) = " 53,483,030 Random_House_Webster's_Unabridged_Dictionary_(En-En)_UTF-8.dsl"
00,107 DCTNS$(15) = " 12,202,934 The_Collins_Cobuild_School_Dictionary_of_American_English_(En-En).dsl"
00,108 DCTNS$(16) = " 137,754,899 Webster's_Unabridged_3_(En-En)_UTF-8.dsl"
00,109 DCTNS$(17) = " 5,340,386 dict2txt_Online_Etymology_Dictionary1.1.txt"
00,110 DCTNS$(18) = "7,015,793,795 enwiktionary-20210920-pages-articles.xml"
00,111 DCTNS$(19) = "1,917,822,288 Machine-Learning_Urban_Dictionary_Definitions_Corpus_(1999_-_May-2016).words.json"
00,112 DCTNS$(20) = " 148,865,399 [French_-_French]_Emile_Littr_UTF-8.mdx.txt"
00,113 DCTNS$(21) = " 176,073,762 [French_-_French]_GrandRobert_UTF-8.mdx.txt"
00,114 DCTNS$(22) = " 97,807,080 [French_-_French]_Larousse_Multidico_UTF-8.mdx.txt"
00,115 DCTNS$(23) = " 282,443,452 [French_-_French]_PetitRobert2007_UTF-8.mdx.txt"
00,116
00,117 _Define A-Z As _INTEGER64
00,118
00,119 Mrev$ = "8.1++"
00,120
00,121 'Set compile-time codepaths:
00,122 'Vanilla (WRAPPERV = 0) or Wrapper (WRAPPERV = 1):
00,123 $Let WRAPPERV = 0
00,124 'Toshiba 8x16 (BIGORSMALL = 1) or NEC 6x12 (BIGORSMALL = 0):
00,125 $Let BIGORSMALL = 1
00,126
00,127 $If WRAPPERV = 0 Then
00,128     WrapFlag = 0 ' 1 means wrapping; 0 means vanilla
00,129 $ElseIf WRAPPERV = 1 Then
00,130     WrapFlag = 1 ' 1 means wrapping; 0 means vanilla
00,131 $Else
00,132 $End If
00,133
00,134 FONT4x = 0 '32px in 98x30
00,135 ToLoadOrNotFlag = 0 ' Should be 0; OLD: 1 means fast load but memory greedy; 0 means slow load but memory efficient
00,136
00,137 ShutDownDuration# = 4 ' 4 seconds
00,138 DclickTime# = 0.33 ' 1/3 of a second, usually it is 0.27s, so 0.33 suits even the slow clickers
00,139 Dim Button1LOG_firstDetection#(4)
00,140 Dim Button1LOG_ForHowLongHeld#(4)
00,141 Dim Button2LOG_firstDetection#(4)
00,142 Dim Button2LOG_ForHowLongHeld#(4)
00,143 PrevClick1# = 0
00,144 PrevClick2# = 0
00,145
00,146 Dim KeyTap_Button1LOG_firstDetection#(4)
00,147 Dim KeyTap_Button1LOG_ForHowLongHeld#(4)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,148 Dim KeyTap_Button2LOG_firstDetection#(4)
00,149 Dim KeyTap_Button2LOG_ForHowLongHoled#(4)
00,150 Dim KeyTap_Button3LOG_firstDetection#(4)
00,151 Dim KeyTap_Button3LOG_ForHowLongHoled#(4)
00,152 Dim KeyTap_Button4LOG_firstDetection#(4)
00,153 Dim KeyTap_Button4LOG_ForHowLongHoled#(4)
00,154 KeyTap_PrevClick1# = 0
00,155 KeyTap_PrevClick2# = 0
00,156 KeyTap_PrevClick3# = 0
00,157 KeyTap_PrevClick4# = 0
00,158
00,159 _ControlChr Off 'without it couldn't _MAPUNICODE these 7, 9..13, 28..31
00,160
00,161 If InStr(LCase$(Command$), "/help") Or InStr(LCase$(Command$), "-h") Then
00,162     $ScreenHide
00,163     $Console
00,164     _Console On
00,165     _ConsoleTitle "Masakari console window"
00,166     _Dest _Console
00,167
00,168     ShowASCIIart
00,169
00,170     '
00,171     ' |_____|
00,172     ' |  /  ' |
00,173     ' |  /  ' |
00,174     ' |  /  ' |
00,175     ' |  /  ' |
00,176     ' |  /  ' |
00,177     ' |  /  ' |
00,178     ' |  /  ' |
00,179     ' |  /  ' |
00,180     ' |  /  ' |
00,181     ' |  /  ' |
00,182     ' |  /  ' |
00,183     ' |  /  ' |
00,184     ' |  /  ' |
00,185     ' |  /  ' |
00,186     ' |  /  ' |
00,187     ' |  /  ' |
00,188     ' |  /  ' |
00,189     ' |  /  ' |
00,190     ' |  /  ' |
00,191     ' |  /  ' |
00,192     ' |  /  ' |
00,193     ' |  /  ' |
00,194     ' |  /  ' |
00,195     ' |  /  ' |
00,196     ' |  /  ' |

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,197 ' / / |
00,198 ' \ \ |
00,199 '   |
00,200 '   |
00,201 ' | |
00,202 ' | |
00,203 ' | |
00,204 ' | |
00,205 ' | |
00,206 ' | |
00,207 ' | |
00,208 ' | |
00,209 ' | |
00,210 ' | |
00,211 ' | |
00,212 ' | |
00,213 ' | |
00,214 ' | |
00,215 ' | |
00,216 ' | |
00,217 ' | |
00,218 ' | |
00,219 ' | |
00,220 ' | |
00,221 ' | |
00,222 ' | |
00,223 ' | |
00,224 ' | |
00,225 ' | |
00,226 ' | |
00,227 ' | |
00,228 ' | |
00,229 ' | |
00,230 ' | |
00,231 ' | |
00,232 ' | |
00,233 ' | |
00,234 ' | |
00,235 ' | |
00,236 ' | |
00,237 ' | |
00,238 ' | |
00,239 ' | |
00,240 ' | |
00,241 ' | |
00,242 ' | |
00,243 ' | |
00,244 ' | |
00,245 ' | |

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,295
00,296 Rem https://www.gb64.org/forum/index.php?topic=4307.msg137147#msg137147
00,297 'Dim Shared memstat As MEMORYSTATUSEX
00,298 'Dim Shared Result As _Integer64
00,299 'Type MEMORYSTATUSEX
00,300 '     dwLength As Long
00,301 '     dwMemoryLoad As Long
00,302 '     ullTotalPhys As _Unsigned _Integer64
00,303 '     ullAvailPhys As _Unsigned _Integer64
00,304 '     ullTotalPageFile As _Unsigned _Integer64
00,305 '     ullAvailPageFile As _Unsigned _Integer64
00,306 '     ullTotalVirtual As _Unsigned _Integer64
00,307 '     ullAvailVirtual As _Unsigned _Integer64
00,308 '     ullAvailExtendedVirtual As _Unsigned _Integer64
00,309 'End Type
00,310 'Declare CustomType Library 'Memory Information using KERNEL32
00,311 '     Function GlobalMemoryStatusEx& (Mstat As MEMORYSTATUSEX)
00,312 'End Declare
00,313 'GoSub Free_MEM:
00,314 'End
00,315 'Free_MEM:
00,316 'memstat.dwLength = Len(memstat): Result = GlobalMemoryStatusEx&(memstat)
00,317 'Free_MEM& = memstat.ullAvailPhys: Print "freeMEM=- ";: Print Using "###,###,###,###"; memstat.ullAvailPhys;: Print " bytes";
00,318 'Return
00,319
00,320 Declare CustomType Library "qsm" ' Notice that 'CustomType' makes things work, using 'STATIC' gives errors during compilation
00,321 'Sub Quicksort_QB64_v7 (ByVal QWORDS As _Offset, ByVal Left As _Integer64, ByVal Right As _Integer64)
00,322 End Declare
00,323
00,324 Declare CustomType Library "qsm_linesize" ' Notice that 'CustomType' makes things work, using 'STATIC' gives errors during compilation
00,325 'Sub Quicksort_QB64_v7_linesize (ByVal QWORDSoft As _Offset, ByVal QWORDSlent As _Offset, ByVal Left As _Integer64, ByVal Right As _Integer64)
00,326 End Declare
00,327 'Quicksort_QB64_v7_linesize MhandleOFF.OFFSET, MhandleLEN.OFFSET, 0, ElementsMinusOne
00,328
00,329 'Declare CustomType Library
00,330 'Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,331 'Function memcmp% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,332 'End Declare
00,333 'a$ = "1234567890"
00,334 'b$ = "ABCDEFGHIJ"
00,335 'memcpy _Offset(a$) + 5, _Offset(b$) + 5, 5
00,336 'Print: Print a$: Print b$
00,337 'SignedInt% = memcmp%(_Offset(a$), _Offset(b$), 5) ' 1 vs A returns -1
00,338 'Print Mid$(a$, 1, 1); " vs "; Mid$(b$, 1, 1); " returns "; SignedInt%
00,339 'SignedInt% = memcmp%(_Offset(b$), _Offset(a$), 5) ' A vs 1 returns 1
00,340 'Print Mid$(b$, 1, 1); " vs "; Mid$(a$, 1, 1); " returns "; SignedInt%
00,341 'SignedInt% = memcmp%(_Offset(a$) + 5, _Offset(b$) + 5, 5) ' F vs F returns 0
00,342 'Print Mid$(a$, 1 + 5, 1); " vs "; Mid$(b$, 1 + 5, 1); " returns "; SignedInt%
00,343

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,344 'On Linux, the compile line should be expanded from:
00,345 'qb64\internal\c\makeline_lnx.txt:
00,346 'g++ -no-pie -w qbx.cpp parts/core/os/lnx/src.a -lGL -lGLU -lX11 -lpthread -ldl -lrt -D FREEGLUT_STATIC -o
00,347 'to:
00,348 'g++ -O3 -mavx2 -maes -no-pie -w qbx.cpp parts/core/os/lnx/src.a -lGL -lGLU -lX11 -lpthread -ldl -lrt -D FREEGLUT_STATIC -o
00,349
00,350 '// manatarka.h:
00,351 '// In order to compile properly, edit the next line from E:\qb64\internal\temp\recompile_win.bat:
00,352 '// c_compiler\bin\g++ -s -Wfatal-errors ...
00,353 '// Insert "-O3 -msse4.1 -maes", or even better "-O3 -mavx2 -maes" for YMM variants, so:
00,354 '// c_compiler\bin\g++ -O3 -mavx2 -maes -s -Wfatal-errors ...
00,355
00,356 'Declare CustomType Library
00,357 '    Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,358 '    Function memcmp% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,359 'End Declare
00,360 'a$ = "1234567890"
00,361 'b$ = "ABCDEFGHJIJ"
00,362 'memcpy _Offset(a$) + 5, _Offset(b$) + 5, 5
00,363 'Print: Print a$: Print b$
00,364 'SignedInt% = memcmp%(_Offset(a$), _Offset(b$), 5) ' 1 vs A returns -1
00,365 'Print Mid$(a$, 1, 1); " vs "; Mid$(b$, 1, 1); " returns "; SignedInt%
00,366 'SignedInt% = memcmp%(_Offset(b$), _Offset(a$), 5) ' A vs 1 returns 1
00,367 'Print Mid$(b$, 1, 1); " vs "; Mid$(a$, 1, 1); " returns "; SignedInt%
00,368 'SignedInt% = memcmp%(_Offset(a$) + 5, _Offset(b$) + 5, 5) ' F vs F returns 0
00,369 'Print Mid$(a$, 1 + 5, 1); " vs "; Mid$(b$, 1 + 5, 1); " returns "; SignedInt%
00,370
00,371 'Declare CustomType Library "manatarka"
00,372 '    Sub UCASE_XMM (ByVal QWORDSrc As _Offset, ByVal QWORDDst As _Offset, ByVal bytesto As _Integer64)
00,373 '    Sub UCASE_XMM_inplace (ByVal QWORDSrc As _Offset, ByVal bytesto As _Integer64)
00,374 '    Function Railgun_Doublet%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer arithmetic) is
to define HaystackLEN as _Offset instead of LONG
00,375 '    Function Railgun_Trolldom_64%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer
arithmetic) is to define HaystackLEN as _Offset instead of _Integer64
00,376 '    Function Railgun_Nyotengu_XMM_YMM_ZMM%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer
arithmetic) is to define HaystackLEN as _Offset instead of _Integer64
00,377 '    'char * Railgun_Doublet (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern)
00,378 '    'char * Railgun_Trolldom_64 (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern)
00,379 '    'char * Railgun_Nyotengu_XMM_YMM_ZMM (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern) // Last change: 2020-Nov-30
00,380 '    Sub DoubleDeuceAES_Gumbotron (ByVal buffer As _Offset, ByVal bytesto As _Integer64, ByVal LoPart As _Offset, ByVal HiPart As _Offset) 'Note: This hasher uses AES extension, the CPU has to support it!
00,381 '    Sub DoubleDeuceAES_Gumbotron_YMM (ByVal buffer As _Offset, ByVal bytesto As _Integer64, ByVal LoPart As _Offset, ByVal HiPart As _Offset) 'Note: This hasher uses AES extension, the CPU has to support it!
00,382 '    'void DoubleDeuceAES_Gumbotron (const uint8_t *buffer, size_t length, uint64_t *LoPart, uint64_t *HiPart) {
00,383 'End Declare
00,384
00,385 'Declare CustomType Library "memKAZE"
00,386 '    Sub memcpyKAZE (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,387 '    Function memcmpKAZE% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,388 'End Declare
00,389

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,390 ' CAUTION: With next 4 lines, all is OK ... under Windows only, for some reason on Linux it works sometime, sometime not?!
00,391 'Declare CustomType Library
00,392 '   Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,393 '   Function memcmp% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,394 'End Declare
00,395
00,396
00,397 ' Examples [::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::]
00,398 'DoubleDeuceAES_Gumbotron_YMM _Offset(TestKey$), Len(TestKey$), _Offset(LoPart), _Offset(HiPart)
00,399
00,400 'MhandleXMM = MemNew(LinesTestLookups + 888 * (LinesTestLookups * 2)) ' BYTE+QWORD+QWORD=17
00,401 'ts# = Timer(.001)
00,402 'LinesTestLookupsUploading = 0
00,403 'For i = 1 To LinesTestLookups
00,404 '   DoubleDeuceAES_Gumbotron_YMM _Offset(NoNeedOfArrayButForBench$(i)), Len(NoNeedOfArrayButForBench$(i)), _Offset(LoPart), _Offset(HiPart)
00,405 '   KeyLen = Len(NoNeedOfArrayButForBench$(i))
00,406 '   If KeyLen >= 256 Then KeyLen = 0
00,407 '   memcpy MhandleXMM.OFFSET + (LinesTestLookupsUploading + 888 * (LinesTestLookupsUploading * 2)) + 0, _Offset(KeyLen), 1
00,408 '   _MemPut MhandleXMM, MhandleXMM.OFFSET + (LinesTestLookupsUploading + 888 * (LinesTestLookupsUploading * 2)) + 1, LoPart
00,409 '   _MemPut MhandleXMM, MhandleXMM.OFFSET + (LinesTestLookupsUploading + 888 * (LinesTestLookupsUploading * 2)) + 1 + 8, HiPart
00,410 '   LinesTestLookupsUploading = LinesTestLookupsUploading + 1
00,411 'Next
00,412 'te# = Timer(.001)
00,413
00,414 'For i = 1 To LinesTestLookups
00,415 '   memcpy _Offset(KeyLen), MhandleXMM.OFFSET + ((i - 1) + 888 * ((i - 1) * 2)) + 0, 1
00,416 '   _MemGet MhandleXMM, MhandleXMM.OFFSET + ((i - 1) + 888 * ((i - 1) * 2)) + 1, LoPart
00,417 '   _MemGet MhandleXMM, MhandleXMM.OFFSET + ((i - 1) + 888 * ((i - 1) * 2)) + 1 + 8, HiPart
00,418 'Next
00,419
00,420 'Hits = 0
00,421 'FoundAt = 1
00,422 'Do
00,423 '   FoundAt = InStr(FoundAt, Haystack$, Needle$)
00,424 '   If FoundAt = 0 Then Exit Do Else FoundAt = FoundAt + 1: Hits = Hits + 1
00,425 'Loop
00,426
00,427 'Hits = 0
00,428 'HaystackLEN% = Len(Haystack$) 'The workaround of lacking casting
00,429 'NeedleLEN = Len(Needle$)
00,430 'PointerHQB64% = _Offset(Haystack$) 'Dim PointerHQB64 As _Offset
00,431 'PointerNQB64% = _Offset(Needle$) 'Dim PointerNQB64 As _Offset
00,432 'FoundAt% = PointerHQB64%
00,433 'Do
00,434 '   FoundAtNew% = Bailgun_Trolldom_64%(FoundAt%, PointerNQB64%, HaystackLEN% - (FoundAt% - PointerHQB64%), NeedleLEN)
00,435 '   If FoundAtNew% = 0 Then Exit Do
00,436 '   FoundAt% = FoundAtNew% + 1: Hits = Hits + 1
00,437 'Loop
00,438

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,488 If XdimCOL = 213 And YdimROW = 60 Then _Title "MASAKARI, revision " + Mrev$ + " 213x60, The 'Holy Axe' English Text Sidekick"
00,489
00,490 Mode4K = 0 'Make it 1 or 0, if 1 then 16 becomes 32 i.e. the font is 4x bigger: 8x16 becomes 16x32
00,491 If wide& >= 1920 * 2 Then Mode4K = 1
00,492
00,493 If FONT4x = 1 Then
00,494     XdimCOL = 112
00,495     YdimROW = 30 ' ensure old laptops with 768pixels vertical will hold the whole window
00,496     If XdimCOL = 112 And YdimROW = 30 Then _Title "MASAKARI, revision " + Mrev$ + " 112x30, The 'Holy Axe' English Text Sidekick"
00,497 End If
00,498
00,499 'Mode4K = 1: YdimROW = YdimROW \ 2: XdimCOL = 96 'debug; Enforcing 32px on 1680 or FHD is good, however status line should be shrunk, also Search Panel need 40lines not 30!
00,500
00,501 ASCIIFrame = 0
00,502 ASCIIFramePEN = 0
00,503
00,504 'EpochStepping.bas
00,505 FramesPerSecond = 500
00,506 LastHit~&& = EpochTime~&& 'assume the start point is at 100%
00,507 Stepping = 10 '0..10 where 10 is 100%, 9 is 90%
00,508
00,509 $If WIN Then
00,510     Const Slash = "\"
00,511 $Else
00,512     CONST Slash = "/"
00,513 $End If
00,514
00,515 $If WINDOWS Then
00,516     PSPlike$ = _CWD$ + "\"
00,517 $Else
00,518     PSPlike$ = _CWD$ + "/"
00,519 $End If
00,520
00,521 'Note1: If the Toshiba (MxPlus_ToshibaTxL2_8x16.ttf) font is not in the starting folder then JPN 6x12 (Px437_DOS-V_re_JPN12.ttf) is loaded instead (if not 4K). On old laptops e.g. 1366x768 it can show the "standard
128x60".
00,522 'Note2: Trying to enforce the maximum, if (4K is 0) and (Toshiba is missing):
00,523 $If BIGORSMALL = 1 Then
00,524     If _FileExists(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf") = 0 Then
00,525         If Mode4K = 0 Then
00,526             If wide& >= 1920 Then
00,527                 XdimCOL = 300 ' 300x6< (FHDx=1920)
00,528                 YdimROW = 80 ' 80x12< (FHDy=1080)
00,529                 _Title "MASAKARI, revision " + Mrev$ + " 300x80, The 'Holy Axe' English Text Sidekick"
00,530             End If
00,531         End If
00,532     End If
00,533 $Else
00,534     'IF _FILEEXISTS(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf") = 0 THEN
00,535     IF Mode4K = 0 THEN

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,536 IF wide& >= 1920 THEN
00,537   XdimCOL = 300 ' 300x6< (FHDx=1920)
00,538   YdimROW = 80 ' 80x12< (FHDy=1080)
00,539   _TITLE "MASAKARI, revision "+Mrev$+" 300x80, The 'Holy Axe' English Text Sidekick"
00,540 END IF
00,541 END IF
00,542 'END IF
00,543 $End If
00,544
00,545 '' SOUND PLAYER [
00,546 's& = _SNDOPEN(PSPlike$+"music_zapsplat_rapid_turnaround.mp3")
00,547 's& = _SNDOPEN(PSPlike$+"music_pioxonaq_agressive_lightning.mp3")
00,548 IF s& <> 0 THEN
00,549 '   _SNDPLAY s& 'check for valid handle before using!
00,550 '   'The _SNDLOOP statement is like _SNDPLAY but the sound is looped. Uses a handle from the _SNDOPEN function.
00,551 '   'Example 1: Playing a previously opened sound at half volume.
00,552 '   '_SNDPLAYCOPY applause&, 0.5
00,553 '   volumeINT% = 9 '10 means 100% means 1.0, down to 0 i.e. 0..10 or 0.0.1,...1.0
00,554 '   StopFlag = 0
00,555 '   DO
00,556 '       k$ = INKEY$
00,557 '       SELECT CASE k$
00,558 '           CASE "n"
00,559 '               IF volumeINT% >= 1 THEN volumeINT% = volumeINT% - 1
00,560 '               _SNDVOL s&, volumeINT% / 10
00,561 '           CASE "m"
00,562 '               IF volumeINT% <= 8 THEN volumeINT% = volumeINT% + 1 ' don't go 100% but up to 90%
00,563 '               _SNDVOL s&, volumeINT% / 10
00,564 '           CASE "p"
00,565 '               _SNDPAUSE s&
00,566 '           CASE " "
00,567 '               _SNDPLAY s&
00,568 '           CASE CHR$(27)
00,569 '               _SNDSTOP s&
00,570 '               _SNDCLOSE s&
00,571 '               StopFlag = 1
00,572 '           END SELECT
00,573 '           LOCATE 1, 1
00,574 '           IF StopFlag = 0 THEN PRINT "Volume: "; volumeINT%; "; Position: "; INT(_SNDGETPOS(s&)); "second, up to "; INT(_SNDLEN(s&))
00,575 '       LOOP UNTIL StopFlag
00,576 '   END IF
00,577 '' SOUND PLAYER ]
00,578
00,579 'spell [
00,580 Dim lfixed As String * 32
00,581 TotalWrd& = 0
00,582 Open PSPlike$ + "masakari.ind" For Binary As #2
00,583 If LOF(2) Then
00,584     TotalWrd& = LOF(2) / 32

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,585      Close #2
00,586 Else
00,587      Close #2
00,588      Kill PSPlike$ + "masakari.ind"
00,589      Open PSPlike$ + "masakari.wrd" For Binary As #2
00,590      If LOF(2) Then
00,591          Close #2
00,592          Open PSPlike$ + "masakari.wrd" For Input As #1
00,593          Open PSPlike$ + "masakari.ind" For Random Access Write As #2 Len = 32
00,594          Do While Not EOF(1)
00,595              Line Input #1, l$
00,596              lfixed$ = l$
00,597              Put #2, , lfixed$: TotalWrd& = TotalWrd& + 1
00,598              FL2wrd& = 1
00,599          Loop
00,600      Close #1, #2
00,601      Else
00,602          Close #2
00,603          Kill PSPlike$ + "masakari.wrd"
00,604      End If
00,605 End If
00,606 'spell ]
00,607
00,608 If InStr(LCase$(Command$), "/ascii") Or InStr(LCase$(Command$), "-ascii") Then
00,609 Else
00,610     GoTo SkipFontReviews
00,611 End If
00,612 'SCREEN 0 'It is better to define your own mode, as:
00,613 handle& = _NewImage(98, 30, 0)
00,614 Screen handle&
00,615 _Dest handle&
00,616
00,617 '_FONT_LOADFONT("C:\windows\fonts\cour.ttf", 32, "MONOSPACE")
00,618 '_FONT_LOADFONT("C:\windows\fonts\lucon.ttf", 32, "MONOSPACE")
00,619 '_FONT_LOADFONT("C:\windows\fonts\MxPlus_Cordata_PPC-400.ttf", 32, "MONOSPACE")
00,620 '_FONT_LOADFONT("C:\windows\fonts\MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
00,621 _Font_LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
00,622
00,623 'RESTORE Microsoft_pc_cp437 'United States MS DOS
00,624 'RESTORE Microsoft_windows_cp1250 'WINDOWS in Central European and Eastern European languages that use Latin script, such as Polish, Czech, Slovak, Hungarian, Slovene, Bosnian, Croatian, Serbian (Latin script), Romanian
and Albanian. It may also be used with the German language.
00,625 'RESTORE Microsoft_windows_cp1251 'Cyrillic alphabet such as Russian, Bulgarian, Serbian Cyrillic and other languages. It is the most widely used for encoding the Bulgarian, Serbian and Macedonian languages.
00,626 'RESTORE Microsoft_pc_cpMIK 'Cyrillic Bulgarian Pravetz 16 for MS-DOS
00,627 Color 4
00,628 If InStr(LCase$(Command$), "/ascii_gesch") Or InStr(LCase$(Command$), "-ascii_gesch") Then
00,629     Restore Microsoft_pc_cpGESCH 'My codepage a.k.a. Gesch
00,630     Print "Showing GRAPHEMES of Gesch (a.k.a. Schpitz) codepage,": Print "it allows browsing German/Italian/French/Spanish/Bulgarian (and its dialect Russian :P):"
00,631 Else
00,632     Restore Microsoft_windows_cp1252

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,633   Print "Showing GRAPHEMES of cp1252 codepage - Windows Western languages with Latin alphabet,";
00,634   Print "it allows browsing... ugh, it sucks:"
00,635 End If
00,636
00,637 If InStr(LCase$(Command$), "/ascii_gesch") Or InStr(LCase$(Command$), "-ascii_gesch") Then
00,638   For ASCIIcode = 0 To 6 'German A:a:O:o:U:u: ss
00,639     Read unicode
00,640     _MapUnicode unicode To ASCIIcode
00,641   Next
00,642   For ASCIIcode = 7 To 7 'Integral high
00,643     Read unicode
00,644     _MapUnicode unicode To ASCIIcode
00,645   Next
00,646   For ASCIIcode = 8 To 8 'Integral low
00,647     Read unicode
00,648     _MapUnicode unicode To ASCIIcode
00,649   Next
00,650   For ASCIIcode = 14 To 15 'French C,c,
00,651     Read unicode
00,652     _MapUnicode unicode To ASCIIcode
00,653   Next
00,654   For ASCIIcode = 16 To 27 'French/Italian E'e' A'a'E'e'I'i'O'o'U'u'
00,655     Read unicode
00,656     _MapUnicode unicode To ASCIIcode
00,657   Next
00,658   'why 28..31 are not available?!
00,659   For ASCIIcode = 128 To 255
00,660     Read unicode
00,661     _MapUnicode unicode To ASCIIcode
00,662   Next
00,663   For ASCIIcode = 220 To 223 ' the unnecessary 4 bold drawing chars are replaced with French AE ae I: i:
00,664     Read unicode
00,665     _MapUnicode unicode To ASCIIcode
00,666   Next
00,667   For ASCIIcode = 127 To 127 'almost equal to
00,668     Read unicode
00,669     _MapUnicode unicode To ASCIIcode
00,670   Next
00,671   For ASCIIcode = 181 To 190 'Cyrillic short vowels
00,672     Read unicode
00,673     _MapUnicode unicode To ASCIIcode
00,674   Next
00,675   For ASCIIcode = 198 To 207 'Cyrillic short vowels
00,676     Read unicode
00,677     _MapUnicode unicode To ASCIIcode
00,678   Next
00,679   For ASCIIcode = 208 To 216 'misc, Spanish
00,680     Read unicode
00,681     _MapUnicode unicode To ASCIIcode

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,682 Next
00,683 For ASCIIcode = 28 To 31 'draw animated block
00,684 Read unicode
00,685 _MapUnicode unicode To ASCIIcode
00,686 Next
00,687 For ASCIIcode = 11 To 12 'draw animated block
00,688 Read unicode
00,689 _MapUnicode unicode To ASCIIcode
00,690 Next
00,691 End If
00,692
00,693 toggleCyanPurpleWhite = 3
00,694 Color toggleCyanPurpleWhite
00,695 If cnt Mod 16 = 0 Then Print: Print String$(3 - Len(LTrim$(Str$(cnt))), "0") + LTrim$(Str$(cnt)); " ";
00,696 For i = 0 To 255
00,697 'IF i <> 11 AND i <> 12 AND i <> 9 AND i <> 7 AND i <> 13 AND i <> 10 THEN
00,698 Print Chr$(i) + " ";
00,699 'ELSE
00,700 'PRINT " ";
00,701 'END IF
00,702 cnt = cnt + 1
00,703 If (cnt <= 255) And (cnt Mod 16 = 0) Then
00,704 If toggleCyanPurpleWhite = 3 Then
00,705 toggleCyanPurpleWhite = 7
00,706 ElseIf toggleCyanPurpleWhite = 7 Then
00,707 toggleCyanPurpleWhite = 9
00,708 Else
00,709 toggleCyanPurpleWhite = 3
00,710 End If
00,711 Color toggleCyanPurpleWhite
00,712 Print: Print String$(3 - Len(LTrim$(Str$(cnt))), "0") + LTrim$(Str$(cnt)); " ";
00,713 End If
00,714 Next
00,715 Color 4
00,716 If InStr(LCase$(Command$), "/ascii_gesch") Or InStr(LCase$(Command$), "-ascii_gesch") Then
00,717 Print: Print: Print "Example (forming a long integral from 209 179 210): ": Print Chr$(209); " Useful for": Print Chr$(179); " making paragraphs": Print Chr$(210); " of your own."
00,718 Else
00,719 Print: Print: Print "Example (forming a long integral from 244 179 245): ": Print Chr$(244); " Useful for": Print Chr$(179); " making paragraphs": Print Chr$(245); " of your own."
00,720 End If
00,721 Color 7
00,722 End
00,723
00,724 Microsoft_pc_cp737: 'Greek MS DOS displays Greek alphabet for algebraic formulas.
00,725 Data 913,914,915,916,917,918,919,920,921,922,923,924,925,926,927,928
00,726 Data 929,931,932,933,934,935,936,937,945,946,947,948,949,950,951,952
00,727 Data 953,954,955,956,957,958,959,960,961,963,962,964,965,966,967,968
00,728 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,729 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,730 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

00,731 Data 969,940,941,942,970,943,972,973,971,974,902,904,905,906,908,910

00,732 Data 911,177,8805,8804,938,939,247,8776,176,8729,183,8730,8319,178,9632,160

00,733

00,734 Microsoft\_pc\_cp775: 'Estonian, Lithuanian and Latvian languages.

00,735 Data 262,252,233,257,228,291,229,263,322,275,342,343,299,377,196,197

00,736 Data 201,230,198,333,246,290,162,346,347,214,220,248,163,216,215,164

00,737 Data 256,298,243,379,380,378,8221,166,169,174,172,189,188,321,171,187

00,738 Data 9617,9618,9619,9474,9508,260,268,280,278,9571,9553,9559,9565,302,352,9488

00,739 Data 9492,9524,9516,9500,9472,9532,370,362,9562,9556,9577,9574,9568,9552,9580,381

00,740 Data 261,269,281,279,303,353,371,363,382,9496,9484,9608,9604,9612,9616,9600

00,741 Data 211,223,332,323,245,213,181,324,310,311,315,316,326,274,325,8217

00,742 Data 173,177,8220,190,182,167,247,8222,176,8729,183,185,179,178,9632,160

00,743

00,744 Microsoft\_pc\_cp850: 'Western Europe, Spain, England

00,745 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,236,196,197

00,746 Data 201,230,198,244,246,242,251,249,255,214,220,248,163,216,215,402

00,747 Data 225,237,243,250,241,209,170,186,191,174,172,189,188,161,171,187

00,748 Data 9617,9618,9619,9474,9508,193,194,192,169,9571,9553,9559,9565,162,165,9488

00,749 Data 9492,9524,9516,9500,9472,9532,227,195,9562,9556,9577,9574,9568,9552,9580,164

00,750 Data 240,208,202,203,200,305,205,206,207,9496,9484,9608,9604,166,204,9600

00,751 Data 211,223,212,210,245,213,181,254,222,218,219,217,253,221,175,180

00,752 Data 173,177,8215,190,182,167,247,184,176,168,183,185,179,178,9632,160

00,753

00,754 Microsoft\_pc\_cp852: 'Central European languages that use Latin script such as Bosnian, Croatian, Czech, Hungarian, Polish, Romanian, Serbian or Slovak.

00,755 Data 199,252,233,226,228,367,263,231,322,235,336,337,238,377,196,262

00,756 Data 201,313,314,244,246,317,318,346,347,214,220,356,357,321,215,269

00,757 Data 225,237,243,250,260,261,381,382,280,281,172,378,268,351,171,187

00,758 Data 9617,9618,9619,9474,9508,193,194,282,350,9571,9553,9559,9565,379,380,9488

00,759 Data 9492,9524,9516,9500,9472,9532,258,259,9562,9556,9577,9574,9568,9552,9580,164

00,760 Data 273,272,270,203,271,327,205,206,283,9496,9484,9608,9604,354,366,9600

00,761 Data 211,223,212,323,324,328,352,353,340,218,341,368,253,221,355,180

00,762 Data 173,733,731,711,728,167,247,184,176,168,729,369,344,345,9632,160

00,763

00,764 Microsoft\_pc\_cp855: 'Cyrillic code page to be used under MS-DOS

00,765 Data 1106,1026,1107,1027,1105,1025,1108,1028,1109,1029,1110,1030,1111,1031,1112,1032

00,766 Data 1113,1033,1114,1034,1115,1035,1116,1036,1118,1038,1119,1039,1102,1070,1098,1066

00,767 Data 1072,1040,1073,1041,1094,1062,1076,1044,1077,1045,1092,1060,1075,1043,171,187

00,768 Data 9617,9618,9619,9474,9508,1093,1061,1080,1048,9571,9553,9559,9565,1081,1049,9488

00,769 Data 9492,9524,9516,9500,9472,9532,1082,1050,9562,9556,9577,9574,9568,9552,9580,164

00,770 Data 1083,1051,1084,1052,1085,1053,1086,1054,1087,9496,9484,9608,9604,1055,1103,9600

00,771 Data 1071,1088,1056,1089,1057,1090,1058,1091,1059,1078,1046,1074,1042,1100,1068,8470

00,772 Data 173,1099,1067,1079,1047,1096,1064,1101,1069,1097,1065,1095,1063,167,9632,160

00,773

00,774 Microsoft\_pc\_cp857: 'Turkish MS DOS

00,775 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,305,196,197

00,776 Data 201,230,198,244,246,242,251,249,304,214,220,248,163,216,350,351

00,777 Data 225,237,243,250,241,209,286,287,191,174,172,189,188,161,171,187

00,778 Data 9617,9618,9619,9474,9508,193,194,192,169,9571,9553,9559,9565,162,165,9488

00,779 Data 9492,9524,9516,9500,9472,9532,227,195,9562,9556,9577,9574,9568,9552,9580,164

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,780 Data 186,170,202,203,200,0,205,206,207,9496,9484,9608,9604,166,204,9600
00,781 Data 211,223,212,210,245,213,181,0,215,218,219,217,236,255,175,180
00,782 Data 173,177,0,190,182,167,247,184,176,168,183,185,179,178,9632,160
00,783
00,784 Microsoft_pc_cp860: 'Portuguese language. MS DOS
00,785 Data 199,252,233,226,227,224,193,231,234,202,232,205,212,236,195,194
00,786 Data 201,192,200,244,245,242,218,249,204,213,220,162,163,217,8359,211
00,787 Data 225,237,243,250,241,209,170,186,191,210,172,189,188,161,171,187
00,788 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,789 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,790 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,791 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,792 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,793
00,794 Microsoft_pc_cp861: 'Icelandic language (as well as other Nordic languages). MS DOS
00,795 Data 199,252,233,226,228,224,229,231,234,235,232,208,240,222,196,197
00,796 Data 201,230,198,244,246,254,251,221,253,214,220,248,163,216,8359,402
00,797 Data 225,237,243,250,193,205,211,218,191,8976,172,189,188,161,171,187
00,798 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,799 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,800 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,801 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,802 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,803
00,804 Microsoft_pc_cp862: 'Hebrew letters in positions 809A hex, but otherwise it is identical to CP437. Now obsolete, see CP1255
00,805 Data 1488,1489,1490,1491,1492,1493,1494,1495,1496,1497,1498,1499,1500,1501,1502,1503
00,806 Data 1504,1505,1506,1507,1508,1509,1510,1511,1512,1513,1514,162,163,165,8359,402
00,807 Data 225,237,243,250,241,209,170,186,191,8976,172,189,188,161,171,187
00,808 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,809 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,810 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,811 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,812 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,813
00,814 Microsoft_pc_cp863: 'French language (mainly in Canada). MS DOS
00,815 Data 199,252,233,226,194,224,182,231,234,235,232,239,238,8215,192,167
00,816 Data 201,200,202,244,203,207,251,249,164,212,220,162,163,217,219,402
00,817 Data 166,180,243,250,168,184,179,175,206,8976,172,189,188,190,171,187
00,818 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,819 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,820 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,821 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,822 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,823
00,824 Microsoft_pc_cp864: 'Arabic MS DOS
00,825 Data 176,183,8729,8730,9618,9472,9474,9532,9508,9516,9500,9524,9488,9484,9492,9496
00,826 Data 946,8734,966,177,189,188,8776,171,187,65271,65272,0,0,65275,65276,0
00,827 Data 160,173,65154,163,164,65156,0,0,65166,65167,65173,65177,1548,65181,65185,65189
00,828 Data 1632,1633,1634,1635,1636,1637,1638,1639,1640,1641,65233,1563,65201,65205,65209,1567
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

00,877 Data 160,171,172,321,164,260,166,167,168,169,350,171,172,173,174,379  
 00,878 Data 176,177,178,322,180,181,182,183,184,261,351,187,317,733,318,380  
 00,879 Data 340,193,194,258,196,313,262,199,268,201,280,203,282,205,206,270  
 00,880 Data 272,323,327,211,212,336,214,215,344,366,218,368,220,221,354,223  
 00,881 Data 341,225,226,259,228,314,263,231,269,233,281,235,283,237,238,271  
 00,882 Data 273,324,328,243,244,337,246,247,345,367,250,369,252,253,355,729  
 00,883

00,884 Microsoft\_windows\_cp1252: 'Windows Western languages with Latin alphabet, used by default in the legacy components of Microsoft Windows in English.

00,885 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,352,8249,338,0,381,0  
 00,886 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,353,8250,339,0,382,376  
 00,887 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175  
 00,888 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191  
 00,889 Data 192,193,194,195,196,197,198,199,200,201,202,203,204,205,206,207  
 00,890 Data 208,209,210,211,212,213,214,215,216,217,218,219,220,221,222,223  
 00,891 Data 224,225,226,227,228,229,230,231,232,233,234,235,236,237,238,239  
 00,892 Data 240,241,242,243,244,245,246,247,248,249,250,251,252,253,254,255  
 00,893

00,894 Microsoft\_windows\_cp1253: 'Greek (but not polytonic Greek) Not fully compatible with ISO 8859-7 (? is located differently).

00,895 Data 8364,0,8218,402,8222,8230,8224,8225,0,8240,0,8249,0,0,0,0  
 00,896 Data 0,8216,8217,8220,8221,8226,8211,8212,0,8482,0,8250,0,0,0,0  
 00,897 Data 160,901,902,163,164,165,166,167,168,169,0,171,172,173,174,8213  
 00,898 Data 176,177,178,179,900,181,182,183,904,905,906,187,908,189,910,911  
 00,899 Data 912,913,914,915,916,917,918,919,920,921,922,923,924,925,926,927  
 00,900 Data 928,929,0,931,932,933,934,935,936,937,938,939,940,941,942,943  
 00,901 Data 944,945,946,947,948,949,950,951,952,953,954,955,956,957,958,959  
 00,902 Data 960,961,962,963,964,965,966,967,968,969,970,971,972,973,974,0  
 00,903

00,904 Microsoft\_windows\_cp1254: 'Turkish

00,905 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,352,8249,338,0,0,0  
 00,906 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,353,8250,339,0,0,376  
 00,907 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175  
 00,908 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191  
 00,909 Data 192,193,194,195,196,197,198,199,200,201,202,203,204,205,206,207  
 00,910 Data 286,209,210,211,212,213,214,215,216,217,218,219,220,304,350,223  
 00,911 Data 224,225,226,227,228,229,230,231,232,233,234,235,236,237,238,239  
 00,912 Data 287,241,242,243,244,245,246,247,248,249,250,251,252,305,351,255  
 00,913

00,914 Microsoft\_windows\_cp1255: 'Hebrew Windows. Modern applications prefer [https://en.wikipedia.org/wiki/UTF-8 UTF-8] or UTF-16 http://www.fileformat.info/info/charset/UTF-16/list.htm to Windows 1255.

00,915 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,0,8249,0,0,0,0  
 00,916 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,0,8250,0,0,0,0  
 00,917 Data 160,161,162,163,8362,165,166,167,168,169,215,171,172,173,174,175  
 00,918 Data 176,177,178,179,180,181,182,183,184,185,247,187,188,189,190,191  
 00,919 Data 1456,1457,1458,1459,1460,1461,1462,1463,1464,1465,0,1467,1468,1469,1470,1471  
 00,920 Data 1472,1473,1474,1475,1520,1521,1522,1523,1524,0,0,0,0,0,0  
 00,921 Data 1488,1489,1490,1491,1492,1493,1494,1495,1496,1497,1498,1499,1500,1501,1502,1503  
 00,922 Data 1504,1505,1506,1507,1508,1509,1510,1511,1512,1513,1514,0,0,8206,8207,0  
 00,923

00,924 Microsoft\_windows\_cp1256: 'Arabic Latin Windows

00,925 Data 8364,1662,8218,402,8222,8230,8224,8225,710,8240,1657,8249,338,1670,1688,1672

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

00,926 Data 1711,8216,8217,8220,8221,8226,8211,8212,1705,8482,1681,8250,339,8204,8205,1722  
 00,927 Data 160,1548,162,163,164,165,166,167,168,169,1726,171,172,173,174,175  
 00,928 Data 176,177,178,179,180,181,182,183,184,185,1563,187,188,189,190,1567  
 00,929 Data 1729,1569,1570,1571,1572,1573,1574,1575,1576,1577,1578,1579,1580,1581,1582,1583  
 00,930 Data 1584,1585,1586,1587,1588,1589,1590,215,1591,1592,1593,1594,1600,1601,1602,1603  
 00,931 Data 224,1604,226,1605,1606,1607,1608,231,232,233,234,235,1609,1610,238,239  
 00,932 Data 1611,1612,1613,1614,244,1615,1616,247,1617,249,1618,251,252,8206,8207,1746

00,933

00,934 Microsoft\_windows\_cp1257: 'Estonian (although that can also be written with Windows-1252), Latvian and Lithuanian languages under Microsoft Windows. It is also possible to write Polish and German.

00,935 Data 8364,0,8218,0,8222,8230,8224,8225,0,8240,0,8249,0,168,711,184  
 00,936 Data 0,8216,8217,8220,8221,8226,8211,8212,0,8482,0,8250,0,175,731,0  
 00,937 Data 160,0,162,163,164,0,166,167,216,169,342,171,172,173,174,198  
 00,938 Data 176,177,178,179,180,181,182,183,248,185,343,187,188,189,190,230  
 00,939 Data 260,302,256,262,196,197,280,274,268,201,377,278,290,310,298,315  
 00,940 Data 352,323,325,211,332,213,214,215,370,321,346,362,220,379,381,223  
 00,941 Data 261,303,257,263,228,229,281,275,269,233,378,279,291,311,299,316  
 00,942 Data 353,324,326,243,333,245,246,247,371,322,347,363,252,380,382,729

00,943

00,944 Microsoft\_windows\_cp1258: 'Vietnamese. [https://en.wikipedia.org/wiki/UTF-8 UTF-8] is the preferred encoding for Vietnamese in modern applications.

00,945 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,0,8249,338,0,0,0  
 00,946 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,0,8250,339,0,0,376  
 00,947 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175  
 00,948 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191  
 00,949 Data 192,193,194,258,196,197,198,199,200,201,202,203,768,205,206,207  
 00,950 Data 272,209,777,211,212,416,214,215,216,217,218,219,220,431,771,223  
 00,951 Data 224,225,226,259,228,229,230,231,232,233,234,235,769,237,238,239  
 00,952 Data 273,241,803,243,244,417,246,247,248,249,250,251,252,432,8363,255

00,953

00,954 Microsoft\_pc\_cp437: 'United States MS DOS

00,955 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,236,196,197  
 00,956 Data 201,230,198,244,246,242,251,249,255,214,220,162,163,165,8359,402  
 00,957 Data 225,237,243,250,241,209,170,186,191,8976,172,189,188,161,171,187  
 00,958 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488  
 00,959 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575  
 00,960 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600  
 00,961 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745  
 00,962 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160

00,963

00,964 Microsoft\_windows\_cp1251: 'Cyrillic alphabet such as Russian, Bulgarian, Serbian Cyrillic and other languages. It is the most widely used for encoding the Bulgarian, Serbian and Macedonian languages.

00,965 Data 1026,1027,8218,1107,8222,8230,8224,8225,8364,8240,1033,8249,1034,1036,1035,1039  
 00,966 Data 1106,8216,8217,8220,8221,8226,8211,8212,0,8482,1113,8250,1114,1116,1115,1119  
 00,967 Data 160,1038,1118,1032,164,1168,166,167,1025,169,1028,171,172,173,174,1031  
 00,968 Data 176,177,1030,1110,1169,181,182,183,1105,8470,1108,187,1112,1029,1109,1111  
 00,969 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055  
 00,970 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071  
 00,971 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087  
 00,972 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103

00,973

00,974 Microsoft\_pc\_cpMIK: 'Cyrillic Bulgarian Pravetz 16 for MS-DOS

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,975 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,976 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,977 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,978 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
00,979 Data 9492,9524,9516,9500,9472,9532,9571,9553,9562,9566,9577,9574,9568,9552,9580,9488
00,980 Data 9617,9618,9619,9474,9508,8470,167,9559,9565,9496,9484,9608,9604,9612,9616,9600
00,981 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,982 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,983
00,984 Microsoft_pc_cpGESCH: 'Schpitz/Gesch (a.k.a. Georgievica a.k.a. Geschovica) is Sanmayce's layout, combining the MIK and 437, in this way: (NOT RECODING only 009,010,013):
00,985 ' shpitz
00,986
00,987 ' DEFINITIONS
00,988 ' top or extreme part
00,989 ' the ultimate, the best of, "cool"
00,990 ' typical, classic, a perfect example of; definitively
00,991
00,992 ' LANGUAGES OF ORIGIN
00,993 ' Yiddish
00,994
00,995 ' ETYMOLOGY
00,996 ' '????? shpits 'tip, peak'
00,997
00,998 ' ALTERNATIVE SPELLINGS
00,999 ' schpitz, shpits, shpitz, shpitsy
01,000
01,001 'NOTES
01,002 'Steinmetz, "Yiddish and English" has an entry for shpits, but the meaning is simply 'tip.'
01,003
01,004 ' ASCII 000..031 have to accomodate German and French '8H203e = 8254 is upperscore '&H00DC=220
01,005 Data 196,228,214,246,&H00DC,&H00FC,223,&H00c1,&H00e1,&h00C7,&H00E7
01,006 'A:a:O:o:U:u:ss A' a' C'c'
01,007 Data &H0c9,&H00e9,&H0c0,&H00e0,&H00c8,&H00e8,&H00cc,&H00ec,&H00d2,&H00f2,&H00d9,&H00f9
01,008 ' E'e' A'a' E'e' I'i' O'o' U'u'
01,009
01,010 'DATA &H00cb,&H00eb,&H00cf,&H00ef
01,011 ' E:e: I:i: ' E:e: are present in Russian, so remove them
01,012
01,013 'First half of big Cyrillic letters:
01,014 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
01,015 'Second half of big Cyrillic letters:
01,016 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
01,017 'First half of small Cyrillic letters:
01,018 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
01,019 '1st third of CP437 drawing symbols:
01,020 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
01,021 '2nd third of CP437 drawing symbols:
01,022 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
01,023 '3rd third of CP437 drawing symbols:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
01,024 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
01,025 'Second half of small Cyrillic letters:
01,026 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
01,027 'Last 16 of CP437 symbols:
01,028 '                                u-kr E:  e:  << >> Up_ ...
01,029 'DATA 8216,8217,8218,8219,8220,8221,8222,8223,176,1118,1025,1105,171,187,175,8230
01,030 '                                MICRO E:  e:  << >> Up_ ...
01,031 Data 8216,8217,8218,8219,8220,8221,8222,8223,176,8h00b5,1025,1105,171,187,175,8230
01,032
01,033 Data &H00c6,&H00E6,&H0152,&H0153
01,034 ' AE ae OEoe
01,035
01,036 Data &h2248
01,037 'almost equal to
01,038 'DATA &H221a
01,039 'square root
01,040
01,041 Data &H0102,&H0103,&H0114,&H0115,&H014e,&H014f,&H040e,&H045e,&H00cf,&H00ef
01,042 'Cyrillic short vowels: AaEeOoYy I:i:
01,043 Data &H00c2,&H00e2,&H00ca,&H00ea,&H00ce,&H00ee,&H00d4,&H00f4,&H00db,&H00fb
01,044 ' A^a^ E^e^ I^i^ O^o^ U^u^
01,045
01,046 Data &h2017,&H2320,&H2321,&h00a1,&h00bf,&h00d1,&h00f1,&h00dd,&h00fd
01,047 'doubleunderline IntegralH IntegralL, Spanish: r! r? N~ n~ Y' y'
01,048
01,049 Data &h00cd,&h00ed,&h00da,&h00fa
01,050 ' I'i'U'u'
01,051
01,052 Data &h00D3,&h00F3
01,053 ' O' o'
01,054
01,055 ' Portuguese makes use of five diacritics: the cedilla (c,), acute accent (a' e' i' o' u'), circumflex accent (a^, e^, o^), tilde (a~, o~), and grave accent (a`, and rarely e', i', o', and u').
01,056
01,057 SkipFontReviews:
01,058 handle& = _NewImage(XdimCOL + IndigoField, YdimROW + 1, 0)
01,059 Screen handle&
01,060 _Dest handle&
01,061
01,062 ' _DEST 0 refers to the present program SCREEN. You can use 0 to refer to the present program
01,063 ' SCREEN.
01,064
01,065 ' _FONT 16 'wish we could use the old 8x16 bitmap/raster fonts from DOS times...
01,066
01,067 'Let's think for people using 4K, then 32px is to be used. Now, 128col x 8 = 1024 (fits well in 1680x) or 198col x 8 = 1584 (fits well in FHD)
01,068 'IF _FILEEXISTS("C:\windows\fonts\MxPlus_ToshibaTxL2_8x16.ttf") = 0 THEN
01,069 $If BIGORSMALL = 0 Then
01,070     IF Mode4K = 0 THEN
01,071         ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
01,072         ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 16, "MONOSPACE")
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,073 ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
01,074 ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 16, "MONOSPACE")
01,075 'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
01,076 _FONT _LOADFONT(PSPlike$ + "Px437_DOS-V_re_JPN12.ttf", 12, "MONOSPACE") '2021-Apr-23, use it on old laptops with 1680- by x, it is 6x12
01,077 ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 12, "MONOSPACE") '2021-Apr-23, use it on old laptops with 1680- by x, it is 6x12
01,078 ELSE
01,079 ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 32, "MONOSPACE")
01,080 ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 32, "MONOSPACE")
01,081 'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
01,082 END IF
01,083 $Else
01,084 If _FileExists(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf") = 0 Then
01,085     If Mode4K = 0 Then
01,086         ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
01,087         ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 16, "MONOSPACE")
01,088         ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
01,089         ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 16, "MONOSPACE")
01,090         'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
01,091         _Font _LoadFont(PSPlike$ + "Px437_DOS-V_re_JPN12.ttf", 12, "MONOSPACE") '2021-Apr-23, use it on old laptops with 1680- by x, it is 6x12
01,092         ' _FONT _LOADFONT(PSPlike$ + "lucon.ttf", 12, "MONOSPACE") '2021-Apr-23, use it on old laptops with 1680- by x, it is 6x12
01,093     Else
01,094         ' _FONT _LOADFONT(PSPlike$ + "cour.ttf", 32, "MONOSPACE")
01,095         _Font _LoadFont(PSPlike$ + "lucon.ttf", 32, "MONOSPACE")
01,096         'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
01,097     End If
01,098 Else
01,099     If Mode4K = 0 Then
01,100         If FONT4x = 1 Then
01,101             _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE") 'Toshiba rules...
01,102         Else
01,103             _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 16, "MONOSPACE") 'Toshiba rules...
01,104         End If
01,105         ' _FONT _LOADFONT(PSPlike$ + "MxPlus_Cordata_PPC-400.ttf", 16, "MONOSPACE") 'Beautiful, yet the '2' has a missing dot - up-right
01,106     Else
01,107         _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
01,108         ' _FONT _LOADFONT(PSPlike$ + "MxPlus_Cordata_PPC-400.ttf", 32, "MONOSPACE")
01,109     End If
01,110 End If
01,111 $End If
01,112
01,113 If _FileExists(PSPlike$ + "Masakari.RGB") = 0 Then
01,114     GoSub SetColorScheme
01,115 Else
01,116     f00 = FreeFile
01,117     Open "Masakari.RGB" For Binary As #f00
01,118     RGB3x16$ = Space$(3 * 16)
01,119     Get #f00, , RGB3x16$
01,120     Close #f00
01,121     For ii = 0 To 15

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,122      _PaletteColor ii, _RGB32(Asc(Mid$(RGB3x16$, 3 * ii + 1, 1)) * 4, Asc(Mid$(RGB3x16$, 3 * ii + 2, 1)) * 4, Asc(Mid$(RGB3x16$, 3 * ii + 3, 1)) * 4)
01,123      Next
01,124 End If
01,125
01,126 'RESTORE Microsoft_windows_cp1252
01,127 Restore Microsoft_pc_cp6503 'My codepage a.k.a. Gesch
01,128 'RESTORE Microsoft_windows_cp1251
01,129
01,130 For ASCIICode = 0 To 6 'German A:a:O:o:U:u: ss
01,131     Read unicode
01,132     _MapUnicode unicode To ASCIICode
01,133 Next
01,134 For ASCIICode = 7 To 7 'Integral high
01,135     Read unicode
01,136     _MapUnicode unicode To ASCIICode
01,137 Next
01,138 For ASCIICode = 8 To 8 'Integral low
01,139     Read unicode
01,140     _MapUnicode unicode To ASCIICode
01,141 Next
01,142 For ASCIICode = 14 To 15 'French C,c,
01,143     Read unicode
01,144     _MapUnicode unicode To ASCIICode
01,145 Next
01,146 For ASCIICode = 16 To 27 'French/Italian E'e' A'a'E'e'I'i'O'o'U'u'
01,147     Read unicode
01,148     _MapUnicode unicode To ASCIICode
01,149 Next
01,150 'why 28..31 are not available?!
01,151 For ASCIICode = 128 To 255
01,152     Read unicode
01,153     _MapUnicode unicode To ASCIICode
01,154 Next
01,155 For ASCIICode = 220 To 223 ' the unnecessary 4 bold drawing chars are replaced with French AE ae I: i:
01,156     Read unicode
01,157     _MapUnicode unicode To ASCIICode
01,158 Next
01,159 For ASCIICode = 127 To 127 'almost equal to
01,160     Read unicode
01,161     _MapUnicode unicode To ASCIICode
01,162 Next
01,163 For ASCIICode = 181 To 190 'Cyrillic short vowels
01,164     Read unicode
01,165     _MapUnicode unicode To ASCIICode
01,166 Next
01,167 For ASCIICode = 198 To 207 'Cyrillic short vowels
01,168     Read unicode
01,169     _MapUnicode unicode To ASCIICode
01,170 Next

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,171 For ASCIICode = 208 To 216 'misc, Spanish
01,172     Read unicode
01,173     _MapUnicode unicode To ASCIICode
01,174 Next
01,175 For ASCIICode = 28 To 31 'draw animated block
01,176     Read unicode
01,177     _MapUnicode unicode To ASCIICode
01,178 Next
01,179 For ASCIICode = 11 To 12 'draw animated block
01,180     Read unicode
01,181     _MapUnicode unicode To ASCIICode
01,182 Next
01,183
01,184 ' Either one must be uncommented:
01,185 _Display
01,186 '_AUTODISPLAY 'no need of refreshing
01,187
01,188 PurpleFlag = 0
01,189
01,190 NormalFRGr = 3
01,191 If PurpleFlag Then NormalFRGr = 9
01,192 NormalBCKGr = 0
01,193 If PurpleFlag Then NormalBCKGr = 0
01,194
01,195 InverseFRGr = 0
01,196 If PurpleFlag Then InverseFRGr = 8
01,197 InverseBCKGr = 3
01,198 If PurpleFlag Then InverseBCKGr = 0
01,199
01,200 Color NormalFRGr, NormalBCKGr
01,201
01,202 ShowASCIIart
01,203
01,204 $If WINDOWS Then
01,205     i = 0
01,206     Do
01,207         i = i + 1
01,208         setting$ = Environ$(i) ' get a setting from the list
01,209         If InStr(setting$, "NUMBER_OF_PROCESSORS") Then NUMBER_OF_PROCESSORS$ = setting$
01,210     Loop Until setting$ = ""
01,211     Print NUMBER_OF_PROCESSORS$
01,212
01,213     ' For some reason the Drag-and-Drop stops working when running as Administrator?! [
01,214     'C$ = ENVIRON$("ALLUSERSPROFILE") 'try desktop for all users
01,215     'SHORTCUT$ = C$ + "\Desktop\" + "Masakari.log.txt" 'create filename for the desktop
01,216     ' For some reason the Drag-and-Drop stops working when running as Administrator?! ]
01,217     SHORTCUT$ = _StartDir$ + "\" + "Masakari.log.txt"
01,218     Open SHORTCUT$ For Append As #13
01,219     Print #13,

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,220 Print #13, "[Written on DATE: " + Date$ + ", TIME: " + Time$ + "]."
```

```

01,221 Print #13,
```

```

01,222 'CLOSE #13
```

```

01,223 Print "The log/output file, opened in APPEND mode: "; Chr$(34); SHORTCUT$; Chr$(34)
```

```

01,224 $End If
```

```

01,225
```

```

01,226 'f = FREEFILE
```

```

01,227 'SHELL _HIDE "cd > PRGMDIR.INF" 'get the current program path
```

```

01,228 'OPEN "PRGMDIR.INF" FOR INPUT AS #f
```

```

01,229 'LINE INPUT #f, current_program_path$
```

```

01,230 'CLOSE #f
```

```

01,231 'KILL "PRGMDIR.INF"
```

```

01,232 'PRINT "Current working directory path (obtained via SHELL _HIDE): "; Chr$(34); current_program_path$; Chr$(34)
```

```

01,233
```

```

01,234 'Ugh, they are reversed?!
```

```

01,235 'PRINT "Current working directory path: "; Chr$(34); _CWD$; Chr$(34)
```

```

01,236 'PRINT "User's program calling path: "; Chr$(34); _STARTDIR$; Chr$(34)
```

```

01,237 Print "Current working directory path: "; Chr$(34); _StartDir$; Chr$(34)
```

```

01,238 Print "User's program calling path: "; Chr$(34); _CWD$; Chr$(34)
```

```

01,239 'PRINT "Command line parameters sent when a program is started: "; Chr$(34); COMMAND$; Chr$(34)
```

```

01,240 'PRINT "_OS$="; _OS$
```

```

01,241
```

```

01,242 'Lazy me, in order to avoid editing the existing r.3+ code FileArray$() became function, array no more!
```

```

01,243 Dim FileArrayWINDOW$(YdimROW)
```

```

01,244
```

```

01,245 $If WINDOWS Then
```

```

01,246     AcceptFileDrop 'enables drag/drop functionality
```

```

01,247     If Command$ = "" Then Print: Print "Drag files from a folder and drop them in this window... May press Alt+X or Alt+Q to eXit/Quit..."
```

```

01,248     If Command$ = "" Then Print "May press Alt+Enter to toggle Fullscreen/Windowed modes. "
```

```

01,249     If Command$ = "" Then Print "In order to have clean/undistorted font shape, _DesktopWidth\ _DesktopHeight = " + LTrim$(Str$( _DesktopWidth)) + "/" + LTrim$(Str$( _DesktopHeight)) + " = " +
```

```

Left$(LTrim$(Str$( _DesktopWidth / _DesktopHeight)), 4)
```

```

01,250     If Command$ = "" Then Print "should be as near as possible (or multiple of) to XdimCOL\YdimROW = " + LTrim$(Str$(XdimCOL)) + "/" + LTrim$(Str$(YdimROW)) + " = " + Left$(LTrim$(Str$(XdimCOL / YdimROW)), 4)
```

```

01,251 $End If
```

```

01,252
```

```

01,253 _Display
```

```

01,254
```

```

01,255 a$ = ""
```

```

01,256 PostfixedToHeader = CsrLin
```

```

01,257 Do
```

```

01,258
```

```

01,259     Locate PostfixedToHeader, 1
```

```

01,260
```

```

01,261     'FOR d = 1 TO _DEVICES 'number of input devices found
```

```

01,262     '     dev$ = _DEVICE$(d)
```

```

01,263     '     IF INSTR(dev$, "[MOUSE]") THEN buttons = _LASTBUTTON(d): wheels = _LASTWHEEL(d): EXIT FOR
```

```

01,264     'NEXT
```

```

01,265     'PRINT
```

```

01,266     'PRINT "Mouse Buttons: "; buttons
```

```

01,267     'PRINT "Mouse Wheels: "; wheels
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,268 'd = _DEVICES ' always read number of devices to enable device input
01,269 'DO WHILE _DEVICEINPUT(2) 'loop only runs during a device 2 mouse event
01,270 'LOOP
01,271
01,272 $If WINDOWS Then
01,273 ' MEMUSAGE by Steve [
01,274 ' GetCPULoad = 0 is idle, 1 is fully used.
01,275 ' Multiply by 100 for a percentage
01,276 Print
01,277 Print "CPU used:          "; LTrim$(Str$(Int((GetCPULoad * 10000) / 100))); "%"; Space$(18)
01,278 Print "Memory used:         "; LTrim$(Str$(MemInUsePercent)); "%"; Space$(18)
01,279 Print "Total Physical Memory: "; Print AddCommas$(TotalPhysicalMem); " bytes"; Space$(18)
01,280 Print "Free Physical Memory: "; Print AddCommas$(FreePhysicalMem); " bytes"; Space$(18)
01,281 Print "Total Paging File:     "; Print AddCommas$(TotalPagingFile); " bytes"; Space$(18)
01,282 Print "Free Paging File:      "; Print AddCommas$(FreePagingFile); " bytes"; Space$(18)
01,283 ' PRINT "Total Virtual Memory: "; PRINT AddCommas$(TotalVirtualMem); " bytes"; SPACE$(18)
01,284 ' PRINT "Free Virtual Memory:  "; PRINT AddCommas$(FreeVirtualMem); " bytes"; SPACE$(18)
01,285 ' MEMUSAGE by Steve ]
01,286 $End If
01,287
01,288 'IF COMMAND$ = "" THEN PRINT: PRINT "May press Alt+X or Alt+Q to eXit/Quit..."
01,289 ReturnCOMBO
01,290
01,291 If YdimROW = 60 Then
01,292 cryFrame = CsrLin
01,293 NextFrame cryFrame + 1
01,294 Locate cryFrame, 1
01,295 End If
01,296
01,297 _Display
01,298
01,299 $If WINDOWS Then
01,300 If _TotalDroppedFiles Then
01,301 'FOR i = 1 TO _TOTALDROPPEDFILES
01,302 'a$ = _DROPPEDFILE(i)
01,303 a$ = _DroppedFile(1)
01,304 'NEXT
01,305 _FinishDrop 'If _FINISHDROP isn't called here then _TOTALDROPPEDFILES never gets reset.
01,306 'ELSE
01,307 'a$ = "Scroller.$$$"
01,308 'SHELL _HIDE "DIR /B *.* > Scroller.$$$"
01,309 End If
01,310 $End If
01,311 $If WINDOWS Then
01,312 If Command$ <> "" Then a$ = _StartDir$ + "\" + Command$
01,313 $Else
01,314 If Command$ <> "" Then a$ = _StartDir$ + "/" + Command$
01,315 $End If
01,316

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,317 'WrapFlag = 0
01,318 'Wwidth% = XdimCOL
01,319 'ul& = 0
01,320 'LongestLine = 0
01,321 'DO WHILE NOT EOF(1)
01,322 '   LINE INPUT #1, l$: ul& = ul& + 1
01,323 '   ExpandTabs l$
01,324 '   IF LEN(l$) > LongestLine THEN LongestLine = LEN(l$)
01,325 '   IF WrapFlag THEN
01,326 '       lX$ = l$
01,327 '       DO WHILE LEN(lX$) > Wwidth%
01,328 '           Glupak% = Wwidth%
01,329 '           DO UNTIL MID$(lX$, Glupak% + 1, 1) = " " AND MID$(lX$, Glupak%, 1) <> " "
01,330 '               Glupak% = Glupak% - 1
01,331 '               IF Glupak% = 0 THEN
01,332 '                   PRINT "Rejecting line(#"; LTRIM$(STR$(ul&)); ", "; LTRIM$(STR$(LEN(l$))); "chars) that cannot be wrapped!"
01,333 '                   _DISPLAY: SYSTEM
01,334 '                   GOTO B4Txpanar
01,335 '               END IF
01,336 '           LOOP
01,337 '           PRINT #3, LEFT$(lX$, Glupak%)
01,338 '           lX$ = STRING$(1, " ") + LTRIM$(MID$(lX$, Glupak% + 1, LEN(lX$) - (Glupak%)))
01,339 '       LOOP
01,340 '       PRINT #3, lX$
01,341 '       B4Txpanar:
01,342 '   END IF
01,343 'LOOP
01,344
01,345 SecondTime = 1
01,346 GettingStarted:
01,347
01,348 If WrapFlag = 1 Then
01,349     'Creating the wrapped temporary file [
01,350     If _FileExists(a$ + ".wrapped") = 0 Then
01,351         If _FileExists(a$) Then
01,352             If SecondTime <> 2 Then Print: Print "Writing wrapped variant..."
01,353             PostfixedToHeader = CsrLin
01,354
01,355             StatuLine$ = Space$(XdimCOL)
01,356             Mid$(StatuLine$, 1, 1) = "["
01,357             Mid$(StatuLine$, XdimCOL, 1) = "]"
01,358
01,359             _Display
01,360             Open a$ For Binary As #1
01,361             Open a$ + ".wrapped" For Output As #3
01,362             Open a$ + ".unwrappable" For Output As #2
01,363             LongestLine = 0
01,364             UnwrappableLines = 0
01,365             FileSize = LOF(1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,366      If FileSize Then
01,367          ChunkBeingOneUnit = FileSize \ (XdimCOL - 1 - 1) + 1: LastPercentage = 0
01,368          NumberOfLFs = 0
01,369          Wwidth% = XdimCOL
01,370          ul& = 0
01,371          ReadBytes = 0
01,372          Do While Not EOF(1)
01,373              Line Input #1, l$: ul& = ul& + 1: ReadBytes = ReadBytes + Len(l$)
01,374              ExpandTabsFULL l$
01,375              If Len(l$) > LongestLine Then LongestLine = Len(l$)
01,376              If WrapFlag Then
01,377                  ' Firstly do the wrapping virtually (in order to avoid writing some wrapped chunks and encounter unwrappable chunk) - we need either a wrapped line (in its entirety) or none:
01,378                  AssumeLineIsWrappable = 1
01,379                  lX$ = l$
01,380                  Do While Len(lX$) > Wwidth%
01,381                      Glupak% = Wwidth%
01,382                      Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak% + 1, 1) = ";" OR
MID$(lX$, Glupak% + 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "
01,383                          InvokeOnce$ = Mid$(lX$, Glupak% + 1, 1)
01,384                          If InStr(" \_/.-!+%=:", InvokeOnce$) And Mid$(lX$, Glupak%, 1) <> " " Then Exit Do
01,385                          'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = ";" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR
InvokeOnce$ = "!" OR InvokeOnce$ = "+" OR InvokeOnce$ = "%") AND MID$(lX$, Glupak%, 1) <> " " THEN EXIT DO
01,386                          Glupak% = Glupak% + 1
01,387                          If Glupak% = 0 Then
01,388                              AssumeLineIsWrappable = 0
01,389                              GoTo B4TxpanarVIRTUAL
01,390                          End If
01,391                      Loop
01,392                      lX$ = LTrim$(Mid$(lX$, Glupak% + 1, Len(lX$) - (Glupak%)))
01,393                      Loop
01,394                      B4TxpanarVIRTUAL:
01,395                      If AssumeLineIsWrappable = 1 Then
01,396                          lX$ = l$
01,397                          Do While Len(lX$) > Wwidth%
01,398                              Glupak% = Wwidth%
01,399                              Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak% + 1, 1) = ";" OR
OR MID$(lX$, Glupak% + 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "
01,400                                  InvokeOnce$ = Mid$(lX$, Glupak% + 1, 1)
01,401                                  If InStr(" \_/.-!+%=:", InvokeOnce$) And Mid$(lX$, Glupak%, 1) <> " " Then Exit Do
01,402                                  'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = ";" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR
InvokeOnce$ = "!" OR InvokeOnce$ = "+" OR InvokeOnce$ = "%") AND MID$(lX$, Glupak%, 1) <> " " THEN EXIT DO
01,403                                  Glupak% = Glupak% + 1
01,404                                  If Glupak% = 0 Then
01,405                                      Print "Rejecting line(#); LTrim$(Str$(ul&)); ", "; LTrim$(Str$(Len(l$))); "chars) that cannot be wrapped!"
01,406                                      Close #1, #3: Kill a$ + ".wrapped"
01,407                                      _Display: End
01,408                                      GoTo B4Txpanar
01,409                                  End If
01,410                              Loop

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,411      Print #3, Left$(IX$, Glupak%)
01,412      'IX$ = STRING$(1, " ") + LTRIM$(MID$(IX$, Glupak% + 1, LEN(IX$) - (Glupak%)))
01,413      IX$ = LTrim$(Mid$(IX$, Glupak% + 1, Len(IX$) - (Glupak%)))
01,414      Loop
01,415      Print #3, IX$
01,416      B4Txpanar:
01,417      Else
01,418          Print #2, l$: UnwrappableLines = UnwrappableLines + 1
01,419      End If
01,420  End If
01,421  If (ReadBytes \ ChunkBeingOneUnit) > LastPercentage Then
01,422      LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
01,423      Mid$(StatuLine$, 2, LastPercentage) = String$(LastPercentage, Chr$(176))
01,424      Locate YdimROW + 1, 1: Color 9, 0: Print StatuLine$;
01,425      _Display
01,426  End If
01,427  Loop
01,428  Else
01,429      Close #1 'when filesize is 0
01,430      System
01,431  End If 'IF FileSize THEN
01,432  Close #1, #3
01,433  a$ = a$ + ".wrapped"
01,434  Locate PostfixedToHeader, 1
01,435  Color NormalFRGr, NormalBCKGr
01,436  End If
01,437  Else
01,438      a$ = a$ + ".wrapped"
01,439  End If
01,440  'Creating the wrapped temporary file ]
01,441  End If
01,442
01,443  TimeA = Timer
01,444
01,445  ' Below fragment is r.4+ with 64bit pointers... to external/internal RAM!
01,446  ' On i5-7200u, SSD Kingston 256GB it loads OED in ???/??? seconds, respectively.
01,447  If _FileExists(a$) Then
01,448      If SecondTime <> 2 Then Print: Print "Loading...": Print
01,449
01,450      PostfixedToHeader = CsrLin
01,451
01,452      StatuLine$ = Space$(XdimCOL)
01,453      Mid$(StatuLine$, 1, 1) = "["
01,454      Mid$(StatuLine$, XdimCOL, 1) = "]"
01,455
01,456      _Display
01,457      Open a$ For Binary As #1
01,458      LongestLine = 0
01,459      FileSize = LOF(1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,460     If FileSize Then
01,461         ChunkBeingOneUnit = FileSize \ (XdimCOL - 1 - 1) + 1: LastPercentage = 0
01,462         NumberOfLFs = 0
01,463         If ToLoadOrNotFlag Then
01,464             Mwholefile = _MemNew(FileSize + 1)
01,465             'TheWholeFile$ = SPACE$(FileSize + 1) ' Grmbl, this comes with limitation probably 2GB, which prompts for buffered e.g. 128KB approach as my old LineWordReporter.c tool.
01,466             'GET #1, , TheWholeFile$
01,467             ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
01,468             chunk128KB$ = SPACE$(128 * 1024 * 1024)
01,469             Do While ReadBytes + (128 * 1024 * 1024) < FileSize
01,470                 Get #1, , chunk128KB$
01,471                 i = 1
01,472                 Do While i <= (128 * 1024 * 1024)
01,473                     i = Instr(i, chunk128KB$, Chr$(10))
01,474                     If i = 0 Then Exit Do
01,475                     NumberOfLFs = NumberOfLFs + 1
01,476                     i = i + 1
01,477                 Loop
01,478                 'FOR i = 1 TO LEN(chunk128KB$)
01,479                 ' IF MID$(chunk128KB$, i, 1) = CHR$(10) THEN NumberOfLFs = NumberOfLFs + 1 ' Boost it with INSTR...
01,480                 'NEXT i
01,481                 _MemPut Mwholefile, Mwholefile.OFFSET + ReadBytes, chunk128KB$
01,482                 ReadBytes = ReadBytes + (128 * 1024 * 1024)
01,483             Loop
01,484             If (FileSize - ReadBytes) Then
01,485                 RemainingChunk$ = SPACE$(FileSize - ReadBytes)
01,486                 Get #1, , RemainingChunk$
01,487                 i = 1
01,488                 Do While i <= Len(RemainingChunk$)
01,489                     i = Instr(i, RemainingChunk$, Chr$(10))
01,490                     If i = 0 Then Exit Do
01,491                     NumberOfLFs = NumberOfLFs + 1
01,492                     i = i + 1
01,493                 Loop
01,494                 'FOR i = 1 TO LEN(RemainingChunk$)
01,495                 ' IF MID$(RemainingChunk$, i, 1) = CHR$(10) THEN NumberOfLFs = NumberOfLFs + 1 ' Boost it with INSTR...
01,496                 'NEXT i
01,497                 _MemPut Mwholefile, Mwholefile.OFFSET + ReadBytes, RemainingChunk$
01,498             End If
01,499         End If
01,500         'Bugfix from 2021-Jan-29, inhere problem exists, in efficient mode we don't know 'NumberOfLFs' therefore should read-it-without-loading-it:
01,501         If ToLoadOrNotFlag = 0 Then
01,502             ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
01,503             chunk128KB$ = SPACE$(128 * 1024 * 1024)
01,504             Do While ReadBytes + (128 * 1024 * 1024) < FileSize
01,505                 Get #1, , chunk128KB$
01,506                 i = 1
01,507                 Do While i <= (128 * 1024 * 1024)
01,508                     i = Instr(i, chunk128KB$, Chr$(10))

```



```

01,509         If i = 0 Then Exit Do
01,510         NumberOfLFs = NumberOfLFs + 1
01,511         i = i + 1
01,512     Loop
01,513     ReadBytes = ReadBytes + (128 * 1024 * 1024)
01,514 Loop
01,515 If (FileSize - ReadBytes) Then
01,516     RemainingChunk$ = Space$(FileSize - ReadBytes)
01,517     Get #1, , RemainingChunk$
01,518     i = 1
01,519     Do While i <= Len(RemainingChunk$)
01,520         i = Instr(i, RemainingChunk$, Chr$(10))
01,521         If i = 0 Then Exit Do
01,522         NumberOfLFs = NumberOfLFs + 1
01,523         i = i + 1
01,524     Loop
01,525 End If
01,526 End If
01,527 Seek #1, 1
01,528 'IF ASC(Byte) < 32 AND Byte <> CHR$(10) AND Byte <> CHR$(13) AND Byte <> CHR$(9) THEN Byte = CHR$(32)
01,529 filecount = 0
01,530 'Ugh, buggy, in r.5 below two "malloc" lines should be using 'NumberOfLFs' not 'FileSize'...
01,531 MhandleOFF = _MemNew(888 * (NumberOfLFs + 1)) 'create new memory block of 8*NumberOfLFs bytes - each line has its own 64bit Offset
01,532 MhandleLEN = _MemNew(888 * (NumberOfLFs + 1)) 'create new memory block of 8*NumberOfLFs bytes - each line has its own 64bit Length, kinda overkill, could be 32bit
01,533
01,534 ' Parser for r.6 [[[
01,535 '     LineLen13 = 0
01,536 '     Byte = ""
01,537 '     'Caution: Many files DON'T end with CRLF, or LF or CR - kinda the last line is not always postixed correctly!
01,538 '     'This means if after the last CRLF, or LF or CR there is at least one byte then the number of total lines should be +1
01,539 '     FOR j = 1 TO FileSize + 1
01,540 '
01,541 '         IF (j \ ChunkBeingOneUnit) > LastPercentage THEN
01,542 '             LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
01,543 '             MID$(StatuLine$, 2, LastPercentage) = STRING$(LastPercentage, CHR$(176))
01,544 '             LOCATE YdimROW + 1, 1: COLOR 9, 0: PRINT StatuLine$;
01,545 '             _DISPLAY
01,546 '         END IF
01,547 '
01,548 '         PrevByte = Byte
01,549 '         IF j <= FileSize THEN
01,550 '             IF ToLoadOrNotFlag THEN
01,551 '                 'Byte = MID$(TheWholeFile$, j, 1)
01,552 '                 _MEMGET Mwholefile, Mwholefile.OFFSET + (j - 1), Byte
01,553 '             ELSE
01,554 '                 GET #1, , Byte
01,555 '             END IF
01,556 '         ELSE
01,557 '             Byte = ""

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,558 '           IF PrevByte <> CHR$(10) THEN Byte = CHR$(10) ' enforce last line to be postfixed ONLY if last char was not LF
01,559 '           END IF
01,560 '           QWORD = j
01,561 '           IF Byte = CHR$(10) THEN 'For now, won't work for MacOS...
01,562 '               QWORD = QWORD - LineLen13 'have to write the current offset minus the length of the line
01,563 '               MEMPUT MhandleOFF, MhandleOFF.OFFSET + 8&& * filecount, QWORD
01,564 '               IF PrevByte = CHR$(13) THEN LineLen13 = LineLen13 - 1
01,565 '               MEMPUT MhandleLEN, MhandleLEN.OFFSET + 8&& * filecount, LineLen13
01,566 '               IF LineLen13 > LongestLine THEN LongestLine = LineLen13
01,567 '               LineLen13 = 0
01,568 '               filecount = filecount + 1
01,569 '           ELSE
01,570 '               LineLen13 = LineLen13 + 1
01,571 '           END IF
01,572 '       NEXT j
01,573 ' Parser for r.6 ]]]
01,574
01,575 ' Parser for r.7 [[[[[[
01,576 If ToLoadOrNotFlag = 0 Then
01,577     ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
01,578     ChunkLen = 128 * 1024
01,579     chunk128KB$ = Space$(ChunkLen) ' +1 for sentinel
01,580     j = 1 ' is the current offset where new GET reads
01,581     QWORDlast = 1
01,582     PrevByte = ""
01,583     Do While ReadBytes + (ChunkLen) < FileSize ' this '<' is important, on purpose not using '<=' since there should be a remnant chunk (where LF postfixing is enforced, eventually)
01,584
01,585         If (j \ ChunkBeingOneUnit) > LastPercentage Then
01,586
01,587             If SecondTime <> 2 Then
01,588                 $If WINDOWS Then
01,589                     Locate PostfixedToHeader, 1
01,590                     ' MEMUSAGE by Steve [
01,591                     ' GetCPULoad = 0 is idle, 1 is fully used.
01,592                     ' Multiply by 100 for a percentage
01,593                     Print "CPU used:           "; LTrim$(Str$(Int((GetCPULoad * 10000) / 100))); "%"; Space$(18)
01,594                     Print "Memory used:          "; LTrim$(Str$(MemInUsePercent)); "%"; Space$(18)
01,595                     Print "Total Physical Memory: "; Print AddCommas$(TotalPhysicalMem); " bytes"; Space$(18)
01,596                     Print "Free Physical Memory:  "; Print AddCommas$(FreePhysicalMem); " bytes"; Space$(18)
01,597                     Print "Total Paging File:     "; Print AddCommas$(TotalPagingFile); " bytes"; Space$(18)
01,598                     Print "Free Paging File:      "; Print AddCommas$(FreePagingFile); " bytes"; Space$(18)
01,599                     Print "Total Virtual Memory:  "; Print AddCommas$(TotalVirtualMem); " bytes"; Space$(18)
01,600                     Print "Free Virtual Memory:   "; Print AddCommas$(FreeVirtualMem); " bytes"; Space$(18)
01,601                     ' MEMUSAGE by Steve ]
01,602                 $End If
01,603             End If
01,604
01,605             LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
01,606             Mid$(StatuLine$, 2, LastPercentage) = String$(LastPercentage, Chr$(176))

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,607         Locate YdimROW + 1, 1: Color 9, 0: Print StatuLine$;
01,608
01,609         If YdimROW = 60 Then
01,610             cryFrame = CsrLin
01,611             NextFramePEN 1
01,612             Locate cryFrame, 1
01,613         End If
01,614
01,615         _Display
01,616     End If
01,617
01,618     LastByte = Right$(chunk128KB$, 1) 'to handle eventual CR, left behind i.e. in previous chunk
01,619     Get #1, j, chunk128KB$
01,620     i = 1
01,621     FoundAt = InStr(i, chunk128KB$, Chr$(10))
01,622     If FoundAt Then
01,623         Do While FoundAt
01,624             If FoundAt = 1 Then PrevByte = LastByte Else PrevByte = Mid$(chunk128KB$, FoundAt - 1, 1)
01,625             QWORD = (j - 1) + FoundAt
01,626             LineLen13 = QWORD - QWORDlast
01,627             _MemPut MhandleOFF, MhandleOFF.OFFSET + 888 * filecount, QWORDlast
01,628             QWORDlast = QWORD + 1
01,629             If PrevByte = Chr$(13) Then LineLen13 = LineLen13 - 1
01,630             _MemPut MhandleLEN, MhandleLEN.OFFSET + 888 * filecount, LineLen13
01,631             If LineLen13 > LongestLine Then LongestLine = LineLen13
01,632             filecount = filecount + 1
01,633             i = FoundAt + 1
01,634             If i > (ChunkLen) Then Exit Do 'could use sentinel (buffer+1), in order this line to drop out
01,635             FoundAt = InStr(i, chunk128KB$, Chr$(10))
01,636         Loop
01,637     End If
01,638     j = j + (ChunkLen)
01,639     ReadBytes = ReadBytes + (ChunkLen)
01,640 Loop
01,641 If (FileSize - ReadBytes) Then
01,642     RemainingChunk$ = Space$(FileSize - ReadBytes) '*1 for sentinel
01,643     LastByte = Right$(chunk128KB$, 1) 'to handle eventual CR, left behind i.e. in previous chunk
01,644     Get #1, RemainingChunk$
01,645     If Right$(RemainingChunk$, 1) <> Chr$(10) Then RemainingChunk$ = RemainingChunk$ + Chr$(10) ' dirty, enforcing not missing the last line (if it is not postfixed with LF)
01,646     'Beware, yes be aware that above line should have been applied for above/first fragment because the filesize could be multiple of the chunk length i.e. no remaining chunk, however it was feinted by
    '<'
01,647     i = 1
01,648     FoundAt = InStr(i, RemainingChunk$, Chr$(10))
01,649     If FoundAt Then
01,650         Do While FoundAt
01,651             If FoundAt = 1 Then PrevByte = LastByte Else PrevByte = Mid$(RemainingChunk$, FoundAt - 1, 1)
01,652             QWORD = (j - 1) + FoundAt
01,653             LineLen13 = QWORD - QWORDlast
01,654             _MemPut MhandleOFF, MhandleOFF.OFFSET + 888 * filecount, QWORDlast

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,655         QWORDlast = QWORD + 1
01,656         If PrevByte = Chr$(13) Then LineLen13 = LineLen13 - 1
01,657         _MemPut MhandleLEN, MhandleLEN.OFFSET + 8&& * filecount, LineLen13
01,658         If LineLen13 > LongestLine Then LongestLine = LineLen13
01,659         filecount = filecount + 1
01,660         i = FoundAt + 1
01,661         If i > (FileSize - ReadBytes) Then Exit Do 'could use sentinel (buffer+1), in order this line to drop out
01,662         FoundAt = InStr(i, RemainingChunk$, Chr$(10))
01,663     Loop
01,664     End If
01,665 End If
01,666 End If
01,667 Seek #1, 1
01,668 ' Parser for r.7 ]]]]]]
01,669
01,670 Else
01,671     Close #1 'when filesize is 0
01,672     System
01,673 End If 'IF FileSize THEN
01,674 End If
01,675
01,676 ' Below fragment is r.4 with 64bit pointers... to external RAM!
01,677 ' On i5-7200u, SSD Kingston 256GB it loads OED in 3,339 seconds, naturally the second attempt is to read byte-by-byte from RAM, however the whole file has to be loaded.
01,678 'IF FILEEXISTS(a$) THEN
01,679 '    PRINT: PRINT "Loading..."
01,680 '    _DISPLAY
01,681 '    OPEN a$ FOR BINARY AS #1
01,682 '    FileSize = LOF(1)
01,683 '    filecount = 0
01,684 '    MhandleOFF = _MEMNEW(8&& * FileSize) 'create new memory block of 8*filesize bytes - each line has its own 64bit Offset
01,685 '    MhandleLEN = _MEMNEW(8&& * FileSize) 'create new memory block of 8*filesize bytes - each line has its own 64bit Length, kinda overkill, could be 32bit
01,686 '    LineLen13 = 0
01,687 '    Byte = ""
01,688 '    'Caution: Many files DON'T end with CRLF, or LF or CR - kinda the last line is not always postixed correctly!
01,689 '    'This means if after the last CRLF, or LF or CR there is at least one byte then the number of total lines should be +1
01,690 '    FOR j = 1 TO FileSize + 1
01,691 '        PrevByte = Byte
01,692 '        IF j <= FileSize THEN
01,693 '            GET #1, , Byte
01,694 '        ELSE
01,695 '            Byte = ""
01,696 '            IF PrevByte <> CHR$(10) THEN Byte = CHR$(10) ' enforce last line to be postfixed ONLY if last char was not LF
01,697 '        END IF
01,698 '        QWORD = j
01,699 '        IF Byte = CHR$(10) THEN 'For now, won't work for MacOS...
01,700 '            QWORD = QWORD - LineLen13 'have to write the current offset minus the length of the line
01,701 '            _MEMPUT MhandleOFF, MhandleOFF.OFFSET + 8&& * filecount, QWORD
01,702 '            IF PrevByte = Chr$(13) THEN LineLen13 = LineLen13 - 1
01,703 '            _MEMPUT MhandleLEN, MhandleLEN.OFFSET + 8&& * filecount, LineLen13

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,704 '          IF LineLen13 > LongestLine THEN LongestLine = LineLen13
01,705 '          LineLen13 = 0
01,706 '          filecount = filecount + 1
01,707 '      ELSE
01,708 '          LineLen13 = LineLen13 + 1
01,709 '      END IF
01,710 '  NEXT j
01,711 'END IF
01,712
01,713 ''PRINT "filecount= "; filecount
01,714 ''PRINT "LongestLine= "; LongestLine
01,715 ''exercise in reproducing the indexed file:
01,716 ''Debugo = CHR$(13) + CHR$(10)
01,717 'Debugo = CHR$(10)
01,718 'OPEN a$ + ".dump" FOR BINARY AS #2
01,719 'FOR i = 1 TO filecount
01,720 '    _MEMGET MhandleOFF, MhandleOFF.OFFSET + 8&& * (i - 1), QWORD
01,721 '    _MEMGET MhandleLEN, MhandleLEN.OFFSET + 8&& * (i - 1), LineLen13
01,722 '    PRINT QWORD, LineLen13:
01,723 '    SEEK #1, QWORD
01,724 '    BufferForLine$ = ""
01,725 '    FOR j = 1 TO LineLen13
01,726 '        GET #1, , Byte 'grmbl, how to load in one call?! As fread() is capable.
01,727 '        PUT #2, , Byte
01,728 '        BufferForLine$ = BufferForLine$ + Byte
01,729 '    NEXT j
01,730 '    PUT #2, , Debugo
01,731 '    PRINT BufferForLine$
01,732 'NEXT i
01,733 ' _DISPLAY
01,734 'END
01,735
01,736 ' Commented fragment is r.3+ with slow strings
01,737 'LongestLine = 0
01,738 'IF _FILEEXISTS(a$) THEN
01,739 '    PRINT: PRINT "Loading..."
01,740 '    _DISPLAY
01,741 '    filecount = 0
01,742 '    OPEN a$ FOR INPUT AS #1
01,743 '    OPEN a$ FOR BINARY AS #1 ' As suggested by SMCNeill and Pete in https://www.qb64.org/forum/index.php?topic=3518.msg128631#msg128631
01,744 '    DO UNTIL EOF(1)
01,745 '        LINE INPUT #1, l$
01,746 '        filecount = filecount + 1
01,747 '    LOOP
01,748 '    CLOSE #1
01,749 '    REDIM FileArray$(1 TO filecount)
01,750 '    filecount = 0
01,751 '    OPEN a$ FOR INPUT AS #1
01,752 '    OPEN a$ FOR BINARY AS #1 ' As suggested by SMCNeill and Pete in https://www.qb64.org/forum/index.php?topic=3518.msg128631#msg128631

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,753 '      FileSize = LOF(1)
01,754 '      DO UNTIL EOF(1)
01,755 '          LINE INPUT #1, l$
01,756 '          ExpandTabs l$
01,757 '          FOR j = 1 TO LEN(l$)
01,758 '              IF MID$(l$, j, 1) < CHR$(32) THEN MID$(l$, j, 1) = CHR$(32)
01,759 '          NEXT j
01,760 '          IF LEN(l$) > LongestLine THEN LongestLine = LEN(l$)
01,761 '          filecount = filecount + 1
01,762 '          FileArray$(filecount) = l$
01,763 '      LOOP
01,764 '      CLOSE #1
01,765 'END IF
01,766
01,767 Limit 5
01,768 If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(81) Or _KeyDown(113)) Then System
01,769 If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(88) Or _KeyDown(120)) Then System
01,770
01,771 '          QB64 _KEYHIT and _KEYDOWN Values
01,772 '
01,773 'Esc  F1  F2  F3  F4  F5  F6  F7  F8  F9  F10 F11 F12 Sys ScL Pause
01,774 ' 27 15104 15360 15616 15872 16128 16384 16640 16896 17152 17408 34048 34304 +316 +302 +019
01,775 ' ~ 1! 2@ 3# 4$ 5% 6^ 7& 8* 9( 0) _ += BkSp Ins Hme PUp NumL / * -
01,776 '126 33 64 35 36 37 94 38 42 40 41 95 43 8 20992 18176 18688 +300 47 42 45
01,777 ' 96 49 50 51 52 53 54 55 56 57 48 45 61
01,778 'Tab Q W E R T Y U I O P [{ ]} \| Del End PDn 7Hme 8/? 9PU +
01,779 ' 9 81 87 69 82 84 89 85 73 79 80 123 125 124 21248 20224 20736 18176 18432 18688 43
01,780 ' 113 119 101 114 116 121 117 105 111 112 91 93 92 55 56 57
01,781 'CapL A S D F G H J K L ;: '" Enter 4/?- 5 6/-?
01,782 '+301 65 83 68 70 71 72 74 75 76 58 34 13 19200 19456 19712 E
01,783 ' 97 115 100 102 103 104 106 107 108 59 39 52 53 54 n
01,784 'Shift Z X C V B N M ,< .> /? Shift ? 1End 2/? 3PD t
01,785 '+304 90 88 67 86 66 78 77 60 62 63 +303 18432 20224 20480 20736 e
01,786 ' 122 120 99 118 98 110 109 44 46 47 49 50 51 r
01,787 'Ctrl Win Alt Spacebar Alt Win Menu Ctrl ?- ? -? 0Ins .Del
01,788 '+306 +311 +308 32 +307 +312 +319 +305 19200 20480 19712 20992 21248 13
01,789 ' 48 46
01,790 '      Lower value = LCase/NumLock On _____ + = add 100000
01,791
01,792 Loop While a$ = ""
01,793
01,794 If XdimCOL = 128 And YdimROW = 40 Then _Title "MASAKARI, revision " + Mrev$ + " 128x40, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,795 If XdimCOL = 128 And YdimROW = 60 Then _Title "MASAKARI, revision " + Mrev$ + " 128x60, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,796 If XdimCOL = 213 And YdimROW = 60 Then _Title "MASAKARI, revision " + Mrev$ + " 213x60, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,797 If XdimCOL = 300 And YdimROW = 80 Then _Title "MASAKARI, revision " + Mrev$ + " 300x80, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,798 If XdimCOL = 112 And YdimROW = 30 Then _Title "MASAKARI, revision " + Mrev$ + " 112x30, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,799 LoadedFile$ = a$
01,800
01,801 TimeB = Timer: SecondTime = 2

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

Listing: MASAKARI_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus_ToshibaTxl2_8x16.ttf; Downloadable at: www.sanmayce.com/Masakari.zip
01,802 'PLAY "L8V2a-c-" 'removed in order to get rid of audio dependency
01,803 CurrentLine = 1
01,804 File_Frame_x = 1
01,805 File_Frame_y = 1
01,806 Color NormalFRGr, NormalBCKGr
01,807 UpdateWindowFrame NormalFRGr, NormalBCKGr
01,808
01,809 ' Here is the layout:
01,810 ' The window-frame is 1..crx or 1..128 | The file-frame is 1..LongestLine
01,811 '           .           .           .
01,812 '           .           .           .
01,813 '           cry        60           filecount
01,814 '           FileArrayWINDOW$(60) | FileArray(filecount)
01,815 ' if File_Frame_x < 128 then PADDING to 128 else File_Frame_x = 1..LongestLine-(128-1)
01,816 ' if File_Frame_y < 60 then PADDING to 60 else File_Frame_y = 1..filecount-(60-1)
01,817
01,818 UpdateCLine 1, 1, InverseFRGr, InverseBCKGr, 1
01,819 StatuLine$ = Space$(XdimCOL)
01,820 Mid$(StatuLine$, 1, 1) = "["
01,821 Mid$(StatuLine$, XdimCOL, 1) = "]"
01,822
01,823 FieldLineNum = Len(AddCommas$(filecount))
01,824 FieldLineLen = Len(AddCommas$(LongestLine))
01,825
01,826 'Dumbo$ = "File Size: " + AddCommas$(FileSize) + "; Longest Line: " + AddCommas$(LongestLine)
01,827 'Dumbo$ = "Filesize: " + AddCommas$(FileSize) + "; Longest: " + AddCommas$(LongestLine)
01,828 Mid$(StatuLine$, 2, Len(Dumbo$)) = Dumbo$
01,829 Locate YdimROW + 1, 1
01,830 Color 9, 0: Print StatuLine$;
01,831 UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1 ' +1 due to [
01,832 'MostRightField = LEN(Dumbo$) + 1 + 1 + LEN("; Line Number: ") + LEN("; Line Length: ") + FieldLineNum + FieldLineLen
01,833 'MostRightField = Len(Dumbo$) + 1 + 1 + Len("; Line: ") + Len("; Linesize: ") + FieldLineNum + FieldLineLen
01,834 UpdateNextToCCL_DONE MostRightField
01,835
01,836 If WrapFlag = 1 Then
01,837     UpdateNextToCCL_UNWRAPPABLEcount MostRightField
01,838 End If
01,839
01,840 CursorS = 30
01,841 CursorE = 31
01,842
01,843 _Display
01,844 Locate 1, 1, 1, CursorS, CursorE
01,845 crx = Pos(0)
01,846 cry = CsrLin
01,847 crxOLD = crx
01,848 cryOLD = cry
01,849 PageScrollBenchmark = 0
01,850 LineScrollBenchmark = 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,851
01,852 tCurrentT1# = Timer(.001)
01,853 tLASTrelease1# = tCurrentT1#
01,854 tLASTpress1# = tCurrentT1#
01,855 tCurrentT2# = Timer(.001)
01,856 tLASTrelease2# = tCurrentT2#
01,857 tLASTpress2# = tCurrentT2#
01,858
01,859 SimulateTabRel = 0
01,860 Do 'mainloop [
01,861     ReturnCOMBO
01,862     IsF12released_Flag = 0
01,863     IsF2released_Flag = 0
01,864     IsF3released_Flag = 0
01,865     IsENTERreleased_Flag = 0
01,866     IsF10released_Flag = 0
01,867     IsF11released_Flag = 0
01,868
01,869     MustBeLONGssigned& = _KeyHit
01,870     If SimulateTabRel = 1 Then SimulateTabRel = 0: MustBeLONGssigned& = -9
01,871     If MustBeLONGssigned& Then
01,872         Select Case MustBeLONGssigned&
01,873             Case -20992: 'Ins released
01,874                 LastHit~&& = EpochTime~&&: Stepping = 10
01,875                 If CursorS = 0 Then
01,876                     $If WINDOWS Then
01,877                         Play "v15148ba"
01,878                     $End If
01,879                     CursorS = 30
01,880                     CursorE = 31
01,881                 Else
01,882                     $If WINDOWS Then
01,883                         Play "v15148eef"
01,884                     $End If
01,885                     CursorS = 0
01,886                     CursorE = 31
01,887                 End If
01,888                 _Display
01,889             Case -9: 'Tab released while LShift is held
01,890                 If _KeyDown(LSHIFTkey&) Then
01,891
01,892                     LastHit~&& = EpochTime~&&: Stepping = 10
01,893                     UpdateCLineHIGHLIGHTword cry, 1, InverseFRGr, InverseBCKGr, cry
01,894                     Locate 1, 1, 0, CursorS, CursorE: _Display 'adding refresh in order to get rid of eventual cursor (caught as visible while blinking)
01,895                     Color 9, 0
01,896
01,897                     ' Scrolling into 3D djamtche [
01,898                     'SUB DrawBoxShadow3 (TopLrow, TopLcol, BottomRow, BottomRcol, Capttn$)
01,899                     dictTAB$ = "Masakari's dictionaries"

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,900      If Len(dictTAB$) > 70 Then Reduced$ = "." + Right$(dictTAB$, 70) Else Reduced$ = dictTAB$
01,901      Call DrawBoxShadowOED(3 + 3, 3, 35, 125, Reduced$, 1)
01,902      BckGRcolor = 1 '6
01,903      PhysicalLine& = 1
01,904      dictTAB_TotalWrd& = 23
01,905      $If WINDOWS Then
01,906          Play "v15148ga"
01,907      $End If

01,908
01,909      s$ = LTrim$(Str$(PhysicalLine&))
01,910      If Len(s$) > 3 Then
01,911          If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
01,912          s$ = ""
01,913          For i = 1 To Len(x$) Step 3
01,914              s$ = s$ + Mid$(x$, i, 3) + ","
01,915          Next
01,916          s$ = Left$(s$, Len(s$) - 1)
01,917      End If
01,918      s$ = LTrim$(s$)
01,919      Padded$ = "0,000,000,000"
01,920      Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
01,921      Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
01,922      Locate , , 0
01,923
01,924      DjamPoolLength = 125 - 3 + 1 - 4
01,925      DjamPoolHeight = 35 - (3 + 3) + 1 - 4
01,926      CurrentPos& = PhysicalLine& ' Notice that CurrentPos& = 1..(dictTAB_TotalWrd& - DjamPoolHeight + 1)
01,927      ' inhere it could be between 1.. due to BINARY search
01,928      If dictTAB_TotalWrd& < DjamPoolHeight Then CurrentPos2max& = 1 Else CurrentPos2max& = (dictTAB_TotalWrd& - DjamPoolHeight + 1)
01,929      If CurrentPos& > CurrentPos2max& Then CurrentPos& = CurrentPos2max&
01,930      PoolLinePos = 1
01,931      Do
01,932          key$ = InKey$
01,933          'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
01,934          If key$ <> "" Then
01,935              code% = Asc(key$):
01,936              If code% Then ' ASC returns any value greater than 0
01,937                  Select Case Asc(key$)
01,938                      Case 13:
01,939                          If CurrentPos& + (PoolLinePos - 1) = 13 Then
01,940                              Call dict02sub
01,941                              GoTo DirtyGo
01,942                          End If
01,943                      Case 32:
01,944                      Case 65 TO 97: 'PRINT key$;
01,945                      Case Asc("a") TO Asc("z"): 'PRINT key$;
01,946                      'CASE 27: COLOR 7, 0: SYSTEM 'END
01,947                  End Select
01,948              Else

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,949         Select Case Asc(key$, 2)
01,950             Case 72:
01,951                 If PoolLinePos > 1 Then
01,952                     PoolLinePos = PoolLinePos - 1
01,953             Else
01,954                 If CurrentPos > 1 Then CurrentPos = CurrentPos - 1
01,955             End If
01,956         Case 80:
01,957             If PoolLinePos < MIN$(DjamPoolHeight, dictTAB_TotalWrd) Then
01,958                 PoolLinePos = PoolLinePos + 1
01,959             Else
01,960                 If CurrentPos < CurrentPos2max Then CurrentPos = CurrentPos + 1
01,961             End If
01,962         Case 75: 'Left
01,963         Case 77: 'Right
01,964         Case 71: 'Home
01,965             PoolLinePos = 1
01,966             CurrentPos = 1
01,967         Case 79: 'End
01,968             PoolLinePos = MIN$(DjamPoolHeight, dictTAB_TotalWrd)
01,969             CurrentPos = CurrentPos2max
01,970         Case 73: 'PgUp
01,971             If CurrentPos - DjamPoolHeight >= 1 Then CurrentPos = CurrentPos - DjamPoolHeight
01,972         Case 81: 'PgDn
01,973             If CurrentPos + DjamPoolHeight <= CurrentPos2max Then CurrentPos = CurrentPos + DjamPoolHeight
01,974         Case 61: 'F3
01,975         Case 13: 'Enter
01,976
01,977             End Select
01,978         End If
01,979     End If
01,980
01,981     For i = 1 To MIN$(DjamPoolHeight, dictTAB_TotalWrd) ' with dictionaries all are bigger than ~40 lines
01,982         PADDED_EntryAll$ = DCTNS$(CurrentPos + (i - 1)) + String$(DjamPoolLength - Len(DCTNS$(CurrentPos + (i - 1))), " ")
01,983         Locate 3 + 3 + 2 + (i - 1), 3 + 2
01,984         If i = PoolLinePos Then Color BckGBcolor, 0 Else Color 0, BckGBcolor
01,985         Print PADDED_EntryAll$;
01,986     Next
01,987     s$ = LTrim$(Str$(CurrentPos + PoolLinePos - 1))
01,988     If Len(s$) > 3 Then
01,989         If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
01,990         s$ = ""
01,991         For i = 1 To Len(x$) Step 3
01,992             s$ = s$ + Mid$(x$, i, 3) + ", "
01,993         Next
01,994         s$ = Left$(s$, Len(s$) - 1)
01,995     End If
01,996     s$ = LTrim$(s$)
01,997     Padded$ = "0,000,000,000"

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,998      Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
01,999      Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
02,000      _Display 'in order to not having frozen screen
02,001      _Limit 60
02,002      Loop Until _KeyDown(27)
02,003      ' Scrolling into 3D djamtche ]
02,004      DirtyGo:
02,005      _KeyClear 2
02,006      UpdateWindowFrame NormalFGR, NormalBCKGr
02,007      _Display
02,008      End If
02,009      Case Else
02,010      End Select
02,011      If MustBeLONGsigned < 0 Then 'negative value means key released
02,012      MustBeLONGsigned = -MustBeLONGsigned
02,013      If MustBeLONGsigned \ 256 = 60 Then
02,014      IsF2released_Flag = 1
02,015      Else
02,016      End If
02,017      If MustBeLONGsigned \ 256 = 61 Then
02,018      IsF3released_Flag = 1
02,019      Else
02,020      End If
02,021      If MustBeLONGsigned \ 256 = 134 Then
02,022      IsF12released_Flag = 1
02,023      Else
02,024      End If
02,025      If MustBeLONGsigned \ 256 = 133 Then
02,026      IsF11released_Flag = 1
02,027      Else
02,028      End If
02,029      If MustBeLONGsigned \ 256 = 68 Then '17408=F10
02,030      IsF10released_Flag = 1
02,031      Else
02,032      End If
02,033      If MustBeLONGsigned = 13 Then
02,034      IsENTERreleased_Flag = 1
02,035      Else
02,036      End If
02,037      End If
02,038      End If
02,039
02,040      If LALT_RALT Then
02,041      LastHit"&& = EpochTime"&&: Stepping = 10
02,042      'PLAY "L4V2f" 'removed in order to get rid of audio dependency
02,043      _KeyClear
02,044      ReportTimeToLoad MostRightField
02,045      _Display
02,046      End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,096      j% = 0
02,097      wrd$ = ""
02,098      Do
02,099          Again2:
02,100          j% = j% + 1
02,101          IamStupid$ = Mid$(SuperSlow$, j%, 1)
02,102          If IamStupid$ >= FiChig$ And IamStupid$ <= SeChig$ Then
02,103              wrd$ = wrd$ + IamStupid$
02,104          End If
02,105          If wrd$ = "" And j% < Matrox% Then GoTo Again2
02,106          If wrd$ = "" And j% = Matrox% Then GoTo Nothing2
02,107          If wrd$ <> "" And j% < Matrox% Then
02,108              IamStupid2$ = Mid$(SuperSlow$, j% + 1, 1)
02,109              If IamStupid2$ >= FiChig$ And IamStupid2$ <= SeChig$ Then GoTo Again2
02,110          End If
02,111
02,112          WRDpresent% = 0
02,113          Open "masakari.wrd" For Binary As #6
02,114          If LOF(6) Then
02,115              Close #6
02,116              Open "masakari.ind" For Random Access Read As #6 Len = 32
02,117              LeftPoint& = 1
02,118              RightPoint& = TotalWrd&
02,119              Do
02,120                  If RightPoint& - LeftPoint& <= 1 Then 'Sxsedtcheta
02,121                      Get #6, LeftPoint&, lfixed$
02,122                      If wrd$ = UCase$(RTrim$(lfixed$)) Then WRDpresent% = 1
02,123                      Get #6, RightPoint&, lfixed$
02,124                      If wrd$ = UCase$(RTrim$(lfixed$)) Then WRDpresent% = 1
02,125                      Exit Do
02,126                  End If
02,127                  SplitPoint& = (LeftPoint& + RightPoint&) \ 2
02,128                  Get #6, SplitPoint&, lfixed$
02,129                  If wrd$ < UCase$(RTrim$(lfixed$)) Then
02,130                      RightPoint& = SplitPoint&
02,131                  ElseIf wrd$ > UCase$(RTrim$(lfixed$)) Then
02,132                      LeftPoint& = SplitPoint&
02,133                  Else
02,134                      WRDpresent% = 1: Exit Do
02,135                  End If
02,136              Loop
02,137              Close #6
02,138          Else
02,139              Close #6
02,140              Kill "masakari.wrd"
02,141              WRDpresent% = 2
02,142          End If
02,143          If WRDpresent% = 0 Then
02,144              For Underdog% = j% - Len(wrd$) + 1 To j%

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,194     KeyTap_ForHowLongWasPressed1# = Timer(0.001) - KeyTap_PrevClick1#
02,195 End If
02,196 If (KeyTap_buttondown2 And KeyTap_PrevClick2# <> 0) Then 'already clicked
02,197     KeyTap_ForHowLongWasPressed2# = Timer(0.001) - KeyTap_PrevClick2#
02,198 End If
02,199 If (KeyTap_buttondown3 And KeyTap_PrevClick3# <> 0) Then 'already clicked
02,200     KeyTap_ForHowLongWasPressed3# = Timer(0.001) - KeyTap_PrevClick3#
02,201 End If
02,202 If (KeyTap_buttondown4 And KeyTap_PrevClick4# <> 0) Then 'already clicked
02,203     KeyTap_ForHowLongWasPressed4# = Timer(0.001) - KeyTap_PrevClick4#
02,204 End If
02,205
02,206 If KeyTap_buttondown1 = 0 And KeyTap_PrevClick1# <> 0 Then 'write to the log
02,207     For MumboJumbo = 1 To 3
02,208         KeyTap_Button1LOG_firstDetection#(MumboJumbo) = KeyTap_Button1LOG_firstDetection#(MumboJumbo + 1)
02,209         KeyTap_Button1LOG_ForHowLongHolded#(MumboJumbo) = KeyTap_Button1LOG_ForHowLongHolded#(MumboJumbo + 1)
02,210     Next
02,211     KeyTap_Button1LOG_firstDetection#(4) = KeyTap_PrevClick1#
02,212     KeyTap_Button1LOG_ForHowLongHolded#(4) = KeyTap_ForHowLongWasPressed1#
02,213     KeyTap_PrevClick1# = 0
02,214 End If
02,215
02,216 If KeyTap_buttondown2 = 0 And KeyTap_PrevClick2# <> 0 Then 'write to the log
02,217     For MumboJumbo = 1 To 3
02,218         KeyTap_Button2LOG_firstDetection#(MumboJumbo) = KeyTap_Button2LOG_firstDetection#(MumboJumbo + 1)
02,219         KeyTap_Button2LOG_ForHowLongHolded#(MumboJumbo) = KeyTap_Button2LOG_ForHowLongHolded#(MumboJumbo + 1)
02,220     Next
02,221     KeyTap_Button2LOG_firstDetection#(4) = KeyTap_PrevClick2#
02,222     KeyTap_Button2LOG_ForHowLongHolded#(4) = KeyTap_ForHowLongWasPressed2#
02,223     KeyTap_PrevClick2# = 0
02,224 End If
02,225
02,226 If KeyTap_buttondown3 = 0 And KeyTap_PrevClick3# <> 0 Then 'write to the log
02,227     For MumboJumbo = 1 To 3
02,228         KeyTap_Button3LOG_firstDetection#(MumboJumbo) = KeyTap_Button3LOG_firstDetection#(MumboJumbo + 1)
02,229         KeyTap_Button3LOG_ForHowLongHolded#(MumboJumbo) = KeyTap_Button3LOG_ForHowLongHolded#(MumboJumbo + 1)
02,230     Next
02,231     KeyTap_Button3LOG_firstDetection#(4) = KeyTap_PrevClick3#
02,232     KeyTap_Button3LOG_ForHowLongHolded#(4) = KeyTap_ForHowLongWasPressed3#
02,233     KeyTap_PrevClick3# = 0
02,234 End If
02,235
02,236 If KeyTap_buttondown4 = 0 And KeyTap_PrevClick4# <> 0 Then 'write to the log
02,237     For MumboJumbo = 1 To 3
02,238         KeyTap_Button4LOG_firstDetection#(MumboJumbo) = KeyTap_Button4LOG_firstDetection#(MumboJumbo + 1)
02,239         KeyTap_Button4LOG_ForHowLongHolded#(MumboJumbo) = KeyTap_Button4LOG_ForHowLongHolded#(MumboJumbo + 1)
02,240     Next
02,241     KeyTap_Button4LOG_firstDetection#(4) = KeyTap_PrevClick4#
02,242     KeyTap_Button4LOG_ForHowLongHolded#(4) = KeyTap_ForHowLongWasPressed4#

```

```

02,243     KeyTap_PrevClick4# = 0
02,244 End If
02,245
02,246 If KeyTap_Button1LOG_firstDetection#(3) And (KeyTap_Button1LOG_firstDetection#(4) - KeyTap_Button1LOG_firstDetection#(3) < DclickTime#) Then Double_LSHIFTkey = 1 Else Double_LSHIFTkey = 0
02,247 If KeyTap_Button2LOG_firstDetection#(3) And (KeyTap_Button2LOG_firstDetection#(4) - KeyTap_Button2LOG_firstDetection#(3) < DclickTime#) Then Double_LCTRLkey = 1 Else Double_LCTRLkey = 0
02,248 If KeyTap_Button3LOG_firstDetection#(3) And (KeyTap_Button3LOG_firstDetection#(4) - KeyTap_Button3LOG_firstDetection#(3) < DclickTime#) Then Double_LALTkey = 1 Else Double_LALTkey = 0
02,249 If KeyTap_Button4LOG_firstDetection#(3) And (KeyTap_Button4LOG_firstDetection#(4) - KeyTap_Button4LOG_firstDetection#(3) < DclickTime#) Then Double_RALTkey = 1 Else Double_RALTkey = 0
02,250 ' keyboard mumbo-jumbo - handling of double-hits ]
02,251
02,252 If Double_LSHIFTkey = 1 Then
02,253     LastHit~&& = EpochTime~&&: Stepping = 10
02,254     Double_LSHIFTkey = 0 'clear from buffer
02,255     For MumboJumbo = 1 To 4
02,256         KeyTap_Button1LOG_firstDetection#(MumboJumbo) = 0
02,257         KeyTap_Button1LOG_ForHowLongHeld#(MumboJumbo) = 0
02,258     Next
02,259     crxInfo = Pos(0)
02,260     cryInfo = CsrLin
02,261     Locate YdimROW + 1, MostRightField: Color 4, 0: Print Space$((XdimCOL - MostRightField) - 0);
02,262     Locate YdimROW + 1, MostRightField: Color 4, 0: Print "; Pressed 'Double_LSHIFTkey', but unassigned.";
02,263     Locate cryInfo, crxInfo, 1, CursorS, CursorE
02,264 End If
02,265 If Double_LCTRLkey = 1 Then
02,266     LastHit~&& = EpochTime~&&: Stepping = 10
02,267     Double_LCTRLkey = 0 'clear from buffer
02,268     For MumboJumbo = 1 To 4
02,269         KeyTap_Button2LOG_firstDetection#(MumboJumbo) = 0
02,270         KeyTap_Button2LOG_ForHowLongHeld#(MumboJumbo) = 0
02,271     Next
02,272     crxInfo = Pos(0)
02,273     cryInfo = CsrLin
02,274     Locate YdimROW + 1, MostRightField: Color 4, 0: Print Space$((XdimCOL - MostRightField) - 0);
02,275     Locate YdimROW + 1, MostRightField: Color 4, 0: Print "; Pressed 'Double_LCTRLkey', but unassigned.";
02,276     Locate cryInfo, crxInfo, 1, CursorS, CursorE
02,277 End If
02,278 If Double_LALTkey = 1 Then
02,279     LastHit~&& = EpochTime~&&: Stepping = 10
02,280     Double_LALTkey = 0 'clear from buffer
02,281     For MumboJumbo = 1 To 4
02,282         KeyTap_Button3LOG_firstDetection#(MumboJumbo) = 0
02,283         KeyTap_Button3LOG_ForHowLongHeld#(MumboJumbo) = 0
02,284     Next
02,285     crxInfo = Pos(0)
02,286     cryInfo = CsrLin
02,287     Locate YdimROW + 1, MostRightField: Color 4, 0: Print Space$((XdimCOL - MostRightField) - 0);
02,288     Locate YdimROW + 1, MostRightField: Color 4, 0: Print "; Pressed 'Double_LALTkey', but unassigned.";
02,289     Locate cryInfo, crxInfo, 1, CursorS, CursorE
02,290
02,291     If FULLscr_Flag = 0 Then

```



```

02,292     FULLscr_Flag = 1
02,293     _FullScreen _Stretch , _Smooth
02,294     $If WINDOWS Then
02,295         Play "v20132f"
02,296     $End If
02,297     Else
02,298         FULLscr_Flag = 0
02,299         _FullScreen _Off
02,300     $If WINDOWS Then
02,301         Play "v20132f"
02,302     $End If
02,303     End If
02,304
02,305 End If
02,306 If Double_RALTkey = 1 Then
02,307     LastHit~&& = EpochTime~&&: Stepping = 10
02,308     Double_RALTkey = 0 'clear from buffer
02,309     For MumboJumbo = 1 To 4
02,310         KeyTap_Button4LOG_firstDetection#(MumboJumbo) = 0
02,311         KeyTap_Button4LOG_ForHowLongHoled#(MumboJumbo) = 0
02,312     Next
02,313     crxInfo = Pos(0)
02,314     cryInfo = CsrLin
02,315     Locate YdimROW + 1, MostRightField: Color 4, 0: Print Space$((XdimCOL - MostRightField) - 0);
02,316     Locate YdimROW + 1, MostRightField: Color 4, 0: Print "; Pressed 'Double_RALTkey', but unassigned.";
02,317     Locate cryInfo, crxInfo, 1, CursorS, CursorE
02,318 End If
02,319
02,320 If _KeyDown(LSHIFTkey&) = 0 And _KeyDown(15616) And _KeyDown(LCTRLkey&) Then 'LCtrl+F3
02,321     LastHit~&& = EpochTime~&&: Stepping = 10
02,322     'CALL DrawBox(3, 3, 7, 44, 8, 7)
02,323     'CALL DrawBoxShadow(3, 3, 7, 44)
02,324     'In case of YdimROW = 40 then shrunk panel
02,325     If YdimROW = 40 Then shrunkP = 14 Else shrunkP = 0
02,326     Call DrawBoxShadow3(3, 3, 48 - shrunkP, 107, "LCtrl+F3: Search Panel")
02,327     UpdateWindowFrame NormalFRGr, NormalBCKGr
02,328     _Display
02,329     IsF3released_Flag = 0
02,330     _KeyClear
02,331 End If
02,332
02,333 'IF _KEYDOWN(LSHIFTkey&) = 0 AND _KEYDOWN(15616) AND _KEYDOWN(LALTkey&) THEN 'LAlt+F3released
02,334 If _KeyDown(LSHIFTkey&) = 0 And _KeyDown(15616) And _KeyDown(LALTkey&) = 0 Then 'F3released
02,335     LastHit~&& = EpochTime~&&: Stepping = 10
02,336     Do While _KeyDown(15616): Locate , , 0: _Display: Loop
02,337     'CALL DrawBox(3, 3, 7, 44, 8, 7)
02,338     'CALL DrawBoxShadow(3, 3, 7, 44)
02,339     'In case of YdimROW = 40 then shrunk panel
02,340     shrunkP = 14

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,341      'SUB DrawBoxShadow3 (TopLrow, TopLcol, BottomRrow, BottomRcol, Captmn$)
02,342      F3pattern$ = DrawBoxShadow3exact$(3 + 13, 3, 35 - shrunkP, 107, "F3: Exact Case-Sensitive Search Panel")
02,343      Do Until IsENTERreleased: Locate , , 0: _Display: Loop 'doesnt work on Linux Fedora 33
02,344      UpdateWindowFrame NormalFRGr, NormalBCKGr
02,345      UpdateNextToOCL_F3 MostRightField
02,346      _Display
02,347
02,348      Do While CurrentLine < filecount
02,349          ' Simulating Dn [
02,350          'CASE 80:
02,351          If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,352          If cry < YdimROW Then
02,353              If cry < filecount Then cry = cry + 1 'down
02,354          Else 'scrolling is needed
02,355              If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,356              'UpdateWindowFrame NormalFRGr, NormalBCKGr
02,357          End If
02,358          ' Simulating Dn ]
02,359          TheNewCL$ = FileArray$(CurrentLine)
02,360          If InStr(TheNewCL$, F3pattern$) Then
02,361              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,362              _Display
02,363              Exit Do
02,364          Else
02,365              If CurrentLine = filecount Then
02,366                  UpdateWindowFrame NormalFRGr, NormalBCKGr
02,367                  _Display
02,368                  $If WINDOWS Then
02,369                      Play "v20132c"
02,370                  $End If
02,371                  Exit Do
02,372              End If
02,373          End If
02,374      Loop
02,375
02,376      ' Interactive search, from the line below the INVERSE one, forward ... ]
02,377      UpdateNextToOCL_DONE MostRightField
02,378      IsF3released_Flag = 0
02,379      _KeyClear
02,380      End If
02,381
02,382      'IF _KEYDOWN(LSHIFTkey&) = 0 AND _KEYDOWN(34304) AND _KEYDOWN(LALTkey&) THEN 'LAlt+F12released
02,383      If _KeyDown(LSHIFTkey&) = 0 And _KeyDown(15360) And _KeyDown(LALTkey&) = 0 Then 'F2released
02,384          LastHit"&& = EpochTime"&&: Stepping = 10
02,385          Do While _KeyDown(15360): Locate , , 0: _Display: Loop
02,386          'CALL DrawBox(3, 3, 7, 44, 8, 7)
02,387          'CALL DrawBoxShadow(3, 3, 7, 44)
02,388          'In case of YdimROW = 40 then shrunk panel
02,389          shrunkP = 14

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,390      'SUB DrawBoxShadow3 (TopLrow, TopLcol, BottomRow, BottomRcol, Captnn$)
02,391      F3pattern$ = DrawBoxShadow3exactRussian$(3 + 13, 3, 35 - shrunkP, 107, "F2: Exact Case-Sensitive Search Panel (Russian Phonetic)")
02,392      Do Until IsENTERreleased: Locate , , 0: _Display: Loop 'doesn't work on Linux Fedora 33
02,393      UpdateWindowFrame NormalFRGr, NormalBCKGr
02,394      UpdateNextToOCL_F3 MostRightField
02,395      _Display
02,396
02,397      Do While CurrentLine < filecount
02,398          ' Simulating Dn [
02,399          'CASE 80:
02,400          If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,401          If cry < YdimROW Then
02,402              If cry < filecount Then cry = cry + 1 'down
02,403          Else 'scrolling is needed
02,404              If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,405              'UpdateWindowFrame NormalFRGr, NormalBCKGr
02,406          End If
02,407          ' Simulating Dn ]
02,408          TheNewCL$ = FileArray$(CurrentLine)
02,409          If InStr(TheNewCL$, F3pattern$) Then
02,410              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,411              _Display
02,412              Exit Do
02,413          Else
02,414              If CurrentLine = filecount Then
02,415                  UpdateWindowFrame NormalFRGr, NormalBCKGr
02,416                  _Display
02,417                  $If WINDOWS Then
02,418                      Play "v20132c"
02,419                  $End If
02,420                  Exit Do
02,421              End If
02,422          End If
02,423      Loop
02,424
02,425      ' Interactive search, from the line below the INVERSE one, forward ... ]
02,426      UpdateNextToOCL_DONE MostRightField
02,427      IsF2released_Flag = 0
02,428      _KeyClear
02,429      End If
02,430
02,431      If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) = 0 And _KeyDown(15616) Then 'LShift+(F3 released)
02,432          LastHit~&& = EpochTime~&&: Stepping = 10
02,433          Do While _KeyDown(15616): Locate , , 0: _Display: Loop
02,434          'DO UNTIL IsF3released: LOCATE , , 0: _DISPLAY: LOOP ' doesn't work on Linux Fedora 33
02,435          UpdateNextToOCL_F3 MostRightField
02,436          _Display
02,437          If InStr(_Clipboard$, Chr$(10)) Then ' _CLIPBOARD$ could be multi-line, so get the first
02,438              a2$ = Left$( _Clipboard$, InStr( _Clipboard$, Chr$(10)) - 1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,439     If Right$(a2$, 1) = Chr$(13) Then a2$ = Left$(a2$, Len(a2$) - 1)
02,440     F3pattern$ = a2$
02,441     Else
02,442         F3pattern$ = _Clipboard$
02,443     End If
02,444     ' Interactive search, from the line below the INVERSE one, forward ... [
02,445     Do While CurrentLine < filecount
02,446         ' Simulating Dn [
02,447         'CASE 80:
02,448         If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,449         If cry < YdimROW Then
02,450             If cry < filecount Then cry = cry + 1 'down
02,451         Else 'scrolling is needed
02,452             If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,453             'UpdateWindowFrame NormalFRGr, NormalBCKGr
02,454         End If
02,455         ' Simulating Dn ]
02,456         TheNewCL$ = FileArray$(CurrentLine)
02,457         If InStr(TheNewCL$, F3pattern$) Then
02,458             UpdateWindowFrame NormalFRGr, NormalBCKGr
02,459             _Display
02,460             Exit Do
02,461         Else
02,462             If CurrentLine = filecount Then
02,463                 UpdateWindowFrame NormalFRGr, NormalBCKGr
02,464                 _Display
02,465                 $If WINDOWS Then
02,466                     Play "v20132c"
02,467                 $End If
02,468                 Exit Do
02,469             End If
02,470         End If
02,471     Loop
02,472
02,473     ' Interactive search, from the line below the INVERSE one, forward ... ]
02,474     UpdateNextToOCL_DONE MostRightField
02,475 End If
02,476
02,477 If _KeyDown(LSHIFTkey&) = 0 And _KeyDown(LCTRLkey&) = 0 And _KeyDown(15616) And _KeyDown(LALTkey&) Then 'Alt+F3released
02,478 LastHit"&& = EpochTime"&&: Stepping = 10
02,479 Do While _KeyDown(15616): Locate , , 0: _Display: Loop
02,480 'DO UNTIL IsF3released: LOCATE , , 0: _DISPLAY: LOOP ' doesn't work on Linux Fedora 33
02,481 UpdateNextToOCL_F3 MostRightField
02,482 _Display
02,483 F3pattern$ = CurrentWord$
02,484
02,485     ' Interactive search, from the line below the INVERSE one, forward ... [
02,486
02,487     Do While CurrentLine < filecount

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,488      ' Simulating Dn [
02,489      'CASE 80:
02,490      If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,491      If cry < YdimROW Then
02,492          If cry < filecount Then cry = cry + 1 'down
02,493      Else 'scrolling is needed
02,494          If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,495          'UpdateWindowFrame NormalFRGr, NormalBCKGr
02,496      End If
02,497      ' Simulating Dn ]
02,498      TheNewCL$ = FileArray$(CurrentLine)
02,499      If InStr(TheNewCL$, F3pattern$) Then
02,500          UpdateWindowFrame NormalFRGr, NormalBCKGr
02,501          _Display
02,502          Exit Do
02,503      Else
02,504          If CurrentLine = filecount Then
02,505              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,506              _Display
02,507              $If WINDOWS Then
02,508                  Play "v20132c"
02,509              $End If
02,510              Exit Do
02,511          End If
02,512      End If
02,513      Loop
02,514
02,515      ' Interactive search, from the line below the INVERSE one, forward ... ]
02,516      UpdateNextToOCL_DONE MostRightField
02,517      End If
02,518
02,519      If LALT_Up = 1 Then
02,520          LastHit~&& = EpochTime~&&: Stepping = 10
02,521          If _FileExists(PSPLike$ + "Masakari.RGB") = 0 Then
02,522              GoSub SetColorScheme
02,523              RGB3x16$ = Space$(3 * 16)
02,524              For attribute = 0 To 15
02,525                  Out &H3C7, attribute 'set color attribute to read
02,526                  red = Inp(&H3C9) 'multiply by 4 to convert intensity to 0 to 255 RGB values
02,527                  grn = Inp(&H3C9)
02,528                  blu = Inp(&H3C9)
02,529                  Mid$(RGB3x16$, 3 * attribute + 1, 1) = Chr$(red)
02,530                  Mid$(RGB3x16$, 3 * attribute + 2, 1) = Chr$(grn)
02,531                  Mid$(RGB3x16$, 3 * attribute + 3, 1) = Chr$(blu)
02,532              Next
02,533          Else
02,534              f00 = FreeFile
02,535              Open "Masakari.RGB" For Binary As #f00
02,536              RGB3x16$ = Space$(3 * 16)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,537      Get #f00, , RGB3x16$
02,538      Close #f00
02,539      For ii = 0 To 15
02,540          _PaletteColor ii, _RGB32(Asc(Mid$(RGB3x16$, 3 * ii + 1, 1)) * 4, Asc(Mid$(RGB3x16$, 3 * ii + 2, 1)) * 4, Asc(Mid$(RGB3x16$, 3 * ii + 3, 1)) * 4)
02,541      Next
02,542      $If WINDOWS Then
02,543          Play "v20132c"
02,544      $End If
02,545      End If
02,546
02,547      SimulateBlinking& = 0
02,548      Locate , , 0
02,549      rChannel% = 0
02,550      gChannel% = 0
02,551      bChannel% = 0
02,552      Brightness% = 0
02,553      GoSub UpdateColors
02,554      Do
02,555          reentrance& = _KeyHit
02,556          If reentrance& Then
02,557              Select Case reentrance&
02,558                  Case -114: "r":
02,559                      If rChannel% > -63 Then rChannel% = rChannel% - 1
02,560                      GoSub UpdateColors
02,561                      $If WINDOWS Then
02,562                          Play "v5148d"
02,563                      $End If
02,564                  Case -82 "R":
02,565                      If rChannel% < 63 Then rChannel% = rChannel% + 1
02,566                      GoSub UpdateColors
02,567                      $If WINDOWS Then
02,568                          Play "v5148d"
02,569                      $End If
02,570                  Case -103: "g":
02,571                      If gChannel% > -63 Then gChannel% = gChannel% - 1
02,572                      GoSub UpdateColors
02,573                      $If WINDOWS Then
02,574                          Play "v5148e"
02,575                      $End If
02,576                  Case -71: "G":
02,577                      If gChannel% < 63 Then gChannel% = gChannel% + 1
02,578                      GoSub UpdateColors
02,579                      $If WINDOWS Then
02,580                          Play "v5148e"
02,581                      $End If
02,582                  Case -98: "b":
02,583                      If bChannel% > -63 Then bChannel% = bChannel% - 1
02,584                      GoSub UpdateColors
02,585                      $If WINDOWS Then

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,586             Play "v5148f"
02,587             $End If
02,588             Case -66: "B":
02,589                 If bChannel% < 63 Then bChannel% = bChannel% + 1
02,590                 GoSub UpdateColors
02,591                 $If WINDOWS Then
02,592                     Play "v5148f"
02,593                 $End If
02,594             Case -105: "i":
02,595                 If Brightness% > -63 Then Brightness% = Brightness% - 1
02,596                 GoSub UpdateColors
02,597                 $If WINDOWS Then
02,598                     Play "v5148a"
02,599                 $End If
02,600             Case -73: "I":
02,601                 If Brightness% < 63 Then Brightness% = Brightness% + 1
02,602                 GoSub UpdateColors
02,603                 $If WINDOWS Then
02,604                     Play "v5148a"
02,605                 $End If
02,606             Case Else
02,607             End Select
02,608         End If
02,609         _Limit 20: _Display: ' _Delay .9
02,610         Loop Until reentrance% = -27
02,611         Do While _KeyDown(LALTkey%) Or _KeyDown(UPkey%): _Limit 10: Loop ' to avoid flickering i.e. reentrance
02,612         UpdateWindowFrame NormalFRGr, NormalBCKGr
02,613         _Display
02,614     End If
02,615
02,616     If LALT_Down = 1 Then
02,617         LastHit%&& = EpochTime%&&: Stepping = 10
02,618         'GoSub UpdateColorSchemeFile
02,619         ReDim red(0 To 15)
02,620         ReDim grn(0 To 15)
02,621         ReDim blu(0 To 15)
02,622         For attribute = 0 To 15
02,623             Out &H3C7, attribute 'set color attribute to read
02,624             red = Inp(&H3C9) * 4 'multiply by 4 to convert intensity to 0 to 255 RGB values
02,625             grn = Inp(&H3C9) * 4
02,626             blu = Inp(&H3C9) * 4
02,627             red(attribute) = red
02,628             grn(attribute) = grn
02,629             blu(attribute) = blu
02,630         Next
02,631         'Sepiascale
02,632         'outputRed = (inputRed * .393) + (inputGreen * .769) + (inputBlue * .189)
02,633         'outputGreen = (inputRed * .349) + (inputGreen * .686) + (inputBlue * .168)
02,634         'outputBlue = (inputRed * .272) + (inputGreen * .534) + (inputBlue * .131)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,635     For i = 0 To 15
02,636         'Sepiascale
02,637         _PaletteColor i, _RGB32(Int(red(i) * .393) + Int(grn(i) * .769) + Int(blu(i) * .189), Int(red(i) * .349) + Int(grn(i) * .686) + Int(blu(i) * .168), Int(red(i) * .272) + Int(grn(i) * .534) + Int(blu(i) *
02,638         'Greyscale
02,639         _PaletteColor i, _RGB32((red(i) + grn(i) + blu(i)) \ 3, (red(i) + grn(i) + blu(i)) \ 3, (red(i) + grn(i) + blu(i)) \ 3)
02,640         'Amberscale
02,641     Next
02,642     For i = 1 To 6
02,643         Dummy$ = DrawBoxShadowDUMMY$(i, 2 + (i - 1) * 6, 2, 6 + (i - 1) * 6, 116, "Dummy window with background color" + Str$(i))
02,644     Next
02,645     Locate , , 0
02,646     Do While InKey$ = ""
02,647         SimulateBlinking& = Not SimulateBlinking&
02,648         If SimulateBlinking& = 0 Then
02,649             Locate 38, 2: Color 0, 7: Print "Note1: File 'Masakari.RGB' has NOT been updated with current (before Greyscaling) color settings.";
02,650             Locate 39, 2: Color 0, 7: Print "Note2: In order to have your own color scheme, you can manually change the values of 'Masakari.RGB' (3x16 bytes long).";
02,651         Else
02,652             'Locate 38, 2: Color 15, 7: Print "Note1: File 'Masakari.RGB' has been updated with current (before Greyscaling) color settings.";
02,653             'Locate 39, 2: Color 15, 7: Print "Note2: In order to have your own color scheme, you can manually change the values of 'Masakari.RGB' (3x16 bytes long).";
02,654         End If
02,655         _Limit 20: _Display: ' _Delay .9
02,656     Loop
02,657     Do While _KeyDown(LALTkey&) Or _KeyDown(DOWNkey&): _Limit 10: Loop ' to avoid flickering i.e. reentrance
02,658     UpdateWindowFrame NormalFRGr, NormalBCKGr
02,659     _Display
02,660 End If
02,661
02,662 If _KeyDown(27) Then
02,663     LastHit~&& = EpochTime~&&: Stepping = 10
02,664     LineScrollBenchmark = 0
02,665     PageScrollBenchmark = 0
02,666 End If
02,667
02,668 If LSHIFT_RSHIFT Or LineScrollBenchmark = 1 Then 'same as scroll page-by-page benchmark, but line-by-line
02,669     LastHit~&& = EpochTime~&&: Stepping = 10
02,670     If LineScrollBenchmark = 0 Then TimeScroll1A = Timer
02,671     LineScrollBenchmark = 1
02,672     'PgDn - just add the page height i.e. 'YdimROW' to 'File_Frame_y'
02,673     ' Don't execute PgDn (advancing the 'CurrentLine') if 'File_Frame_y' is not "eligible":
02,674     'IF File_Frame_y < filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
02,675     'IF File_Frame_y + 1 <= filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
02,676     'CAUTION [
02,677     ' Next three line beep NOT
02,678     'DEFLNG A-Z
02,679     'aaa = 61
02,680     'bbb = -48
02,681     'IF aaa <= bbb THEN BEEP: END
02,682     'Next three line beep

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

02,683 'aaa~& = 61
02,684 'bbb~& = -48
02,685 'IF aaa~& <= bbb~& THEN BEEP: END
02,686 'Next three line beep NOT
02,687 'aaa& = 61
02,688 'bbb& = -48
02,689 'IF aaa& <= bbb& THEN BEEP: END
02,690 'Next line works even when unsigned!
02,691 'IF File_Frame_y + YdimROW - (filecount - (YdimROW - 1)) <= 0 THEN
02,692 'CAUTION ]
02,693 'Next line doesn't work when unsigned!
02,694 If File_Frame_y + 1 <= filecount - (YdimROW - 1) Then
02,695     File_Frame_y = File_Frame_y + 1
02,696     'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
02,697     'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
02,698     If CurrentLine + 1 <= filecount Then CurrentLine = CurrentLine + 1
02,699     UpdateWindowFrame NormalFRGr, NormalBCKGr
02,700 Else
02,701     If LineScrollBenchmark = 1 Then
02,702         TimeScrollB = Timer
02,703         _KeyClear
02,704         ReportTimeToScroll MostRightField
02,705         LineScrollBenchmark = 0
02,706         _Display
02,707     End If
02,708 End If
02,709 End If
02,710
02,711 ' Esc F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 Sys ScL Pause
02,712 ' 27 +59 +60 +61 +62 +63 +64 +65 +66 +67 +68 +133 +134 - - -
02,713 ' '~ 1! 2@ 3# 4$ 5% 6^ 7& 8* 9( 0) - += BkSp Ins Hme PUp NumL / * -
02,714 ' 126 33 64 35 36 37 94 38 42 40 41 95 43 8 +82 +71 +73 - 47 42 45
02,715 ' 96 49 50 51 52 53 54 55 56 57 48 45 61
02,716 ' Tab Q W E R T Y U I O P [{ ]} \| Del End PDn 7Hme 8/ 9PU +
02,717 ' 9 81 87 69 82 84 89 85 73 79 80 123 125 124 +83 +79 +81 +71 +72 +73 43
02,718 ' 113 119 101 114 116 121 117 105 111 112 91 93 92 55 56 57
02,719 ' CapL A S D F G H J K L ;: '" Enter 4/- 5 6/-
02,720 ' - 65 83 68 70 71 72 74 75 76 58 34 13 +75 +76 +77 E
02,721 ' 97 115 100 102 103 104 106 107 108 59 39 52 53 54 n
02,722 '
02,723 ' Shift Z X C V B N M ,< .> /? Shift 1End 2/ 3PD t
02,724 ' * 90 88 67 86 66 78 77 60 62 63 * +72 +79 +80 +81 e
02,725 ' 122 120 99 118 98 110 109 44 46 47 49 50 51 r
02,726 ' Ctrl Win Alt Spacebar Alt Win Menu Ctrl - - 0Ins .Del
02,727 ' * - * 32 * - - * +75 +80 +77 +82 +83 13
02,728 ' 48 46
02,729 '
02,730 ' Italics = LCase/NumLock On _____ + = 2 Byte: CHR$(0) + CHR$(code)
02,731 'NOTE: The above commented table can be copied and pasted directly into the QB64 IDE

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,732
02,733 key$ = InKey$
02,734 'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
02,735 If key$ <> "" Then
02,736     LastHit~&& = EpochTime~&&: Stepping = 10
02,737     code% = Asc(key$):
02,738     If code% Then ' ASC returns any value greater than 0
02,739         Select Case Asc(key$)
02,740             Case 32:
02,741                 a$ = RTrim$(FileArray$(CurrentLine))
02,742                 If _FileExists(a$) Then
02,743                     Close #1
02,744                     _MemFree MhandleOFF
02,745                     _MemFree MhandleLEN
02,746                     If ToLoadOrNotFlag Then
02,747                         _MemFree Mwholefile
02,748                     End If
02,749                     GoTo GettingStarted 'Before going above lines must be executed, i.e. to initialize.
02,750                 End If
02,751                 Case 65 TO 97: 'PRINT key$;
02,752                 Case Asc("a") TO Asc("z"): 'PRINT key$;
02,753                 'CASE 27: COLOR 7, 0: SYSTEM 'END
02,754             End Select
02,755         Else
02,756             Select Case Asc(key$, 2)
02,757                 Case 59:
02,758                     Locate 1, 1, 0, CursorS, CursorE
02,759                     For i = 1 To YdimROW
02,760                         Locate i, 1: Print Space$(XdimCOL);
02,761                     Next
02,762                     Locate 1, 1
02,763                     ShowF1
02,764                     Print "Press ESC...";
02,765                     _Display
02,766                     $If WINDOWS Then
02,767                         Play "v20120g"
02,768                     $End If
02,769                     Do While InKey$ <> Chr$(27)
02,770                         _Limit 10
02,771                         _Display 'in order to not having frozen screen
02,772                     Loop
02,773                     UpdateWindowFrame NormalFRGr, NormalBCKGr
02,774                 Case 133: 'f11
02,775
02,776                 Case 134: 'f12
02,777
02,778                 Case 72:
02,779                     If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
02,780                     If cry > 1 Then

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,781         cry = cry - 1 'up
02,782         Else 'scrolling is needed
02,783             If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
02,784             UpdateWindowFrame NormalFRGr, NormalBCKGr
02,785         End If
02,786     Case 80:
02,787         If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,788         If cry < YdimROW Then
02,789             If cry < filecount Then cry = cry + 1 'down
02,790             Else 'scrolling is needed
02,791                 If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,792                 UpdateWindowFrame NormalFRGr, NormalBCKGr
02,793             End If
02,794         Case 75: If crx > 1 Then crx = crx - 1 'left
02,795         Case 77: If crx < XdimCOL Then crx = crx + 1 'right
02,796         Case 73: 'PgUp
02,797             If File_Frame_y - YdimROW >= 1 Then
02,798                 File_Frame_y = File_Frame_y - YdimROW
02,799                 If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
02,800                 UpdateWindowFrame NormalFRGr, NormalBCKGr
02,801             End If
02,802         Case 81: 'PgDn
02,803             If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
02,804                 File_Frame_y = File_Frame_y + YdimROW
02,805                 'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
02,806                 'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
02,807                 If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
02,808                 UpdateWindowFrame NormalFRGr, NormalBCKGr
02,809             End If
02,810         End Select
02,811     End If
02,812 End If
02,813 If cryOLD <> cry Then
02,814     UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
02,815     cryOLD = cry
02,816 Else 'it 'cry' could be changed by Mouse Wheel too, check it
02,817     AsIfItIsINKEY% = _MouseInput ' Check the mouse status
02,818     'WHILE _MOUSEINPUT: WEND '2021-Apr-23
02,819     If AsIfItIsINKEY% Then
02,820         buttondown1 = _MouseButton(1)
02,821         buttondown2 = _MouseButton(2)
02,822         buttondown3 = _MouseButton(3)
02,823         mwheel = _MouseWheel
02,824     End If
02,825
02,826     If IsF10released_Flag Then
02,827         LastHit~&& = EpochTime~&&: Stepping = 10
02,828         UpdateCLineHIGHLIGHTword cry, 1, InverseFRGr, InverseBCKGr, cry
02,829         Locate 1, 1, 0, CursorS, CursorE: _Display 'adding refresh in order to get rid of eventual cursor (caught as visible while blinking)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,830      Color 9, 0
02,831      Call dict03sub
02,832      _KeyClear 2
02,833      UpdateWindowFrame NormalFRGr, NormalBCKGr
02,834      End If
02,835
02,836      If IsF11released_Flag Then
02,837          LastHit~&& = EpochTime~&&: Stepping = 10
02,838          UpdateCLineHIGHLIGHTword cry, 1, InverseFRGr, InverseBCKGr, cry
02,839          Locate 1, 1, 0, Cursor5, CursorE: _Display 'adding refresh in order to get rid of eventual cursor (caught as visible while blinking)
02,840          Color 9, 0
02,841          Call dict02sub
02,842          _KeyClear 2
02,843          UpdateWindowFrame NormalFRGr, NormalBCKGr
02,844      End If
02,845
02,846      If IsF12released_Flag Then
02,847          LastHit~&& = EpochTime~&&: Stepping = 10
02,848          UpdateCLineHIGHLIGHTword cry, 1, InverseFRGr, InverseBCKGr, cry
02,849          Locate 1, 1, 0, Cursor5, CursorE: _Display 'adding refresh in order to get rid of eventual cursor (caught as visible while blinking)
02,850          Color 9, 0
02,851          Call dict01sub
02,852          _KeyClear 2
02,853          UpdateWindowFrame NormalFRGr, NormalBCKGr
02,854      End If
02,855
02,856      If _KeyDown(LSHIFTkey&) Or _KeyDown(LCTRLkey&) Or _KeyDown(LALTkey&) Then ShutDownIsNoLonger = 1: ShutDownIsNoLongerR = 1 ' do not forget there are key+button shortcuts
02,857
02,858      If (buttondown1 And PrevClick1# = 0) Then 'first detection of the click
02,859          PrevClick1# = Timer(0.001)
02,860          SaveMouseX2 = _MouseX
02,861          SaveMouseY2 = _MouseY
02,862          ShutDownIsNoLonger = 0
02,863      End If
02,864
02,865      If (buttondown2 And PrevClick2# = 0) Then 'first detection of the click
02,866          PrevClick2# = Timer(0.001)
02,867          SaveMouseXR = _MouseX
02,868          SaveMouseYR = _MouseY
02,869          ShutDownIsNoLongerR = 0
02,870      End If
02,871
02,872      If (buttondown1 And PrevClick1# <> 0) Then 'already clicked
02,873          ForHowLongWasPressed1# = Timer(0.001) - PrevClick1#
02,874          If (SaveMouseX2 <> _MouseX) Or (SaveMouseY2 <> _MouseY) Then ShutDownIsNoLonger = 1
02,875      End If
02,876
02,877      If (buttondown2 And PrevClick2# <> 0) Then 'already clicked
02,878          ForHowLongWasPressed2# = Timer(0.001) - PrevClick2#

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,879     If (SaveMouseXR < _MouseX) Or (SaveMouseYR < _MouseY) Then ShutDownIsNoLongerR = 1
02,880     End If
02,881
02,882     If buttowndown1 = 0 And PrevClick1# <> 0 Then 'write to the log
02,883         For MumboJumbo = 1 To 3
02,884             Button1LOG_firstDetection#(MumboJumbo) = Button1LOG_firstDetection#(MumboJumbo + 1)
02,885             Button1LOG_ForHowLongHolded#(MumboJumbo) = Button1LOG_ForHowLongHolded#(MumboJumbo + 1)
02,886         Next
02,887         Button1LOG_firstDetection#(4) = PrevClick1#
02,888         Button1LOG_ForHowLongHolded#(4) = ForHowLongWasPressed1#
02,889         PrevClick1# = 0
02,890     End If
02,891
02,892     If buttowndown2 = 0 And PrevClick2# <> 0 Then 'write to the log
02,893         For MumboJumbo = 1 To 3
02,894             Button2LOG_firstDetection#(MumboJumbo) = Button2LOG_firstDetection#(MumboJumbo + 1)
02,895             Button2LOG_ForHowLongHolded#(MumboJumbo) = Button2LOG_ForHowLongHolded#(MumboJumbo + 1)
02,896         Next
02,897         Button2LOG_firstDetection#(4) = PrevClick2#
02,898         Button2LOG_ForHowLongHolded#(4) = ForHowLongWasPressed2#
02,899         PrevClick2# = 0
02,900     End If
02,901
02,902     If Button1LOG_firstDetection#(3) And (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3) < DclickTime#) Then Double_LeftClick = 1 Else Double_LeftClick = 0
02,903     'IF Double_LeftClick = 1 THEN PRINT "Double_LeftClick detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3)) * 1000); "ms."
02,904
02,905     If Button2LOG_firstDetection#(3) And (Button2LOG_firstDetection#(4) - Button2LOG_firstDetection#(3) < DclickTime#) Then Double_RightClick = 1 Else Double_RightClick = 0
02,906     'IF Double_LeftClick = 1 THEN PRINT "Double_LeftClick detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3)) * 1000); "ms."
02,907
02,908     If (ForHowLongWasPressed1# > ShutDownDuration#) And (ShutDownIsNoLonger = 0) Then 'the pointer should not be moved (in perfect case), inhere should be as the pointer during the initial click
02,909         ShutDownIsNoLonger = 1
02,910         System
02,911     End If
02,912
02,913     If (ForHowLongWasPressed2# > ShutDownDuration#) And (ShutDownIsNoLongerR = 0) Then 'the pointer should not be moved (in perfect case), inhere should be as the pointer during the initial click
02,914         ShutDownIsNoLongerR = 1
02,915         'SimulateTabRel = 1
02,916         'add some common task here, mapped onto held Button2
02,917     End If
02,918 End If
02,919
02,920 If _KeyDown(LSHIFTkey&) And buttowndown1 Then 'Fast Up
02,921     LastHit~&& = EpochTime~&&; Stepping = 10
02,922     If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
02,923     If cry > 1 Then
02,924         cry = cry - 1 'up
02,925     Else 'scrolling is needed
02,926         If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
02,927         UpdateWindowFrame NormalFRGr, NormalBCKGr

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,928      End If
02,929      End If
02,930
02,931      If _KeyDown(LSHIFTkey&) And buttndown2 Then 'Fast Down
02,932          LastHit~&& = EpochTime~&&: Stepping = 10
02,933          If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,934          If cry < YdimROW Then
02,935              If cry < filecount Then cry = cry + 1 'down
02,936          Else 'scrolling is needed
02,937              If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
02,938              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,939          End If
02,940      End If
02,941
02,942      If _KeyDown(LALTkey&) And buttndown1 Then 'CTRL_HOME
02,943          LastHit~&& = EpochTime~&&: Stepping = 10
02,944          LCTRL_HOME = 1
02,945      End If
02,946      If _KeyDown(LALTkey&) And buttndown2 Then 'CTRL_END
02,947          LastHit~&& = EpochTime~&&: Stepping = 10
02,948          LCTRL_END = 1
02,949      End If
02,950
02,951      If _KeyDown(LCTRLkey&) And buttndown1 Then 'Fast PgUp
02,952          LastHit~&& = EpochTime~&&: Stepping = 10
02,953          If File_Frame_y - YdimROW >= 1 Then
02,954              File_Frame_y = File_Frame_y - YdimROW
02,955              If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
02,956              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,957          End If
02,958      End If
02,959
02,960      If _KeyDown(LCTRLkey&) And buttndown2 Then 'Fast PgDn, grmb1, for some reason (LShift+button3) and (LShift+button1) are not spitting?!
02,961          LastHit~&& = EpochTime~&&: Stepping = 10
02,962          If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
02,963              File_Frame_y = File_Frame_y + YdimROW
02,964              'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
02,965              'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
02,966              If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
02,967              UpdateWindowFrame NormalFRGr, NormalBCKGr
02,968          End If
02,969      End If
02,970
02,971      If mwheel = 1 Then ' as if Down
02,972          If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
02,973          If cry < YdimROW Then
02,974              If cry < filecount Then cry = cry + 1 'down
02,975          Else 'scrolling is needed
02,976              If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,977         UpdateWindowFrame NormalFRGr, NormalBCKGr
02,978     End If
02,979 End If
02,980 If mwheel = -1 Then ' as if Up
02,981     If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
02,982     If cry > 1 Then
02,983         cry = cry - 1 'up
02,984     Else 'scrolling is needed
02,985         If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
02,986         UpdateWindowFrame NormalFRGr, NormalBCKGr
02,987     End If
02,988 End If
02,989 If (_KeyDown(LSHIFTkey&) = 0) And buttndown1 Then
02,990     crxOLD = crx
02,991     cryOLD = cry
02,992     If _MouseY <= MIN8(YdimROW, filecount) Then
02,993         cry = _MouseY
02,994         crx = _MouseX
02,995         UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
02,996         Do While cryOLD > cry
02,997             cryOLD = cryOLD - 1
02,998             If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
02,999         Loop
03,000         Do While cryOLD < cry
03,001             cryOLD = cryOLD + 1
03,002             If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
03,003         Loop
03,004     End If
03,005 End If
03,006
03,007 If (mwheel = -1) And buttndown2 Then ' same as LCTRL_HOME
03,008     LCTRL_HOME = 1
03,009 End If
03,010 If (mwheel = 1) And buttndown2 Then ' same as LCTRL_END
03,011     LCTRL_END = 1
03,012 End If
03,013
03,014 If buttndown2 Then ' 'drawing a line' gesture - 100 cells at least long, COMBO: Alt+X, Alt+Q
03,015     ReadOnceY = _MouseY
03,016     ReadOnceX = _MouseX
03,017     If Button2Down = 0 Then Mouse2Press! = Timer: Button2Down = 1
03,018     If WidenessINIT = 0 Then WidenessINIT = ReadOnceX
03,019     If HighnessINIT = 0 Then HighnessINIT = ReadOnceY
03,020     If ReadOnceY < YdimROW + 1 Then Locate ReadOnceY, ReadOnceX: Print Chr$(176); 'don't write trail in status line
03,021 Else
03,022     Button2Down = 0
03,023     Mouse2Release! = Timer
03,024     If WidenessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
03,025         If Abs(_MouseX - WidenessINIT) >= 100 Then System 'PLAY "v1018g" 'bidirectional i.e. can be drawn from left to right and vice versa

```

```

03,026      WidenessINIT = 0
03,027      End If
03,028      'simulate PgDn (from bottom to top, as the hand cursor in PhotoShop), but not repetitive i.e. not-buffered
03,029      If HighnessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
03,030          If (HighnessINIT - _MouseY) >= 5 Then 'PLAY "v1018g" 'bidirectional i.e. can be drawn from left to right and vice versa
03,031
03,032              If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
03,033                  File_Frame_y = File_Frame_y + YdimROW
03,034                  'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
03,035                  'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
03,036                  If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
03,037                  UpdateWindowFrame NormalFRGr, NormalBCKGr
03,038              End If
03,039              ' Here is the layout:
03,040              ' The window-frame is 1..crx or 1..128 | The file-frame is 1..LongestLine
03,041              '
03,042              '
03,043              '          cry          60          |          filecount
03,044              '      FileArrayWINDOW$(60) |      FileArray(filecount)
03,045              ' if File_Frame_x < 128 then PADDING to 128 else File_Frame_x = 1..LongestLine-(128-1)
03,046              ' if File_Frame_y < 60 then PADDING to 60 else File_Frame_y = 1..filecount-(60-1)
03,047
03,048          End If
03,049      End If
03,050      'simulate PgUp (from top to bottom, as the hand cursor in PhotoShop), but not repetitive i.e. not-buffered
03,051      If HighnessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
03,052          If (_MouseY - HighnessINIT) >= 5 Then 'PLAY "v1018g" 'bidirectional i.e. can be drawn from left to right and vice versa
03,053
03,054              If File_Frame_y - YdimROW >= 1 Then
03,055                  File_Frame_y = File_Frame_y - YdimROW
03,056                  If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
03,057                  UpdateWindowFrame NormalFRGr, NormalBCKGr
03,058              End If
03,059
03,060          End If
03,061      End If
03,062      HighnessINIT = 0
03,063      End If
03,064
03,065      If LCTRL_BCTRL Then PageScrollBenchmark = 1: TimeScrollA = Timer
03,066      If buttowndown3 Or PageScrollBenchmark Then 'PgDn - just add the page height i.e. 'YdimROW' to 'File_Frame_y'
03,067          LastHit~&& = EpochTime~&&: Stepping = 10
03,068          ' Don't execute PgDn (advancing the 'CurrentLine') if 'File_Frame_y' is not "eligible":
03,069          'IF File_Frame_y < filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
03,070          'IF File_Frame_y + 1 <= filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
03,071          'CAUTION [
03,072          ' Next three line beep NOT
03,073          'DEFLNG A-Z
03,074          'aaa = 61

```



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,075      'bbb = -48
03,076      'IF aaa <= bbb THEN BEEP: END
03,077      'Next three line beep
03,078      'aaa~& = 61
03,079      'bbb~& = -48
03,080      'IF aaa~& <= bbb~& THEN BEEP: END
03,081      'Next three line beep NOT
03,082      'aaa& = 61
03,083      'bbb& = -48
03,084      'IF aaa& <= bbb& THEN BEEP: END
03,085      'Next line works even when unsigned!
03,086      'IF File_Frame_y + YdimROW - (filecount - (YdimROW - 1)) <= 0 THEN
03,087      'CAUTION ]
03,088      'Next line doesn't work when unsigned!
03,089      If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
03,090          File_Frame_y = File_Frame_y + YdimROW
03,091          'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
03,092          'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
03,093          If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
03,094          UpdateWindowFrame NormalFRGr, NormalBCKGr
03,095      Else
03,096          If PageScrollBenchmark = 1 Then
03,097              TimeScrollB = Timer
03,098              _KeyClear
03,099              ReportTimeToScroll MostRightField
03,100              PageScrollBenchmark = 0
03,101              Display
03,102          End If
03,103      End If
03,104      End If
03,105
03,106      If cryOLD <> cry Then
03,107          UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
03,108          cryOLD = cry
03,109      End If
03,110      If LCTRL_HOME Then
03,111          LastHit~&& = EpochTime~&&: Stepping = 10
03,112          Locate 1, 1, 1, CursorS, CursorE
03,113          crx = Pos(0)
03,114          cry = CsrLin
03,115          crxOLD = crx
03,116          cryOLD = cry
03,117          File_Frame_x = 1
03,118          File_Frame_y = 1
03,119          CurrentLine = 1
03,120          UpdateWindowFrame NormalFRGr, NormalBCKGr
03,121          'DO WHILE _MOUSEINPUT: LOOP
03,122      End If
03,123      If LCTRL_END Then

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,124      LastHit~&& = EpochTime~&&: Stepping = 10
03,125      If filecount >= YdimROW Then
03,126          If filecount - (YdimROW - 1) Then
03,127              Locate YdimROW, 1, 1, CursorS, CursorE
03,128              crx = Pos(0)
03,129              cry = CsrLin
03,130              crxOLD = crx
03,131              cryOLD = cry
03,132              File_Frame_x = 1
03,133              File_Frame_y = filecount - (YdimROW - 1)
03,134              CurrentLine = filecount
03,135              UpdateWindowFrame NormalFRGr, NormalBCKGr
03,136          End If
03,137      Else
03,138          Locate filecount, 1, 1, CursorS, CursorE
03,139          crx = Pos(0)
03,140          cry = CsrLin
03,141          crxOLD = crx
03,142          cryOLD = cry
03,143          File_Frame_x = 1
03,144          File_Frame_y = 1
03,145          CurrentLine = filecount
03,146          UpdateWindowFrame NormalFRGr, NormalBCKGr
03,147      End If
03,148      'DO WHILE _MOUSEINPUT: LOOP
03,149  End If
03,150
03,151  ' @QB64team shared this clear code:
03,152  'DO
03,153  '  WHILE _MOUSEINPUT: WEND
03,154  '  IF _MOUSEBUTTON(1) THEN
03,155  '    IF timeElapsedSince!(lastClick!) <= .3 THEN
03,156  '      PRINT "... and make it double!"
03,157  '    ELSE
03,158  '      PRINT
03,159  '      PRINT "Click!";
03,160  '    END IF
03,161  '    lastClick! = TIMER
03,162  '    WHILE _MOUSEBUTTON(1): i = _MOUSEINPUT: WEND
03,163  '  END IF
03,164  '  _LIMIT 60
03,165  'LOOP UNTIL _KEYHIT = 27
03,166
03,167  'FUNCTION timeElapsedSince! (startTime!)
03,168  '  IF startTime! > TIMER THEN startTime! = startTime! - 86400
03,169  '  timeElapsedSince! = TIMER - startTime!
03,170  'END FUNCTION
03,171
03,172  'WHILE _MOUSEINPUT: WEND

```

```

03,173 'IF _MOUSEBUTTON(1) THEN
03,174 '   IF timeElapsedSince!(lastClick!) <= .3 THEN
03,175 '       ' Double-click
03,176 '       a$ = RTRIM$(FileArray$(CurrentLine))
03,177 '       IF _FILEEXISTS(a$) THEN
03,178 '           CLOSE #1
03,179 '           _MEMFREE MhandleOFF
03,180 '           _MEMFREE MhandleLEN
03,181 '           IF ToLoadOrNotFlag THEN
03,182 '               MEMFREE Mwholefile
03,183 '           END IF
03,184 '           GOTO GettingStarted 'Before going above lines must be executed, i.e. to initialize.
03,185 '       ELSE ' not responsive enough - it misses some double-clicks?!
03,186 '           IF File_Frame_y - YdimROW >= 1 THEN
03,187 '               File_Frame_y = File_Frame_y - YdimROW
03,188 '               IF CurrentLine - YdimROW >= 1 THEN CurrentLine = CurrentLine - YdimROW
03,189 '               UpdateWindowFrame NormalFRGr, NormalBCKGr
03,190 '           END IF
03,191 '       END IF
03,192 '   ELSE
03,193 '       ' Single-click
03,194 '   END IF
03,195 '   lastClick! = TIMER
03,196 '   WHILE _MOUSEBUTTON(1): idummy = _MOUSEINPUT: WEND
03,197 'END IF
03,198
03,199 ' Grmbl, my doulbe-click sucks, therefore going to limbo... Yet, the above replacement nullifies/discards further events :( thus banning my other mouse functionality!
03,200 ' No grmbles anymore, fixed:
03,201 If Double_LeftClick Then
03,202     a$ = RTrim$(FileArray$(CurrentLine))
03,203     If _FileExists(a$) Then
03,204         Close #1
03,205         _MemFree MhandleOFF
03,206         _MemFree MhandleLEN
03,207         If ToLoadOrNotFlag Then
03,208             MemFree Mwholefile
03,209         End If
03,210         GoTo GettingStarted 'Before going above lines must be executed, i.e. to initialize.
03,211     Else ' not responsive enough - it misses some double-clicks?!
03,212         If File_Frame_y - YdimROW >= 1 Then
03,213             File_Frame_y = File_Frame_y - YdimROW
03,214             If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
03,215             UpdateWindowFrame NormalFRGr, NormalBCKGr
03,216         End If
03,217     End If
03,218     Double_LeftClick = 0 'clear from buffer
03,219     For MumboJumbo = 1 To 4
03,220         Button1LOG_firstDetection#(MumboJumbo) = 0
03,221         Button1LOG_ForHowLongHolded#(MumboJumbo) = 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,222     Next
03,223 End If
03,224
03,225 'WHILE _MOUSEINPUT: WEND
03,226 'IF _MOUSEBUTTON(2) THEN
03,227 '   IF timeElapsedSince!(lastClick!) <= .3 THEN
03,228 '       ' Double-click
03,229 '       IF File_Frame_y + YdimROW <= filecount - (YdimROW - 1) THEN
03,230 '           File_Frame_y = File_Frame_y + YdimROW
03,231 '           'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
03,232 '           'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
03,233 '           IF CurrentLine + YdimROW <= filecount THEN CurrentLine = CurrentLine + YdimROW
03,234 '           UpdateWindowFrame NormalFRGr, NormalBCKGr
03,235 '       END IF
03,236 '   ELSE
03,237 '       ' Single-click
03,238 '   END IF
03,239 '   lastClick! = TIMER
03,240 '   WHILE _MOUSEBUTTON(2): idummy = _MOUSEINPUT: WEND
03,241 'END IF
03,242
03,243 ' Grmbl, my doulbe-click sucks, therefore going to limbo... Yet, the above replacement nullifies/discards further events :( thus banning my other mouse functionality!
03,244 ' No grmbls anymore, fixed:
03,245 If Double_RightClick Then
03,246     If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
03,247         File_Frame_y = File_Frame_y + YdimROW
03,248         'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
03,249         'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
03,250         If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
03,251         UpdateWindowFrame NormalFRGr, NormalBCKGr
03,252     End If
03,253     Double_RightClick = 0 'clear from buffer
03,254     For MumboJumbo = 1 To 4
03,255         ButtonZLOG_firstDetection#(MumboJumbo) = 0
03,256         ButtonZLOG_ForHowLongHolded#(MumboJumbo) = 0
03,257     Next
03,258 End If
03,259
03,260 Locate cry, crx, 1, CursorS, CursorE
03,261 UpdateCLine cry, 1, InverseFRGr, InverseBCKGr, cry
03,262 UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1
03,263 _Display
03,264
03,265 If buttondown2 And buttondown1 Then 'pagoda, GRMBL, for now the file being handled SHOULD be in the invocation/home folder!
03,266     buttondown2 = 0
03,267     buttondown1 = 0 ' have to nullify because the cleaning of mouseinputs below prevents zeroing the variable in the main cycle where release is awaited but never registered.
03,268     ShutDownIsNoLonger = 1
03,269     UpdateCLineHIGHLIGHTword cry, 1, InverseFRGr, InverseBCKGr, cry
03,270     UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,271 Locate cry, crx, 0, CursorS, CursorE 'hide the cursor
03,272 _Display
03,273 If Len(CurrentWord$) > 1 Then
03,274 UpdateNextToCCL_BUSY (MostRightField): _Display
03,275 p$ = LoadedFile$
03,276 If InStr(p$, Slash) Then
03,277 pTRIMMED = 0
03,278 Do Until Mid$(p$, Len(p$) - pTRIMMED, 1) = Slash
03,279 pTRIMMED = pTRIMMED + 1
03,280 Loop
03,281 p$ = Right$(p$, pTRIMMED)
03,282 End If
03,283 $If WINDOWS Then
03,284 $If 32BIT Then
03,285 Shell _Hide "call " + "XGRAM_PAGODA5_32bit.bat" + " " + LCase$(CurrentWord$) + " " + Chr$(34) + p$ + Chr$(34)
03,286 $End If
03,287 $If 64BIT Then
03,288 SHELL _HIDE "call " + "XGRAM_PAGODA5_64bit.bat" + " " + LCase$(CurrentWord$) + " " + Chr$(34) + p$ + Chr$(34)
03,289 $End If
03,290 $Else
03,291 SHELL _HIDE "sh " + "XGRAM_PAGODA5.sh" + " " + LCase$(CurrentWord$) + " " + Chr$(34) + p$ + Chr$(34)
03,292 $End If
03,293 UpdateNextToCCL_DONE (MostRightField): _Display
03,294 While _MouseInput: Wend
03,295
03,296 Locate 1, 1, 0, CursorS, CursorE
03,297 For i = 1 To YdimROW
03,298 Locate i, 1: Print Space$(XdimCOL);
03,299 Next
03,300
03,301 If _FileExists(PSPlike$ + p$ + "_" + LCase$(CurrentWord$) + ".PAGODA-order-5.txt") Then
03,302 reportNAME$ = p$ + "_" + LCase$(CurrentWord$) + ".PAGODA-order-5.txt"
03,303 pgd& = FreeFile
03,304 Open PSPlike$ + p$ + "_" + LCase$(CurrentWord$) + ".PAGODA-order-5.txt" For Input As #pgd&
03,305 pgdlines = 0
03,306 Do While Not EOF(pgd&)
03,307 pgdlines = pgdlines + 1
03,308 Line Input #pgd&, dummy1$
03,309 Loop
03,310 Close #pgd&
03,311 ReDim xgrams$(1 To pgdlines)
03,312 SCROLLABLEheight = (YdimROW - 2) - 4 + 1
03,313 pgd& = FreeFile
03,314 Open PSPlike$ + p$ + "_" + LCase$(CurrentWord$) + ".PAGODA-order-5.txt" For Input As #pgd&
03,315 Locate 1, 1
03,316 Print "PAGODA size: "; AddCommas$(pgdlines); " lines or "; AddCommas$(LOF(pgd&)); " bytes";
03,317 Locate 2, 1
03,318 Print "PAGODA name: "; Left$(reportNAME$, XdimCOL - Len("PAGODA name: "));
03,319 pgdlines = 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,320      Do While Not EOF(pgd&)
03,321          pgdlines = pgdlines + 1
03,322          Line Input #pgd&, xgrams$(pgdlines)
03,323          For j = 1 To Len(xgrams$(pgdlines))
03,324              If pgdlines Mod 2 = 0 Then
03,325                  If Mid$(xgrams$(pgdlines), j, 1) = " " Then Mid$(xgrams$(pgdlines), j, 1) = "."
03,326              End If
03,327          Next
03,328      Loop
03,329      Close #pgd&
03,330      For i = 1 To MIN(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,331          Locate i + 3, 1: Print Left$(xgrams$(i), XdimCOL);
03,332      Next
03,333
03,334      End If
03,335      Print
03,336      Print
03,337      Print "Press ESC... for vertical scroll use Up/Down, MouseLeftButton/MouseRightButton or Mouse Wheel.";
03,338      _Display
03,339      $If WINDOWS Then
03,340          Play "v20120g"
03,341      $End If
03,342      LinesScroled = 0: LinesScroledMAX = pgdlines - SCROLLABLEheight
03,343      Do While InKey$ <> Chr$(27)
03,344          If pgdlines > SCROLLABLEheight Then
03,345              AsIfItIsINKEY% = _MouseInput '      Check the mouse status
03,346              buttondown1 = _MouseButton(1)
03,347              buttondown2 = _MouseButton(2)
03,348              buttondown3 = _MouseButton(3)
03,349              mwheel = _MouseWheel
03,350              If buttondown2 Or mwheel = 1 Or _KeyDown(DOWNkey&) Then ' as if Down
03,351                  If LinesScroled < LinesScroledMAX Then
03,352                      LinesScroled = LinesScroled + 1
03,353                  End If
03,354                  For i = 1 To MIN(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,355                      Locate i + 3, 1: Print CROPorPADatRIGHT$(xgrams$(i + LinesScroled), XdimCOL);
03,356                  Next
03,357              End If
03,358              If buttondown1 Or mwheel = -1 Or _KeyDown(UPkey&) Then ' as if Up
03,359                  If LinesScroled > 0 Then
03,360                      LinesScroled = LinesScroled - 1
03,361                  End If
03,362                  For i = 1 To MIN(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,363                      Locate i + 3, 1: Print CROPorPADatRIGHT$(xgrams$(i + LinesScroled), XdimCOL);
03,364                  Next
03,365              End If
03,366          End If
03,367          _Limit 500
03,368          _Display

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,369      Loop
03,370      While _MouseInput: Wend
03,371      UpdateWindowFrame NormalFRGr, NormalBCKGr
03,372      End If
03,373      End If
03,374
03,375      If IsENTERreleased_Flag Then ' Wrap the current/INVERSE line, and scroll it up/down as the PAGODA
03,376      Wwidth% = XdimCOL
03,377      UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1
03,378      Locate cry, crx, 0, CursorS, CursorE 'hide the cursor
03,379      _Display
03,380      If Len(FileArrayFULL$(CurrentLine)) > XdimCOL Then
03,381      UpdateNextToCCL_BUSY (MostRightField): _Display
03,382      l$ = FileArrayFULL$(CurrentLine)
03,383      ExpandTabsFULL l$
03,384      ' [[[
03,385      ' Firstly do the wrapping virtually (in order to avoid writing some wrapped chunks and encounter unwrappable chunk) - we need either a wrapped line (in its entirety) or none:
03,386      AssumeLineIsWrappable = 1
03,387      pgdlines = 0
03,388      lX$ = l$
03,389      Do While Len(lX$) > Wwidth%
03,390      Glupak% = Wwidth%
03,391      Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak%
+ 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "
03,392      InvokeOnce$ = Mid$(lX$, Glupak% + 1, 1)
03,393      If InStr(" \/_;.-!+%=:", InvokeOnce$) And Mid$(lX$, Glupak%, 1) <> " " Then Exit Do
03,394      'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR InvokeOnce$ = "!" OR
InvokeOnce$ = "+" OR InvokeOnce$ = "%") AND MID$(lX$, Glupak%, 1) <> " " THEN EXIT DO
03,395      Glupak% = Glupak% + 1
03,396      If Glupak% = 0 Then
03,397      AssumeLineIsWrappable = 0
03,398      pgdlines = 0
03,399      GoTo B4TxpanarVIRTUAL2
03,400      End If
03,401      Loop
03,402      lX$ = LTrim$(Mid$(lX$, Glupak% + 1, Len(lX$) - (Glupak%)))
03,403      pgdlines = pgdlines + 1
03,404      Loop
03,405      pgdlines = pgdlines + 1 ' don't forget the last chunk/remainder
03,406      ReDim xgrams$(1 To pgdlines)
03,407
03,408      B4TxpanarVIRTUAL2:
03,409      pgdlines = 0
03,410      If AssumeLineIsWrappable = 1 Then
03,411      lX$ = l$
03,412      Do While Len(lX$) > Wwidth%
03,413      Glupak% = Wwidth%
03,414      Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak% + 1, 1) = "," OR MID$(lX$,
Glupak% + 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,415      InvokeOnce$ = Mid$(1X$, Glupak% + 1, 1)
03,416      If InStr(" \/_;.-!+%=:", InvokeOnce$) And Mid$(1X$, Glupak%, 1) <> " " Then Exit Do
03,417      'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = ";" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR InvokeOnce$ = "!" OR
InvokeOnce$ = "+" OR InvokeOnce$ = "%") AND MID$(1X$, Glupak%, 1) <> " " THEN EXIT DO
03,418      Glupak% = Glupak% - 1
03,419      If Glupak% = 0 Then
03,420          GoTo B4Txpanar2
03,421      End If
03,422      Loop
03,423      pgdlines = pgdlines + 1
03,424      xgrams$(pgdlines) = Left$(1X$, Glupak%) 'PRINT #3, LEFT$(1X$, Glupak%)
03,425      1X$ = LTrim$(Mid$(1X$, Glupak% + 1, Len(1X$) - (Glupak%)))
03,426      Loop
03,427      pgdlines = pgdlines + 1 ' don't forget the last chunk/remainder
03,428      xgrams$(pgdlines) = 1X$ 'PRINT #3, 1X$
03,429      B4Txpanar2:
03,430      Else 'unwrappable
03,431          $If WINDOWS Then
03,432              Play "v20160aba"
03,433          $End If
03,434      End If
03,435
03,436      ']]]
03,437      UpdateNextToCCL_DONE (MostRightField): _Display
03,438      While _MouseInput: Wend
03,439
03,440      If pgdlines Then
03,441          Locate 1, 1, 0, CursorS, CursorE
03,442          For i = 1 To YdimROW
03,443              Locate i, 1: Print Space$(XdimCOL);
03,444          Next
03,445
03,446          SCROLLABLEheight = (YdimROW - 2) - 3 + 1
03,447
03,448          Locate 1, 1, 0
03,449          Print "Original Current/INVERSE line size: "; AddCommas$(Len(FileArrayFULL$(CurrentLine))); " bytes ";
03,450
03,451          For i = 1 To MIN$(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,452              Locate i + 2, 1: Print Left$(xgrams$(i), XdimCOL);
03,453          Next
03,454
03,455          Print
03,456          Print
03,457          Print "Release Enter (or press Esc)... for vertical scroll use Up/Down, MouseLeftButton/MouseRightButton or Mouse Wheel.";
03,458          _Display
03,459
03,460          $If WINDOWS Then
03,461              Play "v20120g"
03,462          $End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,463      LinesScrolled = 0: LinesScrolledMAX = pgdlines - SCROLLABLEheight
03,464      Do While InKey$ <> Chr$(27) And IsENTERreleased = 0
03,465          If pgdlines > SCROLLABLEheight Then
03,466              AsIfItIsINKEY% = _MouseInput '      Check the mouse status
03,467              buttondown1 = _MouseButton(1)
03,468              buttondown2 = _MouseButton(2)
03,469              buttondown3 = _MouseButton(3)
03,470              mwheel = _MouseWheel
03,471              If buttondown2 Or mwheel = 1 Or _KeyDown(DOWNkey&) Then ' as if Down
03,472                  If LinesScrolled < LinesScrolledMAX Then
03,473                      LinesScrolled = LinesScrolled + 1
03,474                      End If
03,475                      For i = 1 To MIN(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,476                          Locate i + 2, 1: Print CROPorPADatRIGHT$(xgrams$(i + LinesScrolled), XdimCOL);
03,477                      Next
03,478                  End If
03,479              If buttondown1 Or mwheel = -1 Or _KeyDown(UPkey&) Then ' as if Up
03,480                  If LinesScrolled > 0 Then
03,481                      LinesScrolled = LinesScrolled - 1
03,482                      End If
03,483                      For i = 1 To MIN(SCROLLABLEheight, pgdlines) ' -2 due to PRINT: PRINT "Press ESC..."
03,484                          Locate i + 2, 1: Print CROPorPADatRIGHT$(xgrams$(i + LinesScrolled), XdimCOL);
03,485                      Next
03,486                  End If
03,487              End If
03,488              _Limit 50
03,489              _Display
03,490          Loop
03,491          While _MouseInput: Wend
03,492          UpdateWindowFrame NormalFRGr, NormalBCKGr
03,493      End If 'IF pgdlines THEN
03,494  End If
03,495      _KeyClear 2 'in order to get rid of all eventual 'Enter' hits during above scrolling
03,496  End If 'IF IsENTERreleased_Flag THEN
03,497
03,498  'DO WHILE INKEY$ <> "": LOOP ' have to clear the keyboard buffer
03,499  '_Limit 500 '_LIMIT 30 'commented because the wheel up/down was not working?!
03,500  'Caution: Above line works fine with "mainstream" mouse like a cheap HP 3-button one, but when attaching "game" mouse as White Shark Lancelot, it became superlaggy and unresponsive, maybe due to REPORT RATE aka POLL
RATE being high, had to set it to 125Hz in order to work as if "cheap".
03,501
03,502  'k$ = InKey$
03,503  'If k$ <> "" Then LastHit~&& = EpochTime~&&: Stepping = 10
03,504  Select Case EpochTime~&& - LastHit~&&
03,505      Case Is > 13 * 60: Stepping = 0 'the 13th minute is maximum idleness
03,506      Case Is > 9 * 60: Stepping = 1
03,507      Case Is > 8 * 60: Stepping = 2
03,508      Case Is > 7 * 60: Stepping = 3
03,509      Case Is > 6 * 60: Stepping = 4
03,510      Case Is > 5 * 60: Stepping = 5

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,511      Case Is > 4 * 60: Stepping = 6
03,512      Case Is > 3 * 60: Stepping = 7
03,513      Case Is > 2 * 60: Stepping = 8
03,514      Case Is > 1 * 60: Stepping = 9
03,515      Case Else
03,516      End Select
03,517      _Limit Int(FramesPerSecond * (Stepping * .1)) + 3 'to avoid 0
03,518
03,519 Loop 'mainloop ]
03,520
03,521 End
03,522
03,523 SetColorScheme:
03,524 'Colorscale .000..255:
03,525
03,526 '0  Black      0      0      0
03,527 '1  Dark Blue  0      0     168
03,528 '2  Dark Green 0     168    0
03,529 '3  Dark Cyan  0     168   168
03,530 '4  Dark Red   168    0      0
03,531 '5  Dark Magenta 168    0     168
03,532 '6  Dark Yellow 168    84      0
03,533 '7  Light Grey 168    168   168
03,534
03,535 '8+0 Dark Grey   84     84     84
03,536 '8+1 Blue      84     84    252
03,537 '8+2 Green     84    252     84
03,538 '8+3 Cyan      84    252   252
03,539 '8+4 Red       252    84     84
03,540 '8+5 Magenta   252    84    252
03,541 '8+6 Yellow    252    252     84
03,542 '8+7 White     252    252   252
03,543
03,544 'Colorscale 00..63:
03,545 '0  Black      0      0      0
03,546 '1  Dark Blue  0      0     42
03,547 '2  Dark Green 0     42     0
03,548 '3  Dark Cyan  0     42    42
03,549 '4  Dark Red   42     0      0
03,550 '5  Dark Magenta 42     0     42
03,551 '6  Dark Yellow 42     21     0
03,552 '7  Light Grey 42     42    42
03,553
03,554 '8+0 Dark Grey   21     21    21
03,555 '8+1 Blue      21     21    63
03,556 '8+2 Green     21     63    21
03,557 '8+3 Cyan      21     63    63
03,558 '8+4 Red       63     21    21
03,559 '8+5 Magenta   63     21    63

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,560 '8+6 Yellow      63    63    21
03,561 '8+7 White      63    63    63
03,562
03,563 'Colorscale
03,564 _PaletteColor 0, _RGB32(0, 0, 0)
03,565 _PaletteColor 1, _RGB32(0, 0, 168)
03,566 _PaletteColor 2, _RGB32(0, 168, 0)
03,567 _PaletteColor 3, _RGB32(0, 168, 168)
03,568 _PaletteColor 4, _RGB32(168, 0, 0)
03,569 _PaletteColor 5, _RGB32(168, 0, 168)
03,570 _PaletteColor 6, _RGB32(168, 84, 0)
03,571 _PaletteColor 7, _RGB32(168, 168, 168)
03,572
03,573 _PaletteColor 8, _RGB32(84, 84, 84)
03,574 _PaletteColor 9, _RGB32(84, 84, 252)
03,575 _PaletteColor 10, _RGB32(84, 252, 84)
03,576 _PaletteColor 11, _RGB32(84, 252, 252)
03,577 _PaletteColor 12, _RGB32(252, 84, 84)
03,578 _PaletteColor 13, _RGB32(252, 84, 252)
03,579 _PaletteColor 14, _RGB32(252, 252, 84)
03,580 _PaletteColor 15, _RGB32(252, 252, 252)
03,581
03,582 _PaletteColor 3, _RGB32(0, 168 - 10, 168 - 10) 'darker Cyan
03,583 _PaletteColor 8, _RGB32(84 - 22, 84 - 22, 84 - 22) 'make Grey darker
03,584
03,585 _PaletteColor 4, _RGB32(110, 2, 25) 'Carminish
03,586 _PaletteColor 4 + 8, _RGB32(110 + 44, 2 + 44, 25 + 44) 'make Carmin more lighter
03,587
03,588 _PaletteColor 1, _RGB32(168 - 20, 84 - 20, 0) 'make the Amber more darker
03,589 _PaletteColor 6 + 8, _RGB32(168 + 54, 84 + 54, 0) 'make the Amber more lighter
03,590
03,591 ' Change the Purple to SteelBlue
03,592 _PaletteColor 5, _RGB32(33, 99, 122)
03,593 _PaletteColor 5 + 8, _RGB32(33 + 44, 99 + 44, 122 + 44)
03,594 ' Change the Green to Cyanish
03,595 _PaletteColor 2, _RGB32(1, 80, 80)
03,596 _PaletteColor 2 + 8, _RGB32(1 + 44, 80 + 44, 80 + 44)
03,597
03,598 Return
03,599
03,600 UpdateColors:
03,601 For ii = 0 To 15 ' Choosing not to alter BLACK color ... maybe channels with 000 also...
03,602     NewR% = Asc(Mid$(RGB3x16$, 3 * ii + 1, 1))
03,603     NewG% = Asc(Mid$(RGB3x16$, 3 * ii + 2, 1))
03,604     NewB% = Asc(Mid$(RGB3x16$, 3 * ii + 3, 1))
03,605     If NewR% Then
03,606         If NewR% + rChannel% + Brightness% <= 0 Then
03,607             NewR% = 1
03,608         ElseIf NewR% + rChannel% + Brightness% > 63 Then

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,609         NewR% = 63
03,610     Else
03,611         NewR% = NewR% + rChannel% + Brightness%
03,612     End If
03,613 End If
03,614 If NewG% Then
03,615     If NewG% + gChannel% + Brightness% <= 0 Then
03,616         NewG% = 1
03,617     ElseIf NewG% + gChannel% + Brightness% > 63 Then
03,618         NewG% = 63
03,619     Else
03,620         NewG% = NewG% + gChannel% + Brightness%
03,621     End If
03,622 End If
03,623 If NewB% Then
03,624     If NewB% + bChannel% + Brightness% <= 0 Then
03,625         NewB% = 1
03,626     ElseIf NewB% + bChannel% + Brightness% > 63 Then
03,627         NewB% = 63
03,628     Else
03,629         NewB% = NewB% + bChannel% + Brightness%
03,630     End If
03,631 End If
03,632 _PaletteColor ii, _RGB32(NewR% * 4, NewG% * 4, NewB% * 4)
03,633 Next
03,634
03,635 'SimulateBlinking% = Not SimulateBlinking%
03,636 'If SimulateBlinking% = 0 Then
03,637 Locate 38, 2: Color 0, 7: Print "Note1: Press 'r|g|b' or 'R|G|B' keys to decrease/increase the respective channel by one point, i|I for Brightness, Esc.";
03,638 Locate 39, 2: Color 0, 7: Print "Note2: In order to have your own color scheme, you can manually change the values of 'Masakari.RGB' (3x16 bytes long).";
03,639 Locate 40, 2: Color 0, 7: Print Using "Note3: WYSIWYG, 'Masakari.RGB' has been updated with rChannel%/gChannel%/bChannel%/Brightness% = ####/###/####/####"; rChannel%, gChannel%, bChannel%, Brightness%;
03,640 'Else
03,641 'Locate 38, 2: Color 15, 7: Print "Note1: Press 'r|g|b' or 'R|G|B' keys to respectively decrease/increase the respective channel by one point.";
03,642 'Locate 39, 2: Color 15, 7: Print "Note2: In order to have your own color scheme, you can manually change the values of 'Masakari.RGB' (3x16 bytes long).";
03,643 'End If
03,644 RGB3x16new$ = Space$(3 * 16)
03,645 For ii = 0 To 15
03,646     Out &H3C7, ii 'set color attribute to read
03,647     red = Inp(&H3C9) ' * 4 'multiply by 4 to convert intensity to 0 to 255 RGB values
03,648     grn = Inp(&H3C9) ' * 4
03,649     blu = Inp(&H3C9) ' * 4
03,650     Mid$(RGB3x16new$, 3 * ii + 1, 1) = Chr$(red)
03,651     Mid$(RGB3x16new$, 3 * ii + 2, 1) = Chr$(grn)
03,652     Mid$(RGB3x16new$, 3 * ii + 3, 1) = Chr$(blu)
03,653 Next
03,654 For i = 1 To 6
03,655     Dummy$ = DrawBoxShadowDUMMY$(i, 2 + (i - 1) * 6, 2, 6 + (i - 1) * 6, 116, "Dummy window with background color" + Str$(i) + " = " + Hex2$(Asc(Mid$(RGB3x16new$, 3 * i + 1, 1))) + Hex2$(Asc(Mid$(RGB3x16new$, 3 * i + 2, 1))) + Hex2$(Asc(Mid$(RGB3x16new$, 3 * i + 3, 1))) + ", +8 = " + Hex2$(Asc(Mid$(RGB3x16new$, 3 * (i + 8) + 1, 1))) + Hex2$(Asc(Mid$(RGB3x16new$, 3 * (i + 8) + 2, 1))) + Hex2$(Asc(Mid$(RGB3x16new$, 3 * (i + 8) + 3, 1))))
03,656 Next

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,657 GoSub UpdateColorSchemeFile
03,658
03,659 Return
03,660
03,661 UpdateColorSchemeFile:
03,662 f00 = FreeFile
03,663 Open "Masakari.RGB" For Binary As #f00
03,664 RGB3x16new$ = Space$(3 * 16)
03,665 For ii = 0 To 15
03,666     Out &H3C7, ii 'set color attribute to read
03,667     red = Inp(&H3C9) '* 4 'multiply by 4 to convert intensity to 0 to 255 RGB values
03,668     grn = Inp(&H3C9) '* 4
03,669     blu = Inp(&H3C9) '* 4
03,670     Mid$(RGB3x16new$, 3 * ii + 1, 1) = Chr$(red)
03,671     Mid$(RGB3x16new$, 3 * ii + 2, 1) = Chr$(grn)
03,672     Mid$(RGB3x16new$, 3 * ii + 3, 1) = Chr$(blu)
03,673 Next
03,674 Put #f00, , RGB3x16new$
03,675 Close #f00
03,676 Return
03,677
03,678 Sub ReturnCOMBO
03,679     Shared LSHIFT_RSHIFT
03,680     Shared LCTRL_RCTRL
03,681     Shared LALT_RALT
03,682
03,683     Shared LALT_HOME
03,684     Shared LALT_END
03,685
03,686     Shared LALT_INS
03,687     Shared LALT_DEL
03,688
03,689     Shared LALT_PGUP
03,690     Shared LALT_PGDN
03,691
03,692     Shared LALT_Left
03,693     Shared LALT_Right
03,694     Shared LALT_Up
03,695     Shared LALT_Down
03,696
03,697     Shared RALT_HOME
03,698     Shared RALT_END
03,699
03,700     Shared RALT_INS
03,701     Shared RALT_DEL
03,702
03,703     Shared RALT_PGUP
03,704     Shared RALT_PGDN
03,705

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

03,706 Shared RALT\_Left  
03,707 Shared RALT\_Right  
03,708 Shared RALT\_Up  
03,709 Shared RALT\_Down  
03,710  
03,711 Shared LSHIFT\_HOME  
03,712 Shared LSHIFT\_END  
03,713  
03,714 Shared LSHIFT\_INS  
03,715 Shared LSHIFT\_DEL  
03,716  
03,717 Shared LSHIFT\_PGUP  
03,718 Shared LSHIFT\_PGDN  
03,719  
03,720 Shared LSHIFT\_Left  
03,721 Shared LSHIFT\_Right  
03,722 Shared LSHIFT\_Up  
03,723 Shared LSHIFT\_Down  
03,724  
03,725 Shared RSHIFT\_HOME  
03,726 Shared RSHIFT\_END  
03,727  
03,728 Shared RSHIFT\_INS  
03,729 Shared RSHIFT\_DEL  
03,730  
03,731 Shared RSHIFT\_PGUP  
03,732 Shared RSHIFT\_PGDN  
03,733  
03,734 Shared RSHIFT\_Left  
03,735 Shared RSHIFT\_Right  
03,736 Shared RSHIFT\_Up  
03,737 Shared RSHIFT\_Down  
03,738  
03,739 Shared LCTRL\_HOME  
03,740 Shared LCTRL\_END  
03,741  
03,742 Shared LCTRL\_INS  
03,743 Shared LCTRL\_DEL  
03,744  
03,745 Shared LCTRL\_PGUP  
03,746 Shared LCTRL\_PGDN  
03,747  
03,748 Shared LCTRL\_Left  
03,749 Shared LCTRL\_Right  
03,750 Shared LCTRL\_Up  
03,751 Shared LCTRL\_Down  
03,752  
03,753 Shared RCTRL\_HOME  
03,754 Shared RCTRL\_END

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,755
03,756 Shared RCTRL_INS
03,757 Shared RCTRL_DEL
03,758
03,759 Shared RCTRL_PGUP
03,760 Shared RCTRL_PGDN
03,761
03,762 Shared RCTRL_Left
03,763 Shared RCTRL_Right
03,764 Shared RCTRL_Up
03,765 Shared RCTRL_Down
03,766
03,767 Shared LSHIFT_LCTRL_HOME
03,768 Shared LSHIFT_LCTRL_END
03,769
03,770 Shared LSHIFT_LCTRL_INS
03,771 Shared LSHIFT_LCTRL_DEL
03,772
03,773 Shared LSHIFT_LCTRL_PGUP
03,774 Shared LSHIFT_LCTRL_PGDN
03,775
03,776 Shared LSHIFT_LCTRL_Left
03,777 Shared LSHIFT_LCTRL_Right
03,778 Shared LSHIFT_LCTRL_Up
03,779 Shared LSHIFT_LCTRL_Down
03,780
03,781 Shared LSHIFT_BackSpace
03,782 Shared LSHIFT_TAB
03,783 Shared LSHIFT_SPACE
03,784 Shared LSHIFT_ESC
03,785 Shared LSHIFT_ENTER
03,786
03,787 Shared RSHIFT_BackSpace
03,788 Shared RSHIFT_TAB
03,789 Shared RSHIFT_SPACE
03,790 Shared RSHIFT_ESC
03,791 Shared RSHIFT_ENTER
03,792
03,793 Shared LCTRL_SPACE
03,794 Shared LCTRL_ENTER
03,795
03,796 Shared RCTRL_SPACE
03,797 Shared RCTRL_ENTER
03,798
03,799 If _KeyDown(LSHIFTkey&) And _KeyDown(RSHIFTkey&) Then LSHIFT_RSHIFT = 1 Else LSHIFT_RSHIFT = 0
03,800 If _KeyDown(LCTRLkey&) And _KeyDown(RCTRLkey&) Then LCTRL_RCTRL = 1 Else LCTRL_RCTRL = 0
03,801 If _KeyDown(LALTkey&) And _KeyDown(RALTkey&) Then LALT_RALT = 1 Else LALT_RALT = 0
03,802
03,803 'LALT:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,804 If _KeyDown(LALTkey&) And _KeyDown(HOMEkey&) Then LALT_HOME = 1 Else LALT_HOME = 0
03,805 If _KeyDown(LALTkey&) And _KeyDown(ENDkey&) Then LALT_END = 1 Else LALT_END = 0
03,806
03,807 If _KeyDown(LALTkey&) And _KeyDown(INSkey&) Then LALT_INS = 1 Else LALT_INS = 0
03,808 If _KeyDown(LALTkey&) And _KeyDown(DELkey&) Then LALT_DEL = 1 Else LALT_DEL = 0
03,809
03,810 If _KeyDown(LALTkey&) And _KeyDown(PGUPkey&) Then LALT_PGUP = 1 Else LALT_PGUP = 0
03,811 If _KeyDown(LALTkey&) And _KeyDown(PGDNkey&) Then LALT_PGDN = 1 Else LALT_PGDN = 0
03,812
03,813 If _KeyDown(LALTkey&) And _KeyDown(LEFTkey&) Then LALT_Left = 1 Else LALT_Left = 0
03,814 If _KeyDown(LALTkey&) And _KeyDown(RIGHTkey&) Then LALT_Right = 1 Else LALT_Right = 0
03,815 If _KeyDown(LALTkey&) And _KeyDown(UPkey&) Then LALT_Up = 1 Else LALT_Up = 0
03,816 If _KeyDown(LALTkey&) And _KeyDown(DOWNkey&) Then LALT_Down = 1 Else LALT_Down = 0
03,817
03,818 'RALT:
03,819 If _KeyDown(RALTkey&) And _KeyDown(HOMEkey&) Then RALT_HOME = 1 Else RALT_HOME = 0
03,820 If _KeyDown(RALTkey&) And _KeyDown(ENDkey&) Then RALT_END = 1 Else RALT_END = 0
03,821
03,822 If _KeyDown(RALTkey&) And _KeyDown(INSkey&) Then RALT_INS = 1 Else RALT_INS = 0
03,823 If _KeyDown(RALTkey&) And _KeyDown(DELkey&) Then RALT_DEL = 1 Else RALT_DEL = 0
03,824
03,825 If _KeyDown(RALTkey&) And _KeyDown(PGUPkey&) Then RALT_PGUP = 1 Else RALT_PGUP = 0
03,826 If _KeyDown(RALTkey&) And _KeyDown(PGDNkey&) Then RALT_PGDN = 1 Else RALT_PGDN = 0
03,827
03,828 If _KeyDown(RALTkey&) And _KeyDown(LEFTkey&) Then Print RALT_Left = 1 Else RALT_Left = 0
03,829 If _KeyDown(RALTkey&) And _KeyDown(RIGHTkey&) Then RALT_Right = 1 Else RALT_Right = 0
03,830 If _KeyDown(RALTkey&) And _KeyDown(UPkey&) Then RALT_Up = 1 Else RALT_Up = 0
03,831 If _KeyDown(RALTkey&) And _KeyDown(DOWNkey&) Then RALT_Down = 1 Else RALT_Down = 0
03,832 'LSHIFT:
03,833 If _KeyDown(LSHIFTkey&) And _KeyDown(HOMEkey&) Then LSHIFT_HOME = 1 Else LSHIFT_HOME = 0
03,834 If _KeyDown(LSHIFTkey&) And _KeyDown(ENDkey&) Then LSHIFT_END = 1 Else LSHIFT_END = 0
03,835
03,836 If _KeyDown(LSHIFTkey&) And _KeyDown(INSkey&) Then LSHIFT_INS = 1 Else LSHIFT_INS = 0
03,837 If _KeyDown(LSHIFTkey&) And _KeyDown(DELkey&) Then LSHIFT_DEL = 1 Else LSHIFT_DEL = 0
03,838
03,839 If _KeyDown(LSHIFTkey&) And _KeyDown(PGUPkey&) Then LSHIFT_PGUP = 1 Else LSHIFT_PGUP = 0
03,840 If _KeyDown(LSHIFTkey&) And _KeyDown(PGDNkey&) Then LSHIFT_PGDN = 1 Else LSHIFT_PGDN = 0
03,841
03,842 If _KeyDown(LSHIFTkey&) And _KeyDown(LEFTkey&) Then LSHIFT_Left = 1 Else LSHIFT_Left = 0
03,843 If _KeyDown(LSHIFTkey&) And _KeyDown(RIGHTkey&) Then LSHIFT_Right = 1 Else LSHIFT_Right = 0
03,844 If _KeyDown(LSHIFTkey&) And _KeyDown(UPkey&) Then LSHIFT_Up = 1 Else LSHIFT_Up = 0
03,845 If _KeyDown(LSHIFTkey&) And _KeyDown(DOWNkey&) Then LSHIFT_Down = 1 Else LSHIFT_Down = 0
03,846
03,847 'RSHIFT:
03,848 If _KeyDown(RSHIFTkey&) And _KeyDown(HOMEkey&) Then RSHIFT_HOME = 1 Else RSHIFT_HOME = 0
03,849 If _KeyDown(RSHIFTkey&) And _KeyDown(ENDkey&) Then RSHIFT_END = 1 Else RSHIFT_END = 0
03,850
03,851 If _KeyDown(RSHIFTkey&) And _KeyDown(INSkey&) Then RSHIFT_INS = 1 Else RSHIFT_INS = 0
03,852 If _KeyDown(RSHIFTkey&) And _KeyDown(DELkey&) Then RSHIFT_DEL = 1 Else RSHIFT_DEL = 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

03,853
03,854 If _KeyDown(RSHIFTkey&) And _KeyDown(PGUPkey&) Then RSHIFT_PGUP = 1 Else RSHIFT_PGUP = 0
03,855 If _KeyDown(RSHIFTkey&) And _KeyDown(PGDNkey&) Then RSHIFT_PGDN = 1 Else RSHIFT_PGDN = 0
03,856
03,857 If _KeyDown(RSHIFTkey&) And _KeyDown(LEFTkey&) Then RSHIFT_Left = 1 Else RSHIFT_Left = 0
03,858 If _KeyDown(RSHIFTkey&) And _KeyDown(RIGHTkey&) Then RSHIFT_Right = 1 Else RSHIFT_Right = 0
03,859 If _KeyDown(RSHIFTkey&) And _KeyDown(UPkey&) Then RSHIFT_Up = 1 Else RSHIFT_Up = 0
03,860 If _KeyDown(RSHIFTkey&) And _KeyDown(DOWNkey&) Then RSHIFT_Down = 1 Else RSHIFT_Down = 0
03,861
03,862 'LCTRL:
03,863 If _KeyDown(LCTRLkey&) And _KeyDown(HOMEkey&) Then LCTRL_HOME = 1 Else LCTRL_HOME = 0
03,864 If _KeyDown(LCTRLkey&) And _KeyDown(ENDkey&) Then LCTRL_END = 1 Else LCTRL_END = 0
03,865
03,866 If _KeyDown(LCTRLkey&) And _KeyDown(INSkey&) Then LCTRL_INS = 1 Else LCTRL_INS = 0
03,867 If _KeyDown(LCTRLkey&) And _KeyDown(DELkey&) Then LCTRL_DEL = 1 Else LCTRL_DEL = 0
03,868
03,869 If _KeyDown(LCTRLkey&) And _KeyDown(PGUPkey&) Then LCTRL_PGUP = 1 Else LCTRL_PGUP = 0
03,870 If _KeyDown(LCTRLkey&) And _KeyDown(PGDNkey&) Then LCTRL_PGDN = 1 Else LCTRL_PGDN = 0
03,871
03,872 If _KeyDown(LCTRLkey&) And _KeyDown(LEFTkey&) Then LCTRL_Left = 1 Else LCTRL_Left = 0
03,873 If _KeyDown(LCTRLkey&) And _KeyDown(RIGHTkey&) Then LCTRL_Right = 1 Else LCTRL_Right = 0
03,874 If _KeyDown(LCTRLkey&) And _KeyDown(UPkey&) Then LCTRL_Up = 1 Else LCTRL_Up = 0
03,875 If _KeyDown(LCTRLkey&) And _KeyDown(DOWNkey&) Then LCTRL_Down = 1 Else LCTRL_Down = 0
03,876
03,877 'RCTRL:
03,878 If _KeyDown(RCTRLkey&) And _KeyDown(HOMEkey&) Then RCTRL_HOME = 1 Else RCTRL_HOME = 0
03,879 If _KeyDown(RCTRLkey&) And _KeyDown(ENDkey&) Then RCTRL_END = 1 Else RCTRL_END = 0
03,880
03,881 If _KeyDown(RCTRLkey&) And _KeyDown(INSkey&) Then RCTRL_INS = 1 Else RCTRL_INS = 0
03,882 If _KeyDown(RCTRLkey&) And _KeyDown(DELkey&) Then RCTRL_DEL = 1 Else RCTRL_DEL = 0
03,883
03,884 If _KeyDown(RCTRLkey&) And _KeyDown(PGUPkey&) Then RCTRL_PGUP = 1 Else RCTRL_PGUP = 0
03,885 If _KeyDown(RCTRLkey&) And _KeyDown(PGDNkey&) Then RCTRL_PGDN = 1 Else RCTRL_PGDN = 0
03,886
03,887 If _KeyDown(RCTRLkey&) And _KeyDown(LEFTkey&) Then RCTRL_Left = 1 Else RCTRL_Left = 0
03,888 If _KeyDown(RCTRLkey&) And _KeyDown(RIGHTkey&) Then RCTRL_Right = 1 Else RCTRL_Right = 0
03,889 If _KeyDown(RCTRLkey&) And _KeyDown(UPkey&) Then RCTRL_Up = 1 Else RCTRL_Up = 0
03,890 If _KeyDown(RCTRLkey&) And _KeyDown(DOWNkey&) Then RCTRL_Down = 1 Else RCTRL_Down = 0
03,891
03,892 'LSHIFT+LCTRL: NOTICE: LSHIFT+LCTRL+Left triggers 3 variables on - 1] LSHIFT_LCTRL_Left 2] LSHIFT_Left 3] LCTRL_Left
03,893 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(HOMEkey&) Then LSHIFT_LCTRL_HOME = 1 Else LSHIFT_LCTRL_HOME = 0
03,894 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(ENDkey&) Then LSHIFT_LCTRL_END = 1 Else LSHIFT_LCTRL_END = 0
03,895
03,896 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(INSkey&) Then LSHIFT_LCTRL_INS = 1 Else LSHIFT_LCTRL_INS = 0
03,897 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(DELkey&) Then LSHIFT_LCTRL_DEL = 1 Else LSHIFT_LCTRL_DEL = 0
03,898
03,899 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(PGUPkey&) Then LSHIFT_LCTRL_PGUP = 1 Else LSHIFT_LCTRL_PGUP = 0
03,900 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(PGDNkey&) Then LSHIFT_LCTRL_PGDN = 1 Else LSHIFT_LCTRL_PGDN = 0
03,901

```

```

03,902 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(LEFTkey&) Then LSHIFT_LCTRL_Left = 1 Else LSHIFT_LCTRL_Left = 0
03,903 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(RIGHTkey&) Then LSHIFT_LCTRL_Right = 1 Else LSHIFT_LCTRL_Right = 0
03,904 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(UPkey&) Then LSHIFT_LCTRL_Up = 1 Else LSHIFT_LCTRL_Up = 0
03,905 If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(DOWNkey&) Then LSHIFT_LCTRL_Down = 1 Else LSHIFT_LCTRL_Down = 0
03,906
03,907 'LSHIFT:
03,908 If _KeyDown(LSHIFTkey&) And _KeyDown(BACKSPCkey&) Then LSHIFT_BackSpace = 1 Else LSHIFT_BackSpace = 0
03,909 If _KeyDown(LSHIFTkey&) And _KeyDown(TABkey&) Then LSHIFT_TAB = 1 Else LSHIFT_TAB = 0
03,910 If _KeyDown(LSHIFTkey&) And _KeyDown(SPACEkey&) Then LSHIFT_SPACE = 1 Else LSHIFT_SPACE = 0
03,911 If _KeyDown(LSHIFTkey&) And _KeyDown(ESCkey&) Then LSHIFT_ESC = 1 Else LSHIFT_ESC = 0
03,912 If _KeyDown(LSHIFTkey&) And _KeyDown(ENTERkey&) Then LSHIFT_ENTER = 1 Else LSHIFT_ENTER = 0
03,913 'RSHIFT:
03,914 If _KeyDown(RSHIFTkey&) And _KeyDown(BACKSPCkey&) Then RSHIFT_BackSpace = 1 Else RSHIFT_BackSpace = 0
03,915 If _KeyDown(RSHIFTkey&) And _KeyDown(TABkey&) Then RSHIFT_TAB = 1 Else RSHIFT_TAB = 0
03,916 If _KeyDown(RSHIFTkey&) And _KeyDown(SPACEkey&) Then RSHIFT_SPACE = 1 Else RSHIFT_SPACE = 0
03,917 If _KeyDown(RSHIFTkey&) And _KeyDown(ESCkey&) Then RSHIFT_ESC = 1 Else RSHIFT_ESC = 0
03,918 If _KeyDown(RSHIFTkey&) And _KeyDown(ENTERkey&) Then RSHIFT_ENTER = 1 Else RSHIFT_ENTER = 0
03,919
03,920 'LCTRL:
03,921 If _KeyDown(LCTRLkey&) And _KeyDown(SPACEkey&) Then LCTRL_SPACE = 1 Else LCTRL_SPACE = 0
03,922 If _KeyDown(LCTRLkey&) And _KeyDown(ENTERkey&) Then LCTRL_ENTER = 1 Else LCTRL_ENTER = 0
03,923 'RCTRL:
03,924 If _KeyDown(RCTRLkey&) And _KeyDown(SPACEkey&) Then RCTRL_SPACE = 1 Else RCTRL_SPACE = 0
03,925 If _KeyDown(RCTRLkey&) And _KeyDown(ENTERkey&) Then RCTRL_ENTER = 1 Else RCTRL_ENTER = 0
03,926
03,927 ' _KEYCLEAR
03,928 End Sub
03,929
03,930 Sub ExpandTabsFULL (l$)
03,931 Shared XdimCOL
03,932 If InStr(l$, Chr$(9)) Then
03,933     TabV% = 8
03,934     b$ = "": f% = 0
03,935     PossibleRecalculation = Len(l$)
03,936     'FOR i& = 1 TO MIN(PossibleRecalculation, XdimCOL) 'LEN(l$) ' For some reason going all the length is too slow...?! As it is, the lateral scroll is crippled!
03,937     For i& = 1 To PossibleRecalculation
03,938         RemoveSpecial$ = Mid$(l$, i&, 1)
03,939         If RemoveSpecial$ = Chr$(9) Then
03,940             b$ = b$ + String$((f% \ TabV%) * TabV% + TabV% - f%, " ")
03,941             ' |
03,942             ' |
03,943             ' \|/
03,944             'TabV% - (f% - (f% \ TabV%) * TabV%) =
03,945             'TabV% - (f% MOD TabV%)
03,946             f% = (f% \ TabV%) * TabV% + TabV%
03,947         Else
03,948             b$ = b$ + RemoveSpecial$ 'MID$(l$, i&, 1)
03,949             f% = f% + 1
03,950         End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

03,951     Next
03,952     l$ = b$
03,953 End If
03,954 End Sub
03,955
03,956 Sub ExpandTabs (l$)
03,957     Shared XdimCOL
03,958     If InStr(l$, Chr$(9)) Then
03,959         TabV% = 8
03,960         b$ = "": f% = 0
03,961         PossibleRecalculation = Len(l$)
03,962         For i% = 1 To MIN$(PossibleRecalculation, XdimCOL) 'LEN(l$) ' For some reason going all the length is too slow...?! As it is, the lateral scroll is crippled!
03,963             RemoveSpecial$ = Mid$(l$, i%, 1)
03,964             If RemoveSpecial$ = Chr$(9) Then
03,965                 b$ = b$ + String$((f% \ TabV%) * TabV% + TabV% - f%, " ")
03,966                 ' |
03,967                 ' |
03,968                 ' \|/
03,969                 'TabV% - (f% - (f% \ TabV%) * TabV%) =
03,970                 'TabV% - (f% MOD TabV%)
03,971                 f% = (f% \ TabV%) * TabV% + TabV%
03,972             Else
03,973                 b$ = b$ + RemoveSpecial$ 'MID$(l$, i%, 1)
03,974                 f% = f% + 1
03,975             End If
03,976         Next
03,977         l$ = b$
03,978     End If
03,979 End Sub
03,980
03,981 Function MIN$(YdimROW, filecount)
03,982     If YdimROW < filecount Then MIN$ = YdimROW Else MIN$ = filecount
03,983 End Function
03,984
03,985 Function AddCommas$(numeral)
03,986     s$ = LTrim$(Str$(numeral))
03,987     If Len(s$) > 3 Then
03,988         If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
03,989         s$ = ""
03,990         For i = 1 To Len(x$) Step 3
03,991             s$ = s$ + Mid$(x$, i, 3) + ","
03,992         Next
03,993         s$ = Left$(s$, Len(s$) - 1)
03,994     End If
03,995     AddCommas$ = LTrim$(s$)
03,996 End Function
03,997
03,998 Function AddCommasPaddedNUM$(numeral)
03,999     Shared FieldLineNum

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,000 s$ = LTrim$(Str$(numeral))
04,001 If Len(s$) > 3 Then
04,002   If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
04,003   s$ = ""
04,004   For i = 1 To Len(x$) Step 3
04,005     s$ = s$ + Mid$(x$, i, 3) + ", "
04,006   Next
04,007   s$ = Left$(s$, Len(s$) - 1)
04,008 End If
04,009 s$ = LTrim$(s$)
04,010 Padded$ = Right$("000,000,000,000", FieldLineNum)
04,011 Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
04,012 AddCommasPaddedNUM$ = Padded$
04,013 End Function
04,014
04,015 Function AddCommasPaddedLEN$ (numeral)
04,016   Shared FieldLineLen
04,017   s$ = LTrim$(Str$(numeral))
04,018   If Len(s$) > 3 Then
04,019     If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
04,020     s$ = ""
04,021     For i = 1 To Len(x$) Step 3
04,022       s$ = s$ + Mid$(x$, i, 3) + ", "
04,023     Next
04,024     s$ = Left$(s$, Len(s$) - 1)
04,025   End If
04,026   s$ = LTrim$(s$)
04,027   Padded$ = Right$("000,000,000,000", FieldLineLen)
04,028   Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
04,029   AddCommasPaddedLEN$ = Padded$
04,030 End Function
04,031
04,032 Sub UpdateCLL (1, posit)
04,033   Shared YdimROW ' , FileArray$()
04,034   crx = Pos(0)
04,035   cry = CsrLin
04,036   Locate YdimROW + 1, posit: Color 9, 0
04,037   'PRINT "; Line Number: " + AddCommasPaddedNUM$(1);
04,038   Print "; Line: " + AddCommasPaddedNUM$(1);
04,039   'PRINT "; Line Length: " + AddCommasPaddedLEN$(Len(FileArrayFULL$(1))); 'fix with 'FULL' from jun-02
04,040   Print "; Linesize: " + AddCommasPaddedLEN$(Len(FileArrayFULL$(1))); 'fix with 'FULL' from jun-02
04,041   Locate cry, crx, 1, CursorS, CursorE
04,042 End Sub
04,043
04,044 Sub UpdateNextToCCL_BUSY (posit)
04,045   Shared XdimCOL
04,046   Shared YdimROW
04,047   crx = Pos(0)
04,048   cry = CsrLin

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,049 Locate YdimROW + 1, posit: Color 4, 0: Print Space$((XdimCOL - posit) - 0);
04,050 Locate YdimROW + 1, posit: Color 9, 0: Print "; Status: BUSY";
04,051 Locate cry, crx, 1, CursorS, CursorE
04,052 End Sub
04,053
04,054 Sub UpdateNextToCCL_DONE (posit)
04,055 Shared XdimCOL
04,056 Shared YdimROW
04,057 crx = Pos(0)
04,058 cry = CsrLin
04,059 Locate YdimROW + 1, posit: Color 4, 0: Print Space$((XdimCOL - posit) - 0); 'STRING$((XdimCOL - posit) - 1, " ");
04,060 Locate YdimROW + 1, posit: Color 9, 0: Print "; Status: DONE";
04,061 Locate cry, crx, 1, CursorS, CursorE
04,062 End Sub
04,063
04,064 Sub UpdateNextToCCL_UNWRAPPABLEcount (posit)
04,065 Shared XdimCOL
04,066 Shared YdimROW
04,067 Shared UnwrappableLines
04,068 crx = Pos(0)
04,069 cry = CsrLin
04,070 Locate YdimROW + 1, posit: Color 4, 0: Print Space$((XdimCOL - posit) - 0);
04,071 Locate YdimROW + 1, posit: Color 9, 0: Print "; Unwrappable Lines: " + AddCommas$(UnwrappableLines);
04,072 Locate cry, crx, 1, CursorS, CursorE
04,073 End Sub
04,074
04,075 Sub UpdateCLine (lineToWrite, columnToWrite, FRGR, BACKGR, ln)
04,076 Shared FileArrayWINDOW$(ln)
04,077 Shared XdimCOL
04,078 Shared CurrentWord$
04,079 crx = Pos(0)
04,080 cry = CsrLin
04,081 Color FRGR, BACKGR
04,082 Locate lineToWrite, columnToWrite, 1, CursorS, CursorE
04,083 Print FileArrayWINDOW$(ln);
04,084 'get the word under the cursor [
04,085 LeftM = crx: RightM = crx 'both can go within 1..XdimCOL
04,086 CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM, 1)
04,087 If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,088   Do While LeftM > 1
04,089     CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM - 1, 1)
04,090     If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,091       LeftM = LeftM - 1
04,092     Else
04,093       Exit Do
04,094     End If
04,095   Loop
04,096   Do While RightM < XdimCOL
04,097     CurrentChar$ = Mid$(FileArrayWINDOW$(ln), RightM + 1, 1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,098      If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,099          RightM = RightM + 1
04,100      Else
04,101          Exit Do
04,102      End If
04,103      Loop
04,104  End If
04,105  If RightM - LeftM > 0 Then CurrentWord$ = Mid$(FileArrayWINDOW$(ln), LeftM, RightM - LeftM + 1) Else CurrentWord$ = ""
04,106  'get the word under the cursor ]
04,107  Locate cry, crx, 1, CursorS, CursorE
04,108 End Sub
04,109
04,110 Sub UpdateCLineHIGHLIGHTword (lineToWrite, columnToWrite, FRGR, BACKGR, ln)
04,111     Shared FileArrayWINDOW$()
04,112     Shared XdimCOL
04,113     Shared CurrentWord$
04,114     crx = Pos(0)
04,115     cry = CsrLin
04,116     Color FRGR, BACKGR
04,117     Locate lineToWrite, columnToWrite, 1, CursorS, CursorE
04,118     Print FileArrayWINDOW$(ln);
04,119     'get the word under the cursor [
04,120     LeftM = crx: RightM = crx 'both can go within 1..XdimCOL
04,121     CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM, 1)
04,122     If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,123         Do While LeftM > 1
04,124             CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM - 1, 1)
04,125             If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,126                 LeftM = LeftM - 1
04,127             Else
04,128                 Exit Do
04,129             End If
04,130         Loop
04,131         Do While RightM < XdimCOL
04,132             CurrentChar$ = Mid$(FileArrayWINDOW$(ln), RightM + 1, 1)
04,133             If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
04,134                 RightM = RightM + 1
04,135             Else
04,136                 Exit Do
04,137             End If
04,138         Loop
04,139     End If
04,140     If RightM - LeftM > 0 Then CurrentWord$ = Mid$(FileArrayWINDOW$(ln), LeftM, RightM - LeftM + 1) Else CurrentWord$ = ""
04,141     If CurrentWord$ <> "" Then
04,142         Color 7 + 8, BACKGR
04,143         For qq = LeftM To RightM
04,144             Locate cry, qq, 1, CursorS, CursorE
04,145             Print Mid$(FileArrayWINDOW$(ln), qq, 1);
04,146         Next

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,147 End If
04,148 'get the word under the cursor ]
04,149 Locate cry, crx, 1, CursorS, CursorE
04,150 End Sub
04,151
04,152 Sub UpdateWindowFrame (FRGR, BACKGR)
04,153 Shared YdimROW, filecount, File_Frame_y, File_Frame_x, XdimCOL, FileArrayWINDOW$( ), FileArray$( )
04,154 Color FRGR, BACKGR
04,155 If YdimROW > filecount Then
04,156     For i = 1 To YdimROW
04,157         Locate i, 1: Print Space$(XdimCOL);
04,158     Next
04,159 End If
04,160 For i = 1 To MIN$(YdimROW, filecount)
04,161     DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
04,162     If Len(DumboReadOnceNotThrice$) >= XdimCOL Then
04,163         FileArrayWINDOW$(i) = Mid$(DumboReadOnceNotThrice$, File_Frame_x, XdimCOL)
04,164     Else
04,165         FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + Space$(XdimCOL - Len(DumboReadOnceNotThrice$))
04,166     End If
04,167     'since r.8 the goal is to browse properly binary/.tar files
04,168     'FOR j = 1 TO LEN(FileArrayWINDOW$(i))
04,169     'since r.9 the goal is to browse properly German/Spanish/French/Italian files, SO COMMENTING OUT:
04,170     'IF MID$(FileArrayWINDOW$(i), j, 1) < " " THEN MID$(FileArrayWINDOW$(i), j, 1) = CHR$(32)
04,171     'NEXT
04,172     Locate i, 1: Print FileArrayWINDOW$(i);
04,173 Next
04,174 End Sub
04,175
04,176 Sub ReportTimeToLoad (posit)
04,177 Shared YdimROW, XdimCOL
04,178 Shared TimeA, TimeB
04,179 crx = Pos(0)
04,180 cry = CsrLin
04,181 ElapsedTime% = TimeB - TimeA
04,182 If ElapsedTime% < 0 Then ElapsedTime% = (86400 - TimeA) + TimeB
04,183 Paddedstr$ = "; Loaded in " + LTrim$(Str$(ElapsedTime%)) + " seconds."
04,184 ' P=6
04,185 '[123456] (8-P)-1=2 ' 2021-Jul-26, ugh, the fix is without '-1' because P is not 5 but 6!
04,186 Locate YdimROW + 1, posit: Color 4, 0: Print Space$(XdimCOL - posit) - 0);
04,187 Locate YdimROW + 1, posit: Color 4, 0: Print Paddedstr$;
04,188 Locate cry, crx, 1, CursorS, CursorE
04,189 End Sub
04,190
04,191 Sub ReportTimeToScroll (posit)
04,192 Shared YdimROW, XdimCOL
04,193 Shared TimeScrollA, TimeScrollB
04,194 Shared LineScrollBenchmark
04,195 Shared PageScrollBenchmark

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,196 Shared filecount
04,197   crx = Pos(0)
04,198   cry = CsrLin
04,199   ElapsedTime% = TimeScrollB - TimeScrollA
04,200   If ElapsedTime% < 0 Then ElapsedTime% = (86400 - TimeScrollA) + TimeScrollB
04,201   If ElapsedTime% = 0 Then ElapsedTime% = 1
04,202   'Line-By-Line or Line granularity Scroll Rate; LPS stands for Lines-Per-Second
04,203   'Page-By-Page or Page granularity Scroll Rate; PPS stands for Pages-Per-Second
04,204   Locate YdimROW + 1, posit: Color 4, 0: Print Space$(XdimCOL - posit) - 0);
04,205   If XdimCOL = 213 Then
04,206       If LineScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; DOWNed in"; ElapsedTime%; "seconds, or "; AddCommasPaddedLEN$(filecount / ElapsedTime%); " LPS";
04,207       If PageScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; PGDNed in"; ElapsedTime%; "seconds, or "; AddCommasPaddedLEN$((filecount / YdimROW) / ElapsedTime%); " PPS";
04,208   Else
04,209       If LineScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; DOWNed in"; ElapsedTime%; "seconds.";
04,210       If PageScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; PGDNed in"; ElapsedTime%; "seconds.";
04,211   End If
04,212   Locate cry, crx, 1, Cursor$, CursorE
04,213 End Sub
04,214
04,215 Function FileArrayFULL$(i)
04,216   Shared QWORD, LineLen13
04,217   'SHARED TheWholeFile$
04,218   _MemGet MhandleOFF, MhandleOFF.OFFSET + 8&& * (i - 1), QWORD
04,219   _MemGet MhandleLEN, MhandleLEN.OFFSET + 8&& * (i - 1), LineLen13
04,220   If ToLoadOrNotFlag Then
04,221       'BufferForLine$ = MID$(TheWholeFile$, QWORD, LineLen13)
04,222       BufferForLine$ = Space$(LineLen13)
04,223       _MemGet Mwholefile, Mwholefile.OFFSET + (QWORD - 1), BufferForLine$
04,224   Else
04,225       BufferForLine$ = Space$(LineLen13)
04,226       Seek #1, QWORD
04,227       Get #1, , BufferForLine$
04,228   End If
04,229   'ExpandTabs (BufferForLine$)
04,230   FileArrayFULL$ = BufferForLine$
04,231 End Function
04,232
04,233 Function FileArray$(i)
04,234   Shared QWORD, LineLen13
04,235   'SHARED TheWholeFile$
04,236   _MemGet MhandleOFF, MhandleOFF.OFFSET + 8&& * (i - 1), QWORD
04,237   _MemGet MhandleLEN, MhandleLEN.OFFSET + 8&& * (i - 1), LineLen13
04,238   If ToLoadOrNotFlag Then
04,239       'BufferForLine$ = MID$(TheWholeFile$, QWORD, LineLen13)
04,240       BufferForLine$ = Space$(LineLen13)
04,241       _MemGet Mwholefile, Mwholefile.OFFSET + (QWORD - 1), BufferForLine$
04,242   Else
04,243       BufferForLine$ = Space$(LineLen13)
04,244       Seek #1, QWORD

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,245         Get #1, , BufferForLine$
04,246     End If
04,247     ExpandTabs (BufferForLine$)
04,248     FileArray$ = BufferForLine$
04,249 End Function
04,250
04,251 Sub ShowIndigoWindow
04,252     'COLOR 8, 0: LOCATE i, 1 + LEN(FileArrayWINDOW$(i)): PRINT CHR$(179);: COLOR FGGR, BACKGR
04,253 End Sub
04,254
04,255 'http://rosettacode.org/wiki/Levenshtein_distance#FutureBasic
04,256 'FutureBasic
04,257 'Based on Wikipedia algorithm. Suitable for Pascal strings.
04,258
04,259 'include "ConsoleWindow"
04,260
04,261 'local fn LevenshteinDistance( aStr as Str255, bStr as Str255 ) as long
04,262 'dim as long m, n, i, j, min, k, l
04,263 'dim as long distance( 255, 255 )
04,264
04,265 'm = len(aStr)
04,266 'n = len(bStr)
04,267
04,268 'for i = 0 to m
04,269 '    distance( i, 0 ) = i
04,270 'next
04,271
04,272 'for j = 0 to n
04,273 '    distance( 0, j ) = j
04,274 'next
04,275
04,276 'for j = 1 to n
04,277 '    for i = 1 to m
04,278 '        if mid$( aStr, i, 1 ) == mid$( bStr, j, 1 )
04,279 '            distance( i, j ) = distance( i-1, j-1 )
04,280 '        else
04,281 '            min = distance( i-1, j ) + 1
04,282 '            k   = distance( i, j - 1 ) + 1
04,283 '            l   = distance( i-1, j-1 ) + 1
04,284 '            if k < min then min = k
04,285 '            if l < min then min = l
04,286 '            distance( i, j ) = min
04,287 '        end if
04,288 '    next
04,289 'next
04,290 'end fn = distance( m, n )
04,291
04,292 'dim as long i
04,293 'dim as Str255 testStr( 5, 2 )

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,294
04,295 'testStr( 0, 0 ) = "kitten"      : testStr( 0, 1 ) = "sitting"
04,296 'testStr( 1, 0 ) = "rosettacode" : testStr( 1, 1 ) = "raisethysword"
04,297 'testStr( 2, 0 ) = "Saturday"   : testStr( 2, 1 ) = "Sunday"
04,298 'testStr( 3, 0 ) = "FutureBasic" : testStr( 3, 1 ) = "FutureBasic"
04,299 'testStr( 4, 0 ) = "here's a bunch of words"
04,300 'testStr( 4, 1 ) = "to wring out this code"
04,301
04,302 'for i = 0 to 4
04,303 '   print "1st string = "; testStr( i, 0 )
04,304 '   print "2nd string = "; testStr( i, 1 )
04,305 '   print "Levenshtein distance ="; fn LevenshteinDistance( testStr( i, 0 ), testStr( i, 1 ) )
04,306 '   print
04,307 'next
04,308
04,309 'Output:
04,310
04,311 '1st string = kitten
04,312 '2nd string = sitting
04,313 'Levenshtein distance = 3
04,314
04,315 '1st string = rosettacode
04,316 '2nd string = raisethysword
04,317 'Levenshtein distance = 8
04,318
04,319 '1st string = Saturday
04,320 '2nd string = Sunday
04,321 'Levenshtein distance = 3
04,322
04,323 '1st string = FutureBasic
04,324 '2nd string = FutureBasic
04,325 'Levenshtein distance = 0
04,326
04,327 '1st string = here's a bunch of words
04,328 '2nd string = to wring out this code
04,329 'Levenshtein distance = 18
04,330
04,331 'In information theory and computer science, the Levenshtein distance is a metric for measuring the amount of difference between two sequences (i.e. an edit distance). The Levenshtein distance between two strings is
defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character.
04,332
04,333 Sub DrawBoxShadow3 (TopLrow, TopLcol, BottomRow, BottomRcol, Captmn$)
04,334   Shared FileSize
04,335   Shared LoadedFile$
04,336   Shared PSPLike$
04,337   Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$(), FileArray$()
04,338   'Shadow
04,339   BckGRcolor = 4
04,340   Color 8, BACKGR
04,341   For i = 1 To MIN8(YdimROW, filecount)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,391 Print " | \ / _ \ / / / _ \ | Y \ / _ \ | | \ / _ \"
04,392 Locate , TopLcol + 2
04,393 Print " | _ | _ \ ( _ // _ _ \ ( _ / | _ | / ( _ / | _ | / ( _ / \"
04,394 Locate , TopLcol + 2
04,395 Print " \ \ \ \ \ \ \ \"
04,396 Locate , TopLcol + 2
04,397 Print ""
04,398
04,399 Locate , TopLcol + 2
04,400 Print "A typhoon-class exact & wildcards & Levenshtein Distance (Wagner-Fischer) searcheress"
04,401 Print ""
04,402 Locate , TopLcol + 2
04,403 Print "Note1: The dump (resultant hits) goes to Kazahana.txt file."
04,404 Locate , TopLcol + 2
04,405 Print "Note2: Nine SLOW wildcards are available:"
04,406 Locate , TopLcol + 2
04,407 Print " wildcard '*' any character(s) or empty,"
04,408 Locate , TopLcol + 2
04,409 Print " wildcard '.' any ALPHA character(s) or empty,"
04,410 Locate , TopLcol + 2
04,411 Print " wildcard '' any NON-ALPHA character(s) or empty,"
04,412 Locate , TopLcol + 2
04,413 Print " wildcard '@'/'#' any character {or empty}/{and not empty},"
04,414 Locate , TopLcol + 2
04,415 Print " wildcard '^'/'$' any ALPHA character {or empty}/{and not empty},"
04,416 Locate , TopLcol + 2
04,417 Print " wildcard '!'/'^' any NON-ALPHA character {or empty}/{and not empty}."
04,418 Locate , TopLcol + 2
04,419 Print "Note3: Two FAST wildcards are available:"
04,420 Locate , TopLcol + 2
04,421 Print " wildcard '&' any character(s) or empty,"
04,422 Locate , TopLcol + 2
04,423 Print " wildcard '+' any character and not empty."
04,424 Locate , TopLcol + 2
04,425 Print "Note4: Don't mix SLOW and FAST, the SLOW overrides the FAST. Also, 1 byte exact pattern, not allowed."
04,426 If shrunkP = 0 Then
04,427 Locate , TopLcol + 2
04,428 Print "Exact Case-Sensitive 16-threaded search for all lines up to 26208 chars long:"
04,429 Locate , TopLcol + 2
04,430 Print "Example1: Arnold"
04,431 Locate , TopLcol + 2
04,432 Print "Example2: "; Chr$(34); "metal fatigue"; Chr$(34)
04,433 Locate , TopLcol + 2
04,434 Print "Wildcard Case-Insensitive 16-threaded search for all lines up to 26208 chars long:"
04,435 Locate , TopLcol + 2
04,436 Print "Example3: "; Chr$(34); "out~~~~~ize*"; Chr$(34)
04,437 Locate , TopLcol + 2
04,438 Print " Possible hits within a line starting with O!o: outhyperbolize, OUTSIZE, outsized"
04,439 Locate , TopLcol + 2

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,440 Print "NORMAL/EXHAUSTIVE Fuzzy Case-Insensitive 16-threaded search for all lines up to 156/26208 chars long."
04,441 Locate , TopLcol + 2
04,442 Print "Example4: 3 psychedlicize"
04,443 Locate , TopLcol + 2
04,444 Print "          This line '1234psychedlicize' won't match."
04,445 Locate , TopLcol + 2
04,446 Print "Example5: 2e edelvais"
04,447 Locate , TopLcol + 2
04,448 Print "          Possible hits: edelweiss, edelweisses, psychedelicism"
04,449 Locate , TopLcol + 2
04,450 Print "Note5: Levenshtein search can be NORMAL, as in Example4, matching the whole line."
04,451 Locate , TopLcol + 2
04,452 Print "Note6: Levenshtein search can be EXHAUSTIVE, as in Example5, if LD is postfixed with 'e',"
04,453 Locate , TopLcol + 2
04,454 Print "          matching each position in the line."
04,455 Locate , TopLcol + 2
04,456 Print "Note7: The Needle/Pattern below is set by default as the Current_Word or the CLIPBOARD."
04,457 End If
04,458 Color 7, BckGRcolor
04,459 HelpLine$ = "Edit keys, allowed: Home, Left, Right, End, Backspace, Del, Esc"
04,460 Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print HelpLine$
04,461 _Display
04,462 AddToKazahana$ = InputLine$(TopLrow + 39 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
04,463 StartSt$ = Date$ + " " + Time$
04,464 Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print String$(Len(HelpLine$), " ")
04,465 Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print "Start: "; StartSt$
04,466 Locate 1, 1, 0, CursorS, CursorE
04,467 _Display
04,468
04,469 TimeK1# = Timer(.01)
04,470 $If WINDOWS Then
04,471     Print #13, Chr$(34) + PSPlike$ + "Kazahana.exe" + Chr$(34) + " " + AddToKazahana$ + " " + Chr$(34) + LoadedFile$ + Chr$(34) + " 30539"
04,472 $End If
04,473 $If WINDOWS Then
04,474     'SHELL _DONTWAIT _HIDE CHR$(34) + PSPlike$ + "Kazahana_Hexadecad_GCC_472_SSE41_32bit.exe" + CHR$(34) + " " + AddToKazahana$ + " " + CHR$(34) + LoadedFile$ + CHR$(34) + " 1539" '9 is odd therefore no dump of
pattern field, 9 is bigger than 4 therefore case insensitive wildcard search.
04,475     Shell _Hide Chr$(34) + PSPlike$ + "Kazahana.exe" + Chr$(34) + " " + AddToKazahana$ + " " + Chr$(34) + LoadedFile$ + Chr$(34) + " 30539" '9 is odd therefore no dump of pattern field, 9 is bigger than 4 therefore
case insensitive wildcard search.
04,476 $Else
04,477     Shell _Hide Chr$(34) + PSPlike$ + "kazahana" + Chr$(34) + " " + AddToKazahana$ + " " + Chr$(34) + LoadedFile$ + Chr$(34) + " 30539" '9 is odd therefore no dump of pattern field, 9 is bigger than 4 therefore case
insensitive wildcard search.
04,478 $End If
04,479 TimeK2# = Timer(.01)
04,480 SearchTime# = TimeK2# - TimeK1#
04,481 If SearchTime# < 0 Then SearchTime# = (86400 - TimeK1#) + TimeK2#
04,482 Locate TopLrow + 42 - shrunkP, TopLcol + 2: Print "Ready: "; Date$ + " " + Time$
04,483 f = FreeFile
04,484 Open "Kazahana.txt" For Binary As #f
04,485 KazLen = LOF(f)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,486 Close #f
04,487 SearchRate = FileSize / (SearchTime! + 0.0001)
04,488 Locate TopLrow + 43 - shrunkP, TopLcol + 2: Print "Size of Kazahana.txt: "; AddCommas$(KazLen); " bytes"; "; Search-and-Dump Speed: "; AddCommas$(SearchRate); " bytes/s";
04,489 Locate TopLrow + 44 - shrunkP, TopLcol + 2: Print "Press Esc...";
04,490 _Display
04,491 $If WINDOWS Then
04,492     Play "v20120g"
04,493 $End If
04,494 Do While InKey$ <> Chr$(27)
04,495     _Limit 30
04,496 Loop
04,497 Locate 1, 1, 1, CursorS, CursorE
04,498
04,499 'E:\KAZE_Smxt_Benchmarks\QB64_kit_v1.4_2.48 GB\qb64\Kazahana.Hexadecad_GCC_472_SSE41_32bit.exe
04,500 'Kazahana, a typhoon-class exact & wildcards & Levenshtein Distance (Wagner-Fischer) searcher, r. 1---fix+nowait_critical_nixFIX_Wolfram+fixITER+EX+CS_fix_DEFINE_Trolldom, copyleft Kaze 2019-May-21.
04,501 'Usage: Kazahana [AtMostLevenshteinDistance][e] string textualfile MasterBufferSize
04,502 'Note0: MasterBufferSize is in KB, consider 1024, 3072, 7168 or bigger (up to 2GB). Three additional flags were mapped on this value: all dump
04,503 '       lines (except fuzzy's) will have/lack pattern-source info when the number is even/odd respectively, see Examples #5 and #6.
04,504 '       When MasterBufferSize ends in 0, then No-Dump i.e. hits are only counted.
04,505 'Note0a: Caution! Reported hits are not actual ones but all LINES containing a hit (or hits), e.g. for pattern 'Boom' a line as 'Boom-Boom!' yields one hit not two.
04,506 'Note1: There are three regimes: exact, wildcards and fuzzy searches. First two kick in when 3 parameters are given, fuzzy when 4.
04,507 'Note2: What decides whether exact or wildcards? Of course presence of at least one wildcard. To see exact search see Example #4.
04,508 'Note3: Exact search hits with 'Railgun_Trolldom', not 'Railgun_Sekireigan_Wolfram'.
04,509 'Note4a: Incoming string is automatically lowercased for fuzzy searches i.e. they are case insensitive.
04,510 'Note4b: Incoming string is NOT automatically lowercased for wildcards searches when MasterBufferSize ends in 0..4 i.e. they are case sensitive.
04,511 'Note4c: Incoming string is automatically lowercased for wildcards searches when MasterBufferSize ends in 5..9 i.e. they are case insensitive.
04,512 'Note5: Incoming string could be up to 26208/156 chars for Exact&Wildcard&ExhaustiveFuzzy/Fuzzy respectively.
04,513 'Note5a: Since 2013-Nov-21 Levenshtein search exits not when the incoming line is bigger than 156 chars, now it just skips longer lines.
04,514 'Note5b: Since 2013-Dec-05 Levenshtein search can be EXHAUSTIVE if LD is postfixed with 'e'.
04,515 'Note6: Incoming textualfile could be bigger than 4GB.
04,516 'Note7: Each line should end with [CR]LF, that is Windows or/and UNIX style.
04,517 'Note8: The dump goes to Kazahana.txt file.
04,518 'Note9a: Nine SLOW wildcards are available:
04,519 '       wildcard '*' any character(s) or empty,
04,520 '       wildcard '.' any ALPHA character(s) or empty,
04,521 '       wildcard '`' any NON-ALPHA character(s) or empty,
04,522 '       wildcard '@'/'#' any character {or empty}/{and not empty},
04,523 '       wildcard '^'/'$' any ALPHA character {or empty}/{and not empty},
04,524 '       wildcard '|'/'~' any NON-ALPHA character {or empty}/{and not empty}.
04,525 'Note9b: Two FAST wildcards are available:
04,526 '       wildcard '&' any character(s) or empty,
04,527 '       wildcard '+' any character and not empty.
04,528 'Note9c: Don't mix SLOW and FAST, the SLOW overrides the FAST, i.e. presence of at least one of the 9 wildcards cancels FAST mode.
04,529 'Example1: E:\Kazahana 0 ramjet MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
04,530 'Example2: E:\Kazahana 3 psychedlicize MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
04,531 'Example3: E:\Kazahana "psyched~~~~~ize" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
04,532 'Example4: E:\Kazahana "metal fatigue" enwiki-20121201-pages-articles.xml 7168
04,533 'Example5: E:\Kazahana "out~~~~~ize*" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
04,534 '       E:\>type Kazahana.txt

```

```

Listing: MASAKARI_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus_ToshibaTxl2_8x16.ttf; Downloadable at: www.sanmayce.com/Masakari.zip
04,535 ' [out~~~~~ize*] outhyperbolize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
04,536 ' [out~~~~~ize*] outsize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
04,537 ' [out~~~~~ize*] outsized /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
04,538 ' [out~~~~~ize*] outstrategize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
04,539 ' [out~~~~~ize*] outtyrannize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
04,540 'Example6: E:\>Kazahana "out~~~~~ize*" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1537
04,541 ' E:\>type Kazahana.txt
04,542 ' outhyperbolize
04,543 ' outsize
04,544 ' outsized
04,545 ' outstrategize
04,546 ' outtyrannize
04,547 'Example7: E:\>Kazahana 2e edelvais MASAKARI_General-Purpose_Grade_English_Wordlist.wrd 1024
04,548 ' E:\>type Kazahana.txt
04,549 ' bordelais
04,550 ' bordelaise
04,551 ' edelweiss
04,552 ' edelweisses
04,553 ' foredevised
04,554 ' predellas
04,555 ' psychedelicism
04,556 'Info1: One second seems to have 1,000 clocks.
04,557
04,558 'E:\_KAZE_Smxt_Benchmarks\QB64_kit_v1.4_2.48 GB\qb64>
04,559 End Sub
04,560
04,561
04,562 Function DrawBoxShadowDUMMY$(BckGRcolor, TopLrow, TopLcol, BottomRrow, BottomRcol, Captnn$)
04,563 Shared FileSize
04,564 Shared LoadedFile$
04,565 Shared PSPlike$
04,566 Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$( ), FileArray$( )
04,567 'Shadow
04,568 Color 8, BACKGR
04,569 For i = 1 To MIN$(YdimROW, filecount)
04,570 'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
04,571 'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN
04,572 ' FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
04,573 'ELSE
04,574 ' FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
04,575 'END IF
04,576 If i >= TopLrow + 1 And i <= BottomRrow + 1 Then
04,577 Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
04,578 End If
04,579 Next
04,580 'Outer frame
04,581 Locate TopLrow, TopLcol
04,582 Color 7, BckGRcolor
04,583 Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,584 Color 0, BckGRcolor: Print Chr$(191);
04,585 Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ "; Color 15, BckGRcolor: Print Captmn$; Color 7, BckGRcolor: Print " ]";
04,586 For i = TopLrow + 1 To BottomRow - 1
04,587     Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32)); Color 0, BckGRcolor: Print Chr$(179);
04,588 Next
04,589 Locate BottomRow, TopLcol
04,590 Color 7, BckGRcolor: Print Chr$(192);
04,591 Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)
04,592 'Inner frame
04,593 'In case of YdimROW = 40 then shrunk panel
04,594 shrunkP = 14 'BckGRcolor * 6
04,595 Locate TopLrow + 38 - 23 - shrunkP, TopLcol + 1
04,596 Color 0, BckGRcolor
04,597 Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
04,598 Color 7, BckGRcolor: Print Chr$(191);
04,599 For i = TopLrow + 39 - 23 - shrunkP To TopLrow + 40 - 23 - shrunkP
04,600     Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32)); Color 7, BckGRcolor: Print Chr$(179);
04,601 Next
04,602 Locate TopLrow + 40 - 23 - shrunkP, TopLcol + 1
04,603 Color 0, BckGRcolor: Print Chr$(192);
04,604 Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)
04,605
04,606 Locate TopLrow + 2, TopLcol + 2
04,607 Color 7, BckGRcolor
04,608
04,609 HelpLine$ = "Edit keys, allowed: Home, Left, Right, End, Backspace, Del, Esc"
04,610 Locate TopLrow + 41 - 23 - shrunkP, TopLcol + 3: Print HelpLine$
04,611 Color 0, BckGRcolor
04,612 Locate TopLrow + 41 - 23 - shrunkP, TopLcol + 2: Print "["
04,613 Locate TopLrow + 41 - 23 - shrunkP, TopLcol + 3 + Len(HelpLine$): Print "]"
04,614 Color BckGRcolor, 0: Locate TopLrow + 41 - 23 - shrunkP - 2, TopLcol + 2: Print "Background "; Color BckGRcolor + 8, 0: Print "Background+8"; Color , BckGRcolor: Color 0: Print " COLOR #00 "; Color 8: Print "
COLOR #08 "; Color 9: Print " COLOR #09 "; Color 10: Print " COLOR #10 "; Color 11: Print " COLOR #11 "; Color 12: Print " COLOR #12 "; Color 13: Print " COLOR #13 "; Color 14: Print " COLOR #14 ";
04,615 _Display
04,616 DrawBoxShadowDUMMY$ = InKey$
04,617 Locate 1, 1, 1, CursorS, CursorE
04,618
04,619 End Function
04,620
04,621 Function DrawBoxShadow3exact$(TopLrow, TopLcol, BottomRow, BottomRcol, Captmn$)
04,622     Shared FileSize
04,623     Shared LoadedFile$
04,624     Shared PSPlike$
04,625     Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$(), FileArray$()
04,626     'Shadow
04,627     BckGRcolor = 6
04,628     Color 8, BACKGR
04,629     For i = 1 To MIN$(YdimROW, filecount)
04,630         'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
04,631         'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,632      '   FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
04,633      'ELSE
04,634      '   FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
04,635      'END IF
04,636      If i >= TopLrow + 1 And i <= BottomRow + 1 Then
04,637          Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
04,638      End If
04,639  Next
04,640  'Outer frame
04,641  Locate TopLrow, TopLcol
04,642  Color 7, BckGRcolor
04,643  Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));
04,644  Color 0, BckGRcolor: Print Chr$(191);
04,645  Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ "; Color 15, BckGRcolor: Print Captmn$; Color 7, BckGRcolor: Print " ]";
04,646  For i = TopLrow + 1 To BottomRow - 1
04,647      Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32)); Color 0, BckGRcolor: Print Chr$(179);
04,648  Next
04,649  Locate BottomRow, TopLcol
04,650  Color 7, BckGRcolor: Print Chr$(192);
04,651  Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)
04,652  'Inner frame
04,653  'In case of YdimROW = 40 then shrunk panel
04,654  shrunkP = 14
04,655  Locate TopLrow + 38 - 23 - shrunkP, TopLcol + 1
04,656  Color 0, BckGRcolor
04,657  Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
04,658  Color 7, BckGRcolor: Print Chr$(191);
04,659  For i = TopLrow + 39 - 23 - shrunkP To TopLrow + 40 - 23 - shrunkP
04,660      Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32)); Color 7, BckGRcolor: Print Chr$(179);
04,661  Next
04,662  Locate TopLrow + 40 - 23 - shrunkP, TopLcol + 1
04,663  Color 0, BckGRcolor: Print Chr$(192);
04,664  Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)
04,665
04,666  Locate TopLrow + 2, TopLcol + 2
04,667  Color 7, BckGRcolor
04,668
04,669  HelpLine$ = "Edit keys, allowed: Home, Left, Right, End, Backspace, Del, Esc"
04,670  Locate TopLrow + 41 - 23 - shrunkP, TopLcol + 2: Print HelpLine$
04,671  _Display
04,672  DrawBoxShadow3exact$ = InputLine$(TopLrow + 39 - 23 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
04,673  Locate 1, 1, 1, CursorS, CursorE
04,674
04,675 End Function
04,676
04,677 Function DrawBoxShadow3exactRussian$(TopLrow, TopLcol, BottomRow, BottomRcol, Captmn$)
04,678     Shared FileSize
04,679     Shared LoadedFile$
04,680     Shared PSPlike$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,681 Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$( ), FileArray$( )
04,682 'Shadow
04,683 BckGRcolor = 5
04,684 Color 8, BACKGR
04,685 For i = 1 To MIN$(YdimROW, filecount)
04,686     'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
04,687     'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN
04,688         'FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
04,689     'ELSE
04,690         'FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
04,691     'END IF
04,692     If i >= TopLrow + 1 And i <= BottomRow + 1 Then
04,693         Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
04,694     End If
04,695 Next
04,696 'Outer frame
04,697 Locate TopLrow, TopLcol
04,698 Color 7, BckGRcolor
04,699 Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));
04,700 Color 0, BckGRcolor: Print Chr$(191);
04,701 Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ ";: Color 15, BckGRcolor: Print Captmn$;: Color 7, BckGRcolor: Print " ]";
04,702 For i = TopLrow + 1 To BottomRow - 1
04,703     Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32));: Color 0, BckGRcolor: Print Chr$(179);
04,704 Next
04,705 Locate BottomRow, TopLcol
04,706 Color 7, BckGRcolor: Print Chr$(192);
04,707 Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)
04,708 'Inner frame
04,709 'In case of YdimROW = 40 then shrunk panel
04,710 shrunkP = 14
04,711 Locate TopLrow + 38 - 23 - shrunkP, TopLcol + 1
04,712 Color 0, BckGRcolor
04,713 Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
04,714 Color 7, BckGRcolor: Print Chr$(191);
04,715 For i = TopLrow + 39 - 23 - shrunkP To TopLrow + 40 - 23 - shrunkP
04,716     Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32));: Color 7, BckGRcolor: Print Chr$(179);
04,717 Next
04,718 Locate TopLrow + 40 - 23 - shrunkP, TopLcol + 1
04,719 Color 0, BckGRcolor: Print Chr$(192);
04,720 Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)
04,721
04,722 Locate TopLrow + 2, TopLcol + 2
04,723 Color 7, BckGRcolor
04,724
04,725 HelpLine$ = "Edit keys, allowed: Home, Left, Right, End, Backspace, Del, Esc"
04,726 Locate TopLrow + 41 - 23 - shrunkP, TopLcol + 2: Print HelpLine$
04,727 _Display
04,728 DrawBoxShadow3exactRussian$ = InputLineRussian$(TopLrow + 39 - 23 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
04,729 Locate 1, 1, 1, CursorS, CursorE

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,730
04,731 End Function
04,732
04,733 Function InputLine$ (Irow, Icol, LineLen)
04,734     Static a$
04,735     Shared CurrentWord$
04,736     Locate Irow, Icol, 1, CursorS, CursorE
04,737     MaxCol = Icol + LineLen - 1
04,738     CurCol = Icol 'Icol..MaxCol
04,739     'IF CurrentWord$ = "" THEN
04,740     '     IF INSTR(_CLIPBOARD$, CHR$(10)) THEN ' _CLIPBOARD$ could be multi-line, so get the first
04,741     '         a2$ = LEFT$( _CLIPBOARD$, INSTR(_CLIPBOARD$, CHR$(10)) - 1)
04,742     '         IF RIGHT$(a2$, 1) = CHR$(13) THEN a2$ = LEFT$(a2$, LEN(a2$) - 1)
04,743     '         a$ = LEFT$(a2$, LineLen - 1) '""
04,744     '     ELSE
04,745     '         a$ = LEFT$( _CLIPBOARD$, LineLen - 1) '""
04,746     '     END IF
04,747     '     CurCol = Icol + LEN(a$)
04,748 'ELSE
04,749 '     a$ = CurrentWord$
04,750 'END IF
04,751 Do
04,752     key$ = InKey$
04,753     If key$ <> "" Then
04,754         code% = Asc(key$):
04,755         If code% Then ' ASC returns any value greater than 0
04,756             Select Case Asc(key$)
04,757                 Case 27:
04,758                     a$ = "": CurCol = Icol
04,759                 Case 13:
04,760                     If a$ <> "" Then Exit Do
04,761                 Case 8: 'Backspace
04,762                     If CurCol <> Icol Then
04,763                         If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
04,764                             a$ = Left$(a$, Len(a$) - 1)
04,765                         Else 'inhere we are not at the end i.e. 1..len(a$)-1
04,766                             a$ = Left$(a$, (CurCol - Icol) - 1) + Right$(a$, Len(a$) - (CurCol - Icol))
04,767                         End If
04,768                         CurCol = CurCol - 1
04,769                     End If
04,770                 Case 32 TO 127 - 1:
04,771                     If Len(a$) < LineLen - 1 Then 'a$ = a$ + key$: CurCol = CurCol + 1 '-1 because cursor goes next to the field otherwise
04,772                     If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
04,773                         a$ = a$ + key$
04,774                     Else 'inhere we are not at the end i.e. 1..len(a$)-1
04,775                         ' Insert at (CurCol - Icol) + 1:
04,776                         a$ = Left$(a$, (CurCol - Icol)) + key$ + Right$(a$, Len(a$) - (CurCol - Icol))
04,777                     End If
04,778                     CurCol = CurCol + 1

```

```

04,779         End If
04,780     End Select
04,781 Else
04,782     Select Case Asc(key$, 2)
04,783     Case 75: If CurCol > Icol Then CurCol = CurCol - 1
04,784     Case 77: If CurCol < MaxCol And CurCol < Icol + Len(a$) Then CurCol = CurCol + 1
04,785     Case 71: CurCol = Icol 'Home
04,786     Case 79: CurCol = Icol + Len(a$) 'End
04,787     Case 83: 'Del
04,788         If a$ <> "" Then
04,789             a$ = Left$(a$, (CurCol - Icol)) + Right$(a$, Len(a$) - (CurCol - Icol) - 1)
04,790         End If
04,791     Case 82: 'Ins
04,792     End Select
04,793 End If
04,794 End If
04,795 Locate Irow, Icol, 1, CursorS, CursorE
04,796 Print String$(LineLen, " ");
04,797 Locate Irow, Icol, 1, CursorS, CursorE
04,798 Print a$;
04,799 Locate Irow, CurCol, 1, CursorS, CursorE
04,800 _Display
04,801 _Limit 30
04,802 Loop
04,803 'DO WHILE _KEYDOWN(13): LOCATE , , 0: _DISPLAY: LOOP
04,804 _KeyClear
04,805 InputLine$ = a$
04,806 End Function
04,807
04,808 Function InputLineRussian$ (Irow, Icol, LineLen)
04,809     Static a$
04,810     Shared CurrentWord$
04,811     Locate Irow, Icol, 1, CursorS, CursorE
04,812     MaxCol = Icol + LineLen - 1
04,813     CurCol = Icol 'Icol..MaxCol
04,814     'IF CurrentWord$ = "" THEN
04,815     '     IF INSTR(_CLIPBOARD$, CHR$(10)) THEN ' _CLIPBOARD$ could be multi-line, so get the first
04,816     '         a2$ = LEFT$(_CLIPBOARD$, INSTR(_CLIPBOARD$, CHR$(10)) - 1)
04,817     '         IF RIGHT$(a2$, 1) = CHR$(13) THEN a2$ = LEFT$(a2$, LEN(a2$) - 1)
04,818     '         a$ = LEFT$(a2$, LineLen - 1) '""
04,819     '     ELSE
04,820     '         a$ = LEFT$(_CLIPBOARD$, LineLen - 1) '""
04,821     '     END IF
04,822     '     CurCol = Icol + LEN(a$)
04,823 'ELSE
04,824 '     a$ = CurrentWord$
04,825 'END IF
04,826 Do
04,827     key$ = InKey$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,828      If key$ <> "" Then
04,829          code% = Asc(key$):
04,830          If code% Then ' ASC returns any value greater than 0
04,831              Select Case Asc(key$)
04,832                  Case 27:
04,833                      a$ = ""; CurCol = Icol
04,834                  Case 13:
04,835                      If a$ <> "" Then Exit Do
04,836                  Case 8: 'Backspace
04,837                      If CurCol <> Icol Then
04,838                          If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
04,839                              a$ = Left$(a$, Len(a$) - 1)
04,840                          Else 'inhere we are not at the end i.e. 1..len(a$)-1
04,841                              a$ = Left$(a$, (CurCol - Icol) - 1) + Right$(a$, Len(a$) - (CurCol - Icol))
04,842                          End If
04,843                          CurCol = CurCol - 1
04,844                      End If
04,845                  Case 32 TO 127 - 1:
04,846                      k$ = key$
04,847                      If k$ = "~" Then k$ = Chr$(250)
04,848                      If k$ = "`" Then k$ = Chr$(251)
04,849
04,850                      If k$ = "+" Then k$ = Chr$(154)
04,851                      If k$ = "=" Then k$ = Chr$(234)
04,852
04,853                      If k$ = "Q" Then k$ = Chr$(159)
04,854                      If k$ = "q" Then k$ = Chr$(239)
04,855
04,856                      If k$ = "W" Then k$ = Chr$(152)
04,857                      If k$ = "w" Then k$ = Chr$(232)
04,858
04,859                      If k$ = "E" Then k$ = Chr$(133)
04,860                      If k$ = "e" Then k$ = Chr$(165)
04,861
04,862                      If k$ = "R" Then k$ = Chr$(144)
04,863                      If k$ = "r" Then k$ = Chr$(224)
04,864
04,865                      If k$ = "T" Then k$ = Chr$(146)
04,866                      If k$ = "t" Then k$ = Chr$(226)
04,867
04,868                      If k$ = "Y" Then k$ = Chr$(155)
04,869                      If k$ = "y" Then k$ = Chr$(235)
04,870
04,871                      If k$ = "U" Then k$ = Chr$(147)
04,872                      If k$ = "u" Then k$ = Chr$(227)
04,873
04,874                      If k$ = "I" Then k$ = Chr$(136)
04,875                      If k$ = "i" Then k$ = Chr$(168)
04,876

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,877      If k$ = "O" Then k$ = Chr$(142)
04,878      If k$ = "o" Then k$ = Chr$(174)
04,879
04,880      If k$ = "P" Then k$ = Chr$(143)
04,881      If k$ = "p" Then k$ = Chr$(175)
04,882
04,883      If k$ = "{" Then k$ = Chr$(158)
04,884      If k$ = "[" Then k$ = Chr$(238)
04,885
04,886      If k$ = "}" Then k$ = Chr$(153)
04,887      If k$ = "]" Then k$ = Chr$(233)
04,888
04,889      If k$ = "!" Then k$ = Chr$(157)
04,890      If k$ = "\" Then k$ = Chr$(237)
04,891
04,892      If k$ = "A" Then k$ = Chr$(128)
04,893      If k$ = "a" Then k$ = Chr$(160)
04,894
04,895      If k$ = "S" Then k$ = Chr$(145)
04,896      If k$ = "s" Then k$ = Chr$(225)
04,897
04,898      If k$ = "D" Then k$ = Chr$(132)
04,899      If k$ = "d" Then k$ = Chr$(164)
04,900
04,901      If k$ = "F" Then k$ = Chr$(148)
04,902      If k$ = "f" Then k$ = Chr$(228)
04,903
04,904      If k$ = "G" Then k$ = Chr$(131)
04,905      If k$ = "g" Then k$ = Chr$(163)
04,906
04,907      If k$ = "H" Then k$ = Chr$(151)
04,908      If k$ = "h" Then k$ = Chr$(231)
04,909
04,910      If k$ = "J" Then k$ = Chr$(137)
04,911      If k$ = "j" Then k$ = Chr$(169)
04,912
04,913      If k$ = "K" Then k$ = Chr$(138)
04,914      If k$ = "k" Then k$ = Chr$(170)
04,915
04,916      If k$ = "L" Then k$ = Chr$(139)
04,917      If k$ = "l" Then k$ = Chr$(171)
04,918
04,919      If k$ = "." Then k$ = Chr$(156)
04,920      If k$ = ";" Then k$ = Chr$(236)
04,921
04,922      If k$ = Chr$(34) Then k$ = Chr$(134)
04,923      If k$ = "'" Then k$ = Chr$(166)
04,924
04,925      If k$ = "Z" Then k$ = Chr$(135)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,926         If k$ = "z" Then k$ = Chr$(167)
04,927
04,928         If k$ = "X" Then k$ = Chr$(149)
04,929         If k$ = "x" Then k$ = Chr$(229)
04,930
04,931         If k$ = "C" Then k$ = Chr$(150)
04,932         If k$ = "c" Then k$ = Chr$(230)
04,933
04,934         If k$ = "V" Then k$ = Chr$(130)
04,935         If k$ = "v" Then k$ = Chr$(162)
04,936
04,937         If k$ = "B" Then k$ = Chr$(129)
04,938         If k$ = "b" Then k$ = Chr$(161)
04,939
04,940         If k$ = "N" Then k$ = Chr$(141)
04,941         If k$ = "n" Then k$ = Chr$(173)
04,942
04,943         If k$ = "M" Then k$ = Chr$(140)
04,944         If k$ = "m" Then k$ = Chr$(172)
04,945
04,946         key$ = k$
04,947         If Len(a$) < LineLen - 1 Then 'a$ = a$ + key$: CurCol = CurCol + 1 '-1 because cursor goes next to the field otherwise
04,948             If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
04,949                 a$ = a$ + key$
04,950             Else 'inhere we are not at the end i.e. 1..len(a$)-1
04,951                 ' Insert at (CurCol - Icol) + 1:
04,952                 a$ = Left$(a$, (CurCol - Icol)) + key$ + Right$(a$, Len(a$) - (CurCol - Icol))
04,953             End If
04,954             CurCol = CurCol + 1
04,955         End If
04,956     End Select
04,957 Else
04,958     Select Case Asc(key$, 2)
04,959         Case 75: If CurCol > Icol Then CurCol = CurCol - 1
04,960         Case 77: If CurCol < MaxCol And CurCol < Icol + Len(a$) Then CurCol = CurCol + 1
04,961         Case 71: CurCol = Icol 'Home
04,962         Case 79: CurCol = Icol + Len(a$) 'End
04,963         Case 83: 'Del
04,964             If a$ <> "" Then
04,965                 a$ = Left$(a$, (CurCol - Icol)) + Right$(a$, Len(a$) - (CurCol - Icol) - 1)
04,966             End If
04,967         Case 82: 'Ins
04,968     End Select
04,969 End If
04,970 End If
04,971 Locate Irow, Icol, 1, CursorS, CursorE
04,972 Print String$(LineLen, " ");
04,973 Locate Irow, Icol, 1, CursorS, CursorE
04,974 Print a$;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

04,975      Locate Irow, CurCol, 1, CursorS, CursorE
04,976      _Display
04,977      _Limit 30
04,978      Loop
04,979      'DO WHILE _KEYDOWN(13): LOCATE , , 0: _DISPLAY: LOOP
04,980      _KeyClear
04,981      InputLineRussian$ = a$
04,982 End Function
04,983
04,984 Sub NextFrame (TopRow)
04,985     Shared ASCIIIFrame
04,986     Shared PSPlike$
04,987     Locate TopRow, 73
04,988     ASCIIIFrame = ASCIIIFrame + 1
04,989     If ASCIIIFrame > 20 Then ASCIIIFrame = 1 '1..20
04,990     f = FreeFile
04,991     fr$ = LTrim$(Str$(ASCIIIFrame))
04,992     If Len(fr$) = 1 Then fr$ = "0" + fr$
04,993     If _FileExists(PSPlike$ + "glass" + fr$ + ".txt") Then
04,994         Open PSPlike$ + "glass" + fr$ + ".txt" For Input As #f
04,995         Do While Not EOF(f)
04,996             Line Input #f, a$
04,997             Locate , 73
04,998             Print a$
04,999         Loop
05,000         Close #f
05,001     End If
05,002 End Sub
05,003
05,004 Sub NextFramePEN (TopRow)
05,005     Shared ASCIIIFramePEN
05,006     Shared PSPlike$
05,007     Shared XdimCOL
05,008     If XdimCOL = 213 Then HorizPos = 128 Else HorizPos = 70
05,009     Locate TopRow, HorizPos
05,010     ASCIIIFramePEN = ASCIIIFramePEN + 1
05,011     If ASCIIIFramePEN > 24 Then ASCIIIFramePEN = 1 '1..20
05,012     f = FreeFile
05,013     fr$ = LTrim$(Str$(ASCIIIFramePEN))
05,014     If Len(fr$) = 1 Then fr$ = "0" + fr$
05,015     If _FileExists(PSPlike$ + "pen" + fr$ + ".txt") Then
05,016         Open PSPlike$ + "pen" + fr$ + ".txt" For Input As #f
05,017         Do While Not EOF(f)
05,018             Line Input #f, a$
05,019             Locate , HorizPos
05,020             Print Right$(a$, 80 - 23)
05,021         Loop
05,022         Close #f
05,023     End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,024 End Sub
05,025
05,026 Sub ShowF1
05,027     Shared Mrev$
05,028     PostFix$ = ""
05,029     'IF ToLoadOrNotFlag = 0 THEN PostFix$ = PostFix$ + "_External" ELSE PostFix$ = PostFix$ + "_Fast"
05,030     If WrapFlag = 1 Then PostFix$ = PostFix$ + "_Wrapper" Else PostFix$ = PostFix$ + "_Vanilla"
05,031     Print "Masakari, revision " + Mrev$ + PostFix$ + ", written in QB64 by Kaze, source code downloadable at https://www.qb64.org/forum"
05,032
05,033     $If WINDOWS Then
05,034         Print "Usage: Masakari [filename]![/help][--ascii[_gesch]]"
05,035     $Else
05,036         Print "Usage: Masakari [filename]![-h][--ascii[_gesch]]"
05,037     $End If
05,038     Print "Note: The 'filename' could be a filelist, i.e. a list of filenames (see Space and Double-Left-Click)."

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,073 Print "      If unwrappable lines exist then those lines are dumped to filename+"; Chr$(34); ".unwrappable"; Chr$(34); ","
05,074 Print "      otherwise, the wrapped lines are dumped to filename+"; Chr$(34); ".wrapped"; Chr$(34); ", and auto-loaded."
05,075 Print "      If wrapped file exists during start then it is used, not re-created."
05,076 End Sub
05,077
05,078 Function timeElapsedSince! (startTime!)
05,079     If startTime! > Timer Then startTime! = startTime! - 86400
05,080     timeElapsedSince! = Timer - startTime!
05,081 End Function
05,082
05,083 Function CROPorPADatRIGHT$ (victim$, linelen)
05,084     If Len(victim$) > linelen Then
05,085         CROPorPADatRIGHT$ = Left$(victim$, linelen)
05,086     Else
05,087         CROPorPADatRIGHT$ = victim$ + Space$(linelen - Len(victim$))
05,088     End If
05,089 End Function
05,090
05,091 'CONST LSHIFTkey& = 100304
05,092 'DO
05,093 '    IF _KEYDOWN(LSHIFTkey&) = 0 AND IsF3released THEN PRINT "F3 released ";
05,094 '    _LIMIT 500
05,095 'LOOP
05,096 Function IsF3released
05,097     IsF3released = 0
05,098     MustBeLONGsigned& = _KeyHit
05,099     If MustBeLONGsigned& Then
05,100         If MustBeLONGsigned& < 0 Then 'negative value means key released
05,101             MustBeLONGsigned& = -MustBeLONGsigned&
05,102             If MustBeLONGsigned& \ 256 = 61 Then
05,103                 IsF3released = 1
05,104             Else
05,105                 End If
05,106         End If
05,107     End If
05,108 End Function
05,109
05,110 Function IsENTERreleased
05,111     IsENTERreleased = 0
05,112     MustBeLONGsigned& = _KeyHit
05,113     If MustBeLONGsigned& Then
05,114         If MustBeLONGsigned& < 0 Then 'negative value means key released
05,115             MustBeLONGsigned& = -MustBeLONGsigned&
05,116             If MustBeLONGsigned& = 13 Then
05,117                 IsENTERreleased = 1
05,118             Else
05,119                 End If
05,120         End If
05,121     End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

05,171 Open dict01$ For Binary As #f01
05,172 If LOF(f01) Then
05,173
05,174     f02 = FreeFile
05,175     Open dict01$ + ".ind" For Binary As #f02 'LEN = 35 + 2 + 8 + 8 '+2 for CRLF and DWORD+DWORD as padded HEX
05,176     Do While Not EOF(f01)
05,177         EntryStartAddr& = Seek(f01)
05,178         Line Input #f01, l$
05,179         EntryLen = Seek(f01)
05,180         'IF INSTR(l$, CHR$(9)) > 35 THEN PRINT l$
05,181         If Instr(l$, Chr$(9)) Then
05,182             dict01_lfixed$ = Left$(l$, Instr(l$, Chr$(9)) - 1)
05,183             dict01_lfixedCRLF$ = dict01_lfixed$ + PADzeros$(Hex$(EntryStartAddr&)) + PADzeros$(Hex$(EntryLen - EntryStartAddr&)) + Chr$(13) + Chr$(10)
05,184             Put #f02, dict01_lfixedCRLF$: dict01_TotalWrd& = dict01_TotalWrd& + 1
05,185             'PRINT "EntryStartAddr/EntryLen = "; EntryStartAddr& "/"; EntryLen - EntryStartAddr&
05,186             '_DELAY .01
05,187         End If
05,188     Loop
05,189     Close #f01, #f02
05,190     Print "Total entries for '"; dict01$; "'": dict01_TotalWrd&
05,191     Print "CurrentWord$: "; CurrentWord$
05,192     Shell_Hide "CMD /C sort " + dict01$ + ".ind" + " > " + "sorted_" + dict01$ + ".ind"
05,193     f01 = FreeFile
05,194     Open dict01$ For Binary As #f01
05,195     f02 = FreeFile
05,196     Open "sorted_" + dict01$ + ".ind" For Random Access Read As #f02 Len = RecordLen '35 + 2 + 8 + 8 '+2 for CRLF and DWORD+DWORD as padded HEX
05,197
05,198     'testing 537 line:
05,199     'alimentary          00025E8D00000118
05,200     'PhysicalLine& = 19001 '537
05,201
05,202     ' BINARY SEARCH [
05,203     UCurrentWord$ = LCase$(CurrentWord$)
05,204     WRDpos& = 0
05,205     LeftPoint& = 1
05,206     RightPoint& = dict01_TotalWrd&
05,207     Do
05,208         If RightPoint& - LeftPoint& <= 1 Then 'Sxsedtcheta
05,209             Get #f02, LeftPoint&, dict01_lfixedCRLF$
05,210             If UCurrentWord$ = LCase$(RTrim$(Left$(dict01_lfixedCRLF$, EntryHead))) Then WRDpos& = LeftPoint&: Exit Do
05,211             Get #f02, RightPoint&, dict01_lfixedCRLF$
05,212             If UCurrentWord$ = LCase$(RTrim$(Left$(dict01_lfixedCRLF$, EntryHead))) Then WRDpos& = RightPoint&: Exit Do
05,213             WRDpos& = LeftPoint& 'THIS IS BECAUSE I NEED NEAREST word not zero!
05,214             Exit Do
05,215         End If
05,216         SplitPoint& = (LeftPoint& + RightPoint&) \ 2
05,217         Get #f02, SplitPoint&, dict01_lfixedCRLF$
05,218         If UCurrentWord$ < LCase$(RTrim$(Left$(dict01_lfixedCRLF$, EntryHead))) Then
05,219             RightPoint& = SplitPoint&

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,220      ElseIf UCurrentWord$ > LCase$(RTrim$(Left$(dict01_lfixedCRLF$, EntryHead))) Then
05,221          LeftPoint& = SplitPoint&
05,222      Else
05,223          WRDpos& = SplitPoint& Exit Do
05,224      End If
05,225      Loop
05,226      If WRDpos& < 0 Then
05,227          Do While WRDpos& > 1
05,228              WRDpos& = WRDpos& - 1
05,229              Get #f02, WRDpos&, dict01_lfixedCRLF$
05,230              If UCurrentWord$ < LCase$(RTrim$(Left$(dict01_lfixedCRLF$, EntryHead))) Then
05,231                  WRDpos& = WRDpos& + 1
05,232              Exit Do
05,233          End If
05,234      Loop
05,235      'IF WRDpos& <= TotalWrds - 16 THEN FL2wrds = WRDpos& ELSE FL2wrds = TotalWrds - 16
05,236      End If
05,237      ' BINARY SEARCH ]
05,238
05,239      $If WINDOWS Then
05,240          Play "v20o2l40gaf"
05,241      $End If
05,242
05,243      Physicalline& = WRDpos&
05,244      'GET #f02, (Physicalline& - 1) * (35 + 2 + 8 + 8) + 1, dict01_lfixedCRLF$ 'when OPENed for BINARY
05,245      Get #f02, Physicalline&, dict01_lfixedCRLF$
05,246      EntryAll$ = Space$(dePADzeros&(Mid$(dict01_lfixedCRLF$, (EntryHead + 1) + 8, 8)))
05,247      Get #f01, dePADzeros&(Mid$(dict01_lfixedCRLF$, (EntryHead + 1), 8)), EntryAll$
05,248      Print "EntryHead: "; Left$(EntryAll$, InStr(EntryAll$, Chr$(9)) - 1)
05,249      EntryTail$ = "EntryTail: " + Right$(EntryAll$, Len(EntryAll$) - InStr(EntryAll$, Chr$(9)))
05,250      Print EntryTail$
05,251      'PRINT "[" + EntryAll$ + "]"
05,252      Close #f01, #f02
05,253      _Display
05,254      Do
05,255          key$ = InKey$
05,256          _Limit 60
05,257      Loop Until _KeyDown(27)
05,258      Else
05,259          $If WINDOWS Then
05,260              Play "L16V3203g-g-<<d+"
05,261          $End If
05,262          Close #f01
05,263      End If 'If LOF(f01) Then
05,264      End Sub
05,265
05,266      ' Dictionary_Specification_Language_(ABBYY_Software_House)_Oxford_English_Dictionary_2nd_Edition_Version_4_(En-En)_ANSI.dsl
05,267      ' Dictionary_Specification_Language_(ABBYY_Software_House)_Webster's_Unabridged_3_(En-En)_UTF-8.dsl
05,268      ' Machine-Learning_Urban_Dictionary_Definitions_Corpus_(1999_-_May-2016).words.json

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,269 Sub dict02sub
05,270   Shared CurrentWord$
05,271   Const dict02_EntryHead = 34
05,272   Const dict02_RecordLen = dict02_EntryHead + 8 + 8 + 2 '34+2+8+8
05,273   Dim dict02_lfixed As String * dict02_EntryHead
05,274   Dim dict02_lfixedCRLF As String * dict02_RecordLen
05,275   dict02_TotalWrd& = 0
05,276   dict02_LongestEntryHead& = 0
05,277
05,278   dict02$ = "Dictionary_Specification_Language_(ABBY Software_House)_Oxford_English_Dictionary_2nd_Edition_Version_4_(En-En)_ANSI.dsl"
05,279   Rem Caution: One EntryHead could be multi-line!
05,280   Rem 'cellist
05,281   Rem cellist
05,282
05,283   If _FileExists(PSPlike$ + "sorted_" + dict02$ + ".ind") = 0 Then 'check for existance [[[
05,284       f01 = FreeFile
05,285       Open dict02$ For Binary As #f01
05,286       If LOF(f01) Then
05,287
05,288           f02 = FreeFile
05,289           Open dict02$ + ".ind" For Binary As #f02 'LEN = 35 + 2 + 8 + 8 '+2 for CRLF and DWORD+DWORD as padded HEX
05,290           OldOff& = 1
05,291           Do While Not EOF(f01)
05,292               NewOff& = Seek(f01)
05,293               Line Input #f01, l$
05,294               'IF LEN(l$) > dict02_LongestEntryHead& THEN dict02_LongestEntryHead& = LEN(l$)
05,295               If Left$(l$, 1) = Chr$(9) Then
05,296                   Else
05,297                       If OldOff& <> NewOff& Then
05,298                           EntryStartAddr& = OldOff&
05,299                           EntryLen = NewOff& - OldOff&
05,300                           OldOff& = NewOff&
05,301                           dict02_lfixed$ = lprev$
05,302                           dict02_lfixedCRLF$ = dict02_lfixed$ + PADzeros$(Hex$(EntryStartAddr&)) + PADzeros$(Hex$(EntryLen)) + Chr$(13) + Chr$(10)
05,303                           Put #f02, , dict02_lfixedCRLF$: dict02_TotalWrd& = dict02_TotalWrd& + 1
05,304                       End If
05,305                       lprev$ = l$
05,306                   End If
05,307               Loop
05,308               'The last entry has no END [
05,309               NewOff& = Seek(f01) ' PRINT NewOff& gives LOF(f01)+1
05,310               EntryStartAddr& = OldOff&
05,311               EntryLen = NewOff& - OldOff&
05,312               OldOff& = NewOff&
05,313               dict02_lfixed$ = lprev$
05,314               dict02_lfixedCRLF$ = dict02_lfixed$ + PADzeros$(Hex$(EntryStartAddr&)) + PADzeros$(Hex$(EntryLen)) + Chr$(13) + Chr$(10)
05,315               Put #f02, , dict02_lfixedCRLF$: dict02_TotalWrd& = dict02_TotalWrd& + 1
05,316
05,317               'The last entry has no END ]

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,318
05,319         Close #f01, #f02
05,320         'PRINT dict02_LongestEntryHead& ' 34 for OED
05,321         'PRINT "Total entries for " & dict02$; "':"; dict02_TotalWrds&
05,322
05,323         'PRINT "CurrentWord$: "; CurrentWord$
05,324         Shell _Hide "CMD /C sort " & Chr$(34) & dict02$ & ".ind" & Chr$(34) & " > " & Chr$(34) & "sorted_" & dict02$ & ".ind" & Chr$(34)
05,325
05,326     Else
05,327         $If WINDOWS Then
05,328             Play "L16V3203g-g-<<d+"
05,329         $End If
05,330         Close #f01
05,331     End If 'If LOF(f01) Then
05,332 Else
05,333     dict02_TotalWrds = 277315
05,334 End If 'check for existence ]]
05,335 If _FileExists(PSPlike$ & "sorted_" & dict02$ & ".ind") <> 0 Then 'check for existence [[[
05,336     f01 = FreeFile
05,337     Open dict02$ For Binary As #f01
05,338     f02 = FreeFile
05,339     Open "sorted_" & dict02$ & ".ind" For Random Access Read As #f02 Len = dict02_RecordLen '35 + 2 + 8 + 8 '+2 for CRLF and DWORD+DWORD as padded HEX
05,340
05,341     'testing 537 line:
05,342     'alimentary          00025E8D00000118
05,343     'PhysicalLine& = 19001 '537
05,344
05,345     ' BINARY SEARCH [
05,346     UCurrentWord$ = LCase$(CurrentWord$)
05,347     WRDpos& = 0
05,348     LeftPoint& = 1
05,349     RightPoint& = dict02_TotalWrds
05,350     Do
05,351         If RightPoint& - LeftPoint& <= 1 Then 'Sxsedtceta
05,352             Get #f02, LeftPoint&, dict02_lfixedCRLF$
05,353             If UCurrentWord$ = LCase$(RTrim$(Left$(dict02_lfixedCRLF$, dict02_EntryHead))) Then WRDpos& = LeftPoint&: Exit Do
05,354             Get #f02, RightPoint&, dict02_lfixedCRLF$
05,355             If UCurrentWord$ = LCase$(RTrim$(Left$(dict02_lfixedCRLF$, dict02_EntryHead))) Then WRDpos& = RightPoint&: Exit Do
05,356             WRDpos& = LeftPoint& 'THIS IS BECAUSE I NEED NEAREST word not zero!
05,357             Exit Do
05,358         End If
05,359         SplitPoint& = (LeftPoint& + RightPoint&) \ 2
05,360         Get #f02, SplitPoint&, dict02_lfixedCRLF$
05,361         If UCurrentWord$ < LCase$(RTrim$(Left$(dict02_lfixedCRLF$, dict02_EntryHead))) Then
05,362             RightPoint& = SplitPoint&
05,363         ElseIf UCurrentWord$ > LCase$(RTrim$(Left$(dict02_lfixedCRLF$, dict02_EntryHead))) Then
05,364             LeftPoint& = SplitPoint&
05,365         Else
05,366             WRDpos& = SplitPoint&: Exit Do

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,367     End If
05,368     Loop
05,369     If WRDpos& < 0 Then
05,370         Do While WRDpos& > 1
05,371             WRDpos& = WRDpos& - 1
05,372             Get #f02, WRDpos&, dict02_lfixedCRLF$
05,373             If UCurrentWord$ < LCase$(RTrim$(Left$(dict02_lfixedCRLF$, dict02_EntryHead))) Then
05,374                 WRDpos& = WRDpos& + 1
05,375             Exit Do
05,376         End If
05,377     Loop
05,378     'IF WRDpos& <= TotalWrd& - 16 THEN FL2wrd& = WRDpos& ELSE FL2wrd& = TotalWrd& - 16
05,379 End If
05,380 ' BINARY SEARCH ]
05,381
05,382 $If WINDOWS Then
05,383     Play "v20o2l20geb"
05,384 $End If
05,385 Physicalline& = WRDpos&
05,386 'Physicalline& = 277315 '277315 total; WARNING! 61358 is 'dictionary' - cannot be found because words as 'dicæarch' confuse the binary search!
05,387 'GET #f02, (Physicalline& - 1) * (35 + 2 + 8 + 8) + 1, dict02_lfixedCRLF$ 'when OPENed for BINARY
05,388 Get #f02, Physicalline&, dict02_lfixedCRLF$
05,389
05,390
05,391 ' Scrolling into 3D djamtche [
05,392 'SUB DrawBoxShadow3 (TopLrow, TopLcol, BottomRow, BottomRcol, Captnn$)
05,393 If Len(dict02$) > 70 Then Reduced$ = "..." + Right$(dict02$, 70) Else Reduced$ = dict02$
05,394 Call DrawBoxShadowOED(3 + 3, 3, 35, 125, Reduced$, 2)
05,395 BckGRcolor = 2 '6
05,396 s$ = LTrim$(Str$(Physicalline&))
05,397 If Len(s$) > 3 Then
05,398     If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
05,399     s$ = ""
05,400     For i = 1 To Len(x$) Step 3
05,401         s$ = s$ + Mid$(x$, i, 3) + ", "
05,402     Next
05,403     s$ = Left$(s$, Len(s$) - 1)
05,404 End If
05,405 s$ = LTrim$(s$)
05,406 Padded$ = "0,000,000,000"
05,407 Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
05,408 Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
05,409 Locate , , 0
05,410
05,411 DjampoolLength = 125 - 3 + 1 - 4
05,412 DjampoolHeight = 35 - (3 + 3) + 1 - 4
05,413 CurrentPos& = Physicalline& ' Notice that CurrentPos& = 1..(dict02_TotalWrd& - DjampoolHeight + 1)
05,414 ' inhere it could be between 1.. due to BINARY search
05,415 If CurrentPos& > (dict02_TotalWrd& - DjampoolHeight + 1) Then CurrentPos& = (dict02_TotalWrd& - DjampoolHeight + 1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,416 PoolLinePos = 1
05,417 Do
05,418   key$ = InKey$
05,419   'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
05,420   If key$ <> "" Then
05,421     code% = Asc(key$):
05,422     If code% Then ' ASC returns any value greater than 0
05,423       Select Case Asc(key$)
05,424         Case 13:
05,425           Call DrawBoxShadowOEDin(3 + 3, 3, 35, 125, Reduced$)
05,426           BckGColor = 2
05,427           Locate , , 0
05,428           Get #f02, CurrentPos% + (PoolLinePos - 1), dict02_lfixedCRLF$
05,429           EntryAll$ = Space$(dePADzeros&(Mid$(dict02_lfixedCRLF$, (dict02_EntryHead + 1) + 8, 8)))
05,430           Get #f01, dePADzeros&(Mid$(dict02_lfixedCRLF$, (dict02_EntryHead + 1), 8)), EntryAll$
05,431           f03 = FreeFile
05,432           Open "Entry" For Output As #f03
05,433           Close #f03
05,434           Open "Entry.untabbed_wrapped" For Output As #f03
05,435           Close #f03
05,436           Open "Entry" For Binary As #f03
05,437           EntryHead$ = "EntryHead: " + Left$(EntryAll$, InStr(EntryAll$, Chr$(13)) - 1) + Chr$(13) + Chr$(10)
05,438           Put #f03, , EntryHead$
05,439           EntryTail$ = "EntryTail: " + Right$(EntryAll$, Len(EntryAll$) - InStr(EntryAll$, Chr$(9)))
05,440           Put #f03, , EntryTail$
05,441           Close #f03
05,442           'Creating the wrapped temporary file [
05,443           Open "Entry" For Binary As #f03
05,444           f04 = FreeFile
05,445           Open "Entry.untabbed_wrapped" For Output As #f04
05,446
05,447           Wwidth% = DjamPoolLength
05,448           WrappedLines = 0
05,449           Do While Not EOF(f03)
05,450             Line Input #f03, l$
05,451             ExpandTabsFULL l$
05,452             ' Firstly do the wrapping virtually (in order to avoid writing some wrapped chunks and encounter unwrappable chunk) - we need either a wrapped line (in its entirety) or none:
05,453             AssumeLineIsWrappable = 1
05,454             lX$ = l$
05,455             Do While Len(lX$) > Wwidth%
05,456               Glupak% = Wwidth%
05,457               Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak% + 1, 1) = ";")
OR MID$(lX$, Glupak% + 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "
05,458               InvokeOnce$ = Mid$(lX$, Glupak% + 1, 1)
05,459               If InStr("\/_;.-!+%=:", InvokeOnce$) And Mid$(lX$, Glupak%, 1) <> " " Then Exit Do
05,460               'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = ";" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR
InvokeOnce$ = "!") OR InvokeOnce$ = "+" OR InvokeOnce$ = "%") AND MID$(lX$, Glupak%, 1) <> " " THEN EXIT DO
05,461               Glupak% = Glupak% + 1
05,462               If Glupak% = 0 Then

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,463             AssumeLineIsWrappable = 0
05,464             GoTo B4TxpanarVIRTUAL3
05,465         End If
05,466         Loop
05,467         lX$ = LTrim$(Mid$(lX$, Glupak% + 1, Len(lX$) - (Glupak%)))
05,468     Loop
05,469     B4TxpanarVIRTUAL3:
05,470     If AssumeLineIsWrappable = 1 Then
05,471         lX$ = l$
05,472         Do While Len(lX$) > Wwidth%
05,473             Glupak% = Wwidth%
05,474             Do 'UNTIL (MID$(lX$, Glupak% + 1, 1) = " " OR MID$(lX$, Glupak% + 1, 1) = "\" OR MID$(lX$, Glupak% + 1, 1) = "/" OR MID$(lX$, Glupak% + 1, 1) = "_" OR MID$(lX$, Glupak% + 1, 1) =
";" OR MID$(lX$, Glupak% + 1, 1) = ",") AND MID$(lX$, Glupak%, 1) <> " "
05,475                 InvokeOnce$ = Mid$(lX$, Glupak% + 1, 1)
05,476                 If InStr(" \_ ; , . - ! + % = : ", InvokeOnce$) And Mid$(lX$, Glupak%, 1) <> " " Then Exit Do
05,477                 'IF (InvokeOnce$ = " " OR InvokeOnce$ = "\" OR InvokeOnce$ = "/" OR InvokeOnce$ = "_" OR InvokeOnce$ = ";" OR InvokeOnce$ = "," OR InvokeOnce$ = "." OR InvokeOnce$ = "-" OR
InvokeOnce$ = "! " OR InvokeOnce$ = "+ " OR InvokeOnce$ = "%") AND MID$(lX$, Glupak%, 1) <> " " THEN EXIT DO
05,478                 Glupak% = Glupak% + 1
05,479                 If Glupak% = 0 Then
05,480                     Beep
05,481                     GoTo B4Txpanar3
05,482                 End If
05,483             Loop
05,484             Print #f04, Left$(lX$, Glupak%): WrappedLines = WrappedLines + 1
05,485             'lX$ = STRING$(1, " ") + LTRIM$(MID$(lX$, Glupak% + 1, LEN(lX$) - (Glupak%)))
05,486             lX$ = LTrim$(Mid$(lX$, Glupak% + 1, Len(lX$) - (Glupak%)))
05,487         Loop
05,488         Print #f04, lX$: WrappedLines = WrappedLines + 1
05,489         B4Txpanar3:
05,490     Else
05,491         Beep
05,492     End If
05,493 Loop
05,494 Close #f03, #f04
05,495 'Creating the wrapped temporary file ]
05,496
05,497 ReDim Djamtc$(1 To WrappedLines)
05,498 Open "Entry.untabbed_wrapped" For Input As #f04
05,499 jj = 1
05,500 Do While Not EOF(f04)
05,501     Line Input #f04, Djamtc$(jj): jj = jj + 1
05,502 Loop
05,503 Close #f04
05,504
05,505 For i = 1 To DjamPoolHeight
05,506     Locate 3 + 3 + 2 + (i - 1), 3 + 2
05,507     Print String$(DjamPoolLength, " ");
05,508 Next
05,509

```

```

05,510 CurrentPos2& = 1 ' Notice that CurrentPos2& = 1.. (WrappedLines - DjamPoolHeight + 1)
05,511 If WrappedLines < DjamPoolHeight Then CurrentPos2max& = 1 Else CurrentPos2max& = (WrappedLines - DjamPoolHeight + 1)
05,512 If CurrentPos2& > CurrentPos2max& Then CurrentPos2& = CurrentPos2max&
05,513 PoolLinePos2 = 1
05,514 Do
05,515     key$ = InKey$
05,516     'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
05,517     If key$ <> "" Then
05,518         code% = Asc(key$):
05,519         If code% Then ' ASC returns any value greater than 0
05,520             Else
05,521                 Select Case Asc(key$, 2)
05,522                     Case 72:
05,523                         If PoolLinePos2 > 1 Then
05,524                             PoolLinePos2 = PoolLinePos2 - 1
05,525                         Else
05,526                             If CurrentPos2& > 1 Then CurrentPos2& = CurrentPos2& - 1
05,527                         End If
05,528                     Case 80:
05,529                         If PoolLinePos2 < MIN$(DjamPoolHeight, WrappedLines) Then
05,530                             PoolLinePos2 = PoolLinePos2 + 1
05,531                         Else
05,532                             If CurrentPos2& < CurrentPos2max& Then CurrentPos2& = CurrentPos2& + 1
05,533                         End If
05,534                     Case 75: 'Left
05,535                     Case 77: 'Right
05,536                     Case 71: 'Home
05,537                         PoolLinePos2 = 1
05,538                         CurrentPos2& = 1
05,539                     Case 79: 'End
05,540                         PoolLinePos2 = MIN$(DjamPoolHeight, WrappedLines)
05,541                         CurrentPos2& = CurrentPos2max&
05,542                     Case 73: 'PgUp
05,543                         If CurrentPos2& - DjamPoolHeight >= 1 Then CurrentPos2& = CurrentPos2& - DjamPoolHeight
05,544                     Case 81: 'PgDn
05,545                         If CurrentPos2& + DjamPoolHeight <= CurrentPos2max& Then CurrentPos2& = CurrentPos2& + DjamPoolHeight
05,546                     Case 61: 'F3
05,547                     Case 13: 'Enter
05,548
05,549                 End Select
05,550             End If
05,551         End If
05,552
05,553         _Limit 60
05,554         GoSub Txpo2a
05,555     Loop Until _KeyDown(8)
05,556     Call DrawBoxShadowOED(3 + 3, 3, 35, 125, Reduced$, 2)
05,557     BckGRcolor = 2 '6
05,558     Locate , , 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,559         Case 32:
05,560         Case 65 TO 97: 'PRINT key$;
05,561         Case Asc("a") TO Asc("z"): 'PRINT key$;
05,562         'CASE 27: COLOR 7, 0: SYSTEM 'END
05,563     End Select
05,564 Else
05,565     Select Case Asc(key$, 2)
05,566     Case 72:
05,567         If PoolLinePos > 1 Then
05,568             PoolLinePos = PoolLinePos - 1
05,569         Else
05,570             If CurrentPos& > 1 Then CurrentPos& = CurrentPos& - 1
05,571         End If
05,572     Case 80:
05,573         If PoolLinePos < DjamPoolHeight Then
05,574             PoolLinePos = PoolLinePos + 1
05,575         Else
05,576             If CurrentPos& < (dict02_TotalWrd& - DjamPoolHeight + 1) Then CurrentPos& = CurrentPos& + 1
05,577         End If
05,578     Case 75: 'Left
05,579     Case 77: 'Right
05,580     Case 71: 'Home
05,581         PoolLinePos = 1
05,582         CurrentPos& = 1
05,583     Case 79: 'End
05,584         PoolLinePos = DjamPoolHeight
05,585         CurrentPos& = (dict02_TotalWrd& - DjamPoolHeight + 1)
05,586     Case 73: 'PgUp
05,587         If CurrentPos& - DjamPoolHeight >= 1 Then CurrentPos& = CurrentPos& - DjamPoolHeight
05,588     Case 81: 'PgDn
05,589         If CurrentPos& + DjamPoolHeight <= (dict02_TotalWrd& - DjamPoolHeight + 1) Then CurrentPos& = CurrentPos& + DjamPoolHeight
05,590     Case 61: 'F3
05,591     Case 13: 'Enter
05,592
05,593     End Select
05,594 End If
05,595 End If
05,596
05,597 _Limit 60
05,598 If _KeyDown(13) Then ' Enter
05,599
05,600 End If
05,601 If _KeyDown(18176) Then ' Home
05,602
05,603 End If
05,604 If _KeyDown(20224) Then ' End
05,605
05,606 End If
05,607 If _KeyDown(15616) Then ' F3

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,608
05,609         End If
05,610         If _KeyDown(18688) Then ' PgUp
05,611
05,612         End If
05,613         If _KeyDown(20736) Then ' PgDn
05,614
05,615         End If
05,616
05,617         GoSub Txpo2
05,618         Loop Until _KeyDown(27)
05,619         ' Scrolling into 3D djamtche ]
05,620         Close #f01, #f02
05,621     End If 'check for existance ]]]
05,622     ' KEYCLEAR 2
05,623     Do While InKey$ <> "": Loop
05,624     While _MouseInput: Wend
05,625     Exit Sub
05,626
05,627 Txpo2:
05,628 For i = 1 To MIN$(DjamPoolHeight, dict02_TotalWrd&) ' with dictionaries all are bigger than ~40 lines
05,629     Get #f02, CurrentPos& + (i - 1), dict02_lfixedCRLF$
05,630     EntryAll$ = Space$(dePADzeros&(Mid$(dict02_lfixedCRLF$, (dict02_EntryHead + 1) + 8, 8)))
05,631     Get #f01, dePADzeros&(Mid$(dict02_lfixedCRLF$, (dict02_EntryHead + 1), 8)), EntryAll$
05,632     PADDED_EntryAll$ = Left$(EntryAll$, InStr(EntryAll$, Chr$(13)) - 1) + String$(DjamPoolLength - Len(Left$(EntryAll$, InStr(EntryAll$, Chr$(13)) - 1)), " ")
05,633     Locate 3 + 3 + 2 + (i - 1), 3 + 2
05,634     If i = PoolLinePos Then Color BckGRcolor, 0 Else Color 0, BckGRcolor
05,635     Print PADDED_EntryAll$;
05,636 Next
05,637 s$ = LTrim$(Str$(CurrentPos& + PoolLinePos - 1))
05,638 If Len(s$) > 3 Then
05,639     If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
05,640     s$ = ""
05,641     For i = 1 To Len(x$) Step 3
05,642         s$ = s$ + Mid$(x$, i, 3) + ", "
05,643     Next
05,644     s$ = Left$(s$, Len(s$) - 1)
05,645 End If
05,646 s$ = LTrim$(s$)
05,647 Padded$ = "0,000,000,000"
05,648 Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
05,649 Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
05,650 _Display 'in order to not having frozen screen
05,651 Return
05,652
05,653 Txpo2a:
05,654 For i = 1 To MIN$(DjamPoolHeight, WrappedLines) ' with dictionaries all are bigger than ~40 lines
05,655     PADDED_EntryAll$ = Djamtche$(CurrentPos2& + (i - 1) + String$(DjamPoolLength - Len(Djamtche$(CurrentPos2& + (i - 1))), " ")
05,656     Locate 3 + 3 + 2 + (i - 1), 3 + 2

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,657     If i = PoolLinePos2 Then Color BckGRcolor, 0 Else Color 0, BckGRcolor
05,658     Print PADDED_EntryAll$;
05,659     Next
05,660     s$ = LTrim$(Str$(CurrentPos2& + PoolLinePos2 - 1))
05,661     If Len(s$) > 3 Then
05,662         If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
05,663         s$ = ""
05,664         For i = 1 To Len(x$) Step 3
05,665             s$ = s$ + Mid$(x$, i, 3) + ","
05,666         Next
05,667         s$ = Left$(s$, Len(s$) - 1)
05,668     End If
05,669     s$ = LTrim$(s$)
05,670     Padded$ = "0,000,000,000"
05,671     Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
05,672     Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
05,673     _Display 'in order to not having frozen screen
05,674     Return
05,675
05,676 End Sub
05,677
05,678 Sub dict03sub
05,679     Shared CurrentWord$
05,680     Const dict03_RecordLen = 47 '4+10+31+2 for CRLF
05,681     Dim dict03_lfixedCRLF As String * dict03_RecordLen
05,682     Dim dict03_lfixedCRLFhighlight As String * dict03_RecordLen
05,683     _Define A-Z As _INTEGER64 ' T0-D0: Grr, this very file can be 2+B lines, all & should be removed and be defined all as 8bytes
05,684     dict03_TotalWrd = 0
05,685
05,686     dict03$ = "Schizandrafild_Corpus_revision_D_(45-corpora_-unique-words).sorted"
05,687
05,688     If _FileExists(PSPlike$ + dict03$ + ".ind") = 0 Then 'check for existance [[[
05,689         f01 = FreeFile
05,690         Open dict03$ For Binary As #f01
05,691         If LOF(f01) Then
05,692
05,693             f02 = FreeFile
05,694             Open dict03$ + ".ind" For Binary As #f02 'LEN = 35 + 2 + 8 + 8 '+2 for CRLF and DWORD+DWORD as padded HEX
05,695             Do While Not EOF(f01)
05,696                 Line Input #f01, l$
05,697                 dict03_lfixedCRLF$ = l$ + String$(45 - Len(l$), " ") + Chr$(13) + Chr$(10)
05,698                 Put #f02, , dict03_lfixedCRLF$: dict03_TotalWrd = dict03_TotalWrd + 1
05,699             Loop
05,700             Close #f01, #f02
05,701             'SHELL_HIDE "CMD /C sort " + CHR$(34) + dict03$ + ".ind" + CHR$(34) + " > " + CHR$(34) + "sorted_" + dict03$ + ".ind" + CHR$(34)
05,702
05,703         Else
05,704             $If WINDOWS Then
05,705                 Play "L16V3203g-g-<<d+"

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,706         $End If
05,707         Close #f01
05,708     End If 'If LOF(f01) Then
05,709 Else
05,710     dict03_TotalWrd = 1005624651
05,711 End If 'check for existence ]]]
05,712 If _FileExists(PSPLike$ + dict03$ + ".ind") <> 0 Then 'check for existence [[[
05,713     f02 = FreeFile
05,714     Open PSPLike$ + dict03$ + ".ind" For Random Access Read As #f02 Len = dict03_RecordLen '35 + 2 + 8 + 8 '2 for CRLF and DWORD+DWORD as padded HEX
05,715
05,716     'testing 537 line:
05,717     'alimentary          00025E8D00000118
05,718     'PhysicalLine = 19001 '537
05,719
05,720     ' BINARY SEARCH [
05,721     UCurrentWord$ = LCase$(CurrentWord$)
05,722     WRDpos = 0
05,723     LeftPoint = 1
05,724     RightPoint = dict03_TotalWrd
05,725     Do
05,726         If RightPoint - LeftPoint <= 1 Then 'Sxsedtcheta
05,727             Get #f02, LeftPoint, dict03_lfixedCRLF$
05,728             If UCurrentWord$ = LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then WRDpos = LeftPoint: Exit Do
05,729             Get #f02, RightPoint, dict03_lfixedCRLF$
05,730             If UCurrentWord$ = LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then WRDpos = RightPoint: Exit Do
05,731             WRDpos = LeftPoint 'THIS IS BECAUSE I NEED NEAREST word not zero!
05,732             Exit Do
05,733         End If
05,734         SplitPoint = (LeftPoint + RightPoint) \ 2
05,735         Get #f02, SplitPoint, dict03_lfixedCRLF$
05,736         If UCurrentWord$ < LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then
05,737             RightPoint = SplitPoint
05,738         ElseIf UCurrentWord$ > LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then
05,739             LeftPoint = SplitPoint
05,740         Else
05,741             WRDpos = SplitPoint: Exit Do
05,742         End If
05,743     Loop
05,744     If WRDpos <> 0 Then
05,745         Do While WRDpos > 1
05,746             WRDpos = WRDpos - 1
05,747             Get #f02, WRDpos, dict03_lfixedCRLF$
05,748             If UCurrentWord$ <> LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then
05,749                 WRDpos = WRDpos + 1
05,750             Exit Do
05,751         End If
05,752     Loop
05,753     'IF WRDpos <= TotalWrd& - 16 THEN FL2wrd& = WRDpos ELSE FL2wrd& = TotalWrd& - 16
05,754 End If

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,755 ' BINARY SEARCH ]
05,756
05,757 $If WINDOWS Then
05,758     Play "v20o2l20geb"
05,759 $End If
05,760 PhysicalLine = WRDpos
05,761 'PhysicalLine = 277315 '277315 total; WARNING! 61358 is 'dictionary' - cannot be found because words as 'dicæarch' confuse the binary search!
05,762 'GET #f02, (PhysicalLine - 1) * (35 + 2 + 8 + 8) + 1, dict03_lfixedCRLF$ 'when OPENed for BINARY
05,763 Get #f02, PhysicalLine, dict03_lfixedCRLF$
05,764
05,765 'PRINT "EntryHead: "; LEFT$(EntryAll$, INSTR(EntryAll$, CHR$(13)) - 1)
05,766 'EntryTail$ = "EntryTail: " + RIGHT$(EntryAll$, LEN(EntryAll$) - INSTR(EntryAll$, CHR$(9)))
05,767 'PRINT EntryTail$
05,768 'PRINT "[" + EntryAll$ + "]"
05,769
05,770 ' Scrolling into 3D djamtche [
05,771 'SUB DrawBoxShadow3 (TopLrow, TopLcol, BottomRow, BottomRcol, Captnn$)
05,772 If Len(dict03$) > 70 Then Reduced$ = "..." + Right$(dict03$, 70) Else Reduced$ = dict03$
05,773 Call DrawBoxShadowOED(3 + 3, 3, 35, 125, Reduced$, 6)
05,774 BckGRcolor = 6
05,775 s$ = LTrim$(Str$(PhysicalLine))
05,776 If Len(s$) > 3 Then
05,777     If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
05,778     s$ = ""
05,779     For i = 1 To Len(x$) Step 3
05,780         s$ = s$ + Mid$(x$, i, 3) + ", "
05,781     Next
05,782     s$ = Left$(s$, Len(s$) - 1)
05,783 End If
05,784 s$ = LTrim$(s$)
05,785 Padded$ = "0,000,000,000"
05,786 Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
05,787 Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[" Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
05,788 Locate , , 0: _Display
05,789
05,790 DjamPoolLength = 125 - 3 + 1 - 4
05,791 DjamPoolHeight = 35 - (3 + 3) + 1 - 4
05,792 CurrentPos = PhysicalLine ' Notice that CurrentPos = 1..(dict03_TotalWrd - DjamPoolHeight + 1)
05,793 If CurrentPos > (dict03_TotalWrd - DjamPoolHeight + 1) Then CurrentPos = (dict03_TotalWrd - DjamPoolHeight + 1)
05,794 PoolLinePos = 1
05,795 Do
05,796     key$ = InKey$
05,797     'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
05,798     If key$ <> "" Then
05,799         code% = Asc(key$):
05,800         If code% Then ' ASC returns any value greater than 0
05,801             Select Case Asc(key$)
05,802                 Case 32:
05,803                 Case 65 TO 97: 'PRINT key$;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,804      Case Asc("a") TO Asc("z"): 'PRINT key$;
05,805      'CASE 27: COLOR 7, 0: SYSTEM 'END
05,806      End Select
05,807      Else
05,808      Select Case Asc(key$, 2)
05,809      Case 72:
05,810          If PoolLinePos > 1 Then
05,811              PoolLinePos = PoolLinePos - 1
05,812          Else
05,813              If CurrentPos > 1 Then CurrentPos = CurrentPos - 1
05,814          End If
05,815      Case 80:
05,816          If PoolLinePos < DjamPoolHeight Then
05,817              PoolLinePos = PoolLinePos + 1
05,818          Else
05,819              If CurrentPos < (dict03_TotalWrd - DjamPoolHeight + 1) Then CurrentPos = CurrentPos + 1
05,820          End If
05,821      Case 75: 'Left
05,822      Case 77: 'Right
05,823      Case 71: 'Home
05,824          PoolLinePos = 1
05,825          CurrentPos = 1
05,826      Case 79: 'End
05,827          PoolLinePos = DjamPoolHeight
05,828          CurrentPos = (dict03_TotalWrd - DjamPoolHeight + 1)
05,829      Case 73: 'PgUp
05,830          If CurrentPos - DjamPoolHeight >= 1 Then CurrentPos = CurrentPos - DjamPoolHeight
05,831      Case 81: 'PgDn
05,832          If CurrentPos + DjamPoolHeight <= (dict03_TotalWrd - DjamPoolHeight + 1) Then CurrentPos = CurrentPos + DjamPoolHeight
05,833      Case 61: 'F3
05,834      Case 13: 'Enter
05,835      End Select
05,836      End If
05,837      End If
05,838
05,839      _Limit 60
05,840      If _KeyDown(13) Then ' Enter
05,841
05,842      End If
05,843      If _KeyDown(18176) Then ' Home
05,844
05,845      End If
05,846      If _KeyDown(20224) Then ' End
05,847
05,848      End If
05,849      If _KeyDown(15616) Then ' F3
05,850
05,851      End If
05,852      If _KeyDown(18688) Then ' PgUp

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,853
05,854     End If
05,855     If _KeyDown(20736) Then ' PgDn
05,856
05,857     End If
05,858
05,859     GoSub Txpo3
05,860     Loop Until _KeyDown(27)
05,861     ' Scrolling into 3D djamtche ]
05,862     Close #f02
05,863 End If 'check for existence ]]]
05,864 ' _KEYCLEAR 2
05,865 Do While InKey$ <> "": Loop
05,866 While _MouseInput: Wend
05,867 Exit Sub
05,868
05,869 Txpo3:
05,870 If CurrentPos > (dict03_TotalWrd - DjamPoolHeight + 1) Then ' because dict03_TotalWrd - (dict03_TotalWrd - DjamPoolHeight + 1) + 1 isDjamPoolHeight
05,871     PoolLinePos = CurrentPos - (dict03_TotalWrd - DjamPoolHeight + 1)
05,872 End If
05,873 Get #f02, CurrentPos + (PoolLinePos - 1), dict03_lfixedCRLFhighlight$ 'we need this in order to highlight (lowering the strain on the eye) all such words
05,874 For i = 1 To MIN$(DjamPoolHeight, dict03_TotalWrd) ' with dictionaries all are bigger than ~40 lines
05,875     Get #f02, CurrentPos + (i - 1), dict03_lfixedCRLF$
05,876     Locate 3 + 3 + 2 + (i - 1), 3 + 2
05,877     If i = PoolLinePos Then
05,878         Color BckGRcolor, 0
05,879     Else
05,880         If UCurrentWord$ = LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then Color 7, BckGRcolor Else Color 0, BckGRcolor
05,881         If LCase$(RTrim$(Mid$(dict03_lfixedCRLFhighlight$, 15, 31))) = LCase$(RTrim$(Mid$(dict03_lfixedCRLF$, 15, 31))) Then Color 15, BckGRcolor
05,882     End If
05,883     'PRINT LEFT$(dict03_lfixedCRLF$, INSTR(dict03_lfixedCRLF$, CHR$(13)) - 1) + STRING$(DjamPoolLength - LEN(LEFT$(dict03_lfixedCRLF$, INSTR(dict03_lfixedCRLF$, CHR$(13)) - 1)), " ");
05,884     For j = 1 To 45
05,885         If Left$(Schizandra$(j), 3) = Left$(dict03_lfixedCRLF$, 3) Then
05,886             addon$ = RTrim$(Right$(Schizandra$(j), Len(Schizandra$(j)) - 4))
05,887             PaddingLEN = DjamPoolLength - Len(Left$(dict03_lfixedCRLF$, Instr(dict03_lfixedCRLF$, Chr$(13)) - 1))
05,888             If Len(addon$) <= PaddingLEN Then
05,889                 addon$ = addon$ + String$(PaddingLEN - Len(addon$), " ")
05,890             Else
05,891                 addon$ = Left$(addon$, PaddingLEN)
05,892             End If
05,893             Print Left$(dict03_lfixedCRLF$, Instr(dict03_lfixedCRLF$, Chr$(13)) - 1) + addon$;
05,894         End If
05,895     Next
05,896 Next
05,897 s$ = LTrim$(Str$(CurrentPos + PoolLinePos - 1))
05,898 If Len(s$) > 3 Then
05,899     If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
05,900     s$ = ""
05,901     For i = 1 To Len(x$) Step 3

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,902      s$ = s$ + Mid$(x$, i, 3) + ","
05,903      Next
05,904      s$ = Left$(s$, Len(s$) - 1)
05,905      End If
05,906      s$ = LTrim$(s$)
05,907      Padded$ = "0,000,000,000"
05,908      Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
05,909      Color 0, BckGRcolor: Locate 35, 3 + 97: Print "[ Entry: "; Color 7, BckGRcolor: Print Padded$; Color 0, BckGRcolor: Print " ]";
05,910      _Display 'in order to not having frozen screen
05,911      Return
05,912 End Sub
05,913
05,914 Function PADzeros$ (hexstr$)
05,915     PADzeros$ = String$(8 - Len(hexstr$), "0") + hexstr$
05,916 End Function
05,917 Function dePADzeros& (hexstr$)
05,918     dePADzeros& = Val("&H" + hexstr$)
05,919 End Function
05,920
05,921 Sub DrawBoxShadowOED (TopLrow, TopLcol, BottomRow, BottomRcol, Captmn$, BckGRcolor)
05,922     Shared FileSize
05,923     Shared LoadedFile$
05,924     Shared PSPLike$
05,925     Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$( ), FileArray$( )
05,926     'Shadow
05,927     Color 8, BACKGR
05,928     For i = 1 To MIN$(YdimROW, filecount)
05,929         'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
05,930         'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN
05,931             '    FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
05,932         'ELSE
05,933             '    FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
05,934         'END IF
05,935         If i >= TopLrow + 1 And i <= BottomRow + 1 Then
05,936             Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
05,937         End If
05,938     Next
05,939     'Outer frame
05,940     Locate TopLrow, TopLcol
05,941     Color 7, BckGRcolor
05,942     Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));
05,943     Color 0, BckGRcolor: Print Chr$(191);
05,944     Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ Dictionary: "; Color 15, BckGRcolor: Print Captmn$; Color 7, BckGRcolor: Print " ]";
05,945     For i = TopLrow + 1 To BottomRow - 1
05,946         Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32)); Color 0, BckGRcolor: Print Chr$(179);
05,947     Next
05,948     Locate BottomRow, TopLcol
05,949     Color 7, BckGRcolor: Print Chr$(192);
05,950     Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

05,951 'Inner frame
05,952 'In case of YdimROW = 40 then shrunk panel
05,953 shrunkP = 0
05,954 Locate TopLrow + 1, TopLcol + 1
05,955 Color 0, BckGRcolor
05,956 Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
05,957 Color 7, BckGRcolor: Print Chr$(191);
05,958 For i = TopLrow + 2 To BottomRow - 1
05,959     Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32)); Color 7, BckGRcolor: Print Chr$(179);
05,960 Next
05,961 Locate BottomRow - 1, TopLcol + 1
05,962 Color 0, BckGRcolor: Print Chr$(192);
05,963 Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)
05,964
05,965 Locate TopLrow + 2, TopLcol + 2
05,966 Color 7, BckGRcolor
05,967
05,968 HelpLine$ = "Up,Dn,PgUp,PgDn,Home,End,Enter,Esc,F3"
05,969 Color 0, BckGRcolor: Locate BottomRow, TopLcol + 2: Print "[ Keys: "; Color 7, BckGRcolor: Print HelpLine$; Color 0, BckGRcolor: Print " ]";
05,970 Color 0, BckGRcolor: Locate BottomRow, TopLcol + 2 + 49: Print "[ Jump: "; Color 7, BckGRcolor: Print "ForwardToFirstLineMatchingThisText"; Color 0, BckGRcolor: Print " ]";
05,971 _Display
05,972 'Match$ = InputLine$(TopLrow + 39 - 23 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
05,973 Locate 1, 1, 1, CursorS, CursorE
05,974 End Sub
05,975
05,976 Sub DrawBoxShadowOEDin (TopLrow, TopLcol, BottomRow, BottomRcol, Captnn$)
05,977 Shared FileSize
05,978 Shared LoadedFile$
05,979 Shared PSPlike$
05,980 Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$( ), FileArray$( )
05,981 'Shadow
05,982 BckGRcolor = 2
05,983 Color 8, BACKGR
05,984 For i = 1 To MIN$(YdimROW, filecount)
05,985     'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
05,986     'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN
05,987         'FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
05,988     'ELSE
05,989         'FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
05,990     'END IF
05,991     If i >= TopLrow + 1 And i <= BottomRow + 1 Then
05,992         Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
05,993     End If
05,994 Next
05,995 'Outer frame
05,996 Locate TopLrow, TopLcol
05,997 Color 7, BckGRcolor
05,998 Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));
05,999 Color 0, BckGRcolor: Print Chr$(191);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,000 Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ Dictionary: ";: Color 15, BckGRcolor: Print Captn$;: Color 7, BckGRcolor: Print " ]";
06,001 For i = TopLrow + 1 To BottomRow - 1
06,002     Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32));: Color 0, BckGRcolor: Print Chr$(179);
06,003 Next
06,004 Locate BottomRow, TopLcol
06,005 Color 7, BckGRcolor: Print Chr$(192);
06,006 Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)
06,007 'Inner frame
06,008 'In case of YdimROW = 40 then shrunk panel
06,009 shrunkP = 0
06,010 Locate TopLrow + 1, TopLcol + 1
06,011 Color 0, BckGRcolor
06,012 Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
06,013 Color 7, BckGRcolor: Print Chr$(191);
06,014 For i = TopLrow + 2 To BottomRow - 1
06,015     Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32));: Color 7, BckGRcolor: Print Chr$(179);
06,016 Next
06,017 Locate BottomRow - 1, TopLcol + 1
06,018 Color 0, BckGRcolor: Print Chr$(192);
06,019 Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)
06,020
06,021 Locate TopLrow + 2, TopLcol + 2
06,022 Color 7, BckGRcolor
06,023
06,024 HelpLine2$ = "Up,Dn,PgUp,PgDn,Home,End,Backspace,F3"
06,025 Color 0, BckGRcolor: Locate BottomRow, TopLcol + 2: Print "[ Keys: ";: Color 7, BckGRcolor: Print HelpLine2$;: Color 0, BckGRcolor: Print " ]";
06,026 Color 0, BckGRcolor: Locate BottomRow, TopLcol + 2 + 49: Print "[ Jump: ";: Color 7, BckGRcolor: Print "ForwardToFirstLineMatchingThisText";: Color 0, BckGRcolor: Print " ]";
06,027 _Display
06,028 'Match$ = InputLine$(TopLrow + 39 - 23 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
06,029 Locate 1, 1, 1, CursorS, CursorE
06,030 End Sub
06,031
06,032
06,033 'mouse mumbo-jumbo [
06,034
06,035 'SCREEN _NEWIMAGE(600, 900, 256)
06,036 '_SCREENMOVE 32, 32
06,037
06,038 'ShutDownDuration# = 4 ' 4 seconds
06,039 'DclickTime# = 0.33 ' 1/3 of a second, usually it is 0.27s, so 0.33 suits even the slow clickers
06,040 'DIM Button1LOG_firstDetection#(4)
06,041 'DIM Button1LOG_ForHowLongHoled#(4)
06,042 'PrevClick1# = 0
06,043 'PRINT "You may exit with Esc or by holding left mouse button for 4 seconds..."
06,044 'DO WHILE INKEY$ <> CHR$(27)
06,045
06,046 '     AsIfItIsINKEY% = _MOUSEINPUT '         Check the mouse status
06,047 '     IF AsIfItIsINKEY% THEN
06,048 '         buttowndown1 = _MOUSEBUTTON(1)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,049 '      buttondown2 = _MOUSEBUTTON(2)
06,050 '      buttondown3 = _MOUSEBUTTON(3)
06,051 '      mwheel = _MOUSEWHEEL
06,052 '  END IF
06,053
06,054 '  IF (buttondown1 AND PrevClick1# = 0) THEN 'first detection of the click
06,055 '    PrevClick1# = TIMER(0.001)
06,056 '  END IF
06,057
06,058 '  IF (buttondown1 AND PrevClick1# <> 0) THEN 'already clicked
06,059 '    PrevClick1# = TIMER(0.001)
06,060 '    ForHowLongWasPressed1# = TIMER(0.001) - PrevClick1#
06,061 '  END IF
06,062
06,063 '  IF buttondown1 = 0 AND PrevClick1# <> 0 THEN 'write to the log
06,064 '    PRINT
06,065 '    FOR MumboJumbo = 1 TO 4
06,066 '      PRINT "OLD"; Button1LOG_firstDetection#(MumboJumbo), Button1LOG_ForHowLongHoded#(MumboJumbo)
06,067 '    NEXT
06,068
06,069 '    FOR MumboJumbo = 1 TO 3
06,070 '      Button1LOG_firstDetection#(MumboJumbo) = Button1LOG_firstDetection#(MumboJumbo + 1)
06,071 '      Button1LOG_ForHowLongHoded#(MumboJumbo) = Button1LOG_ForHowLongHoded#(MumboJumbo + 1)
06,072 '    NEXT
06,073 '    Button1LOG_firstDetection#(4) = PrevClick1#
06,074 '    Button1LOG_ForHowLongHoded#(4) = ForHowLongWasPressed1#
06,075 '    PrevClick1# = 0
06,076
06,077 '    FOR MumboJumbo = 1 TO 4
06,078 '      PRINT "NEW"; Button1LOG_firstDetection#(MumboJumbo), Button1LOG_ForHowLongHoded#(MumboJumbo)
06,079 '    NEXT
06,080
06,081 '  END IF
06,082
06,083 '  IF Button1LOG_firstDetection#(3) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3) < DclickTime#) THEN Double_LeftClick = 1 ELSE Double_LeftClick = 0
06,084 '  IF Double_LeftClick = 1 THEN PRINT "Double_LeftClick detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3)) * 1000); "ms."
06,085
06,086 '  IF Button1LOG_firstDetection#(2) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(2) < DclickTime# * 2) THEN Triple_LeftClick = 1 ELSE Triple_LeftClick = 0
06,087 '  IF Triple_LeftClick = 1 THEN PRINT "Triple_LeftClick detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(2)) * 1000); "ms."
06,088
06,089 '  IF Button1LOG_firstDetection#(1) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(1) < DclickTime# * 3) THEN Quadruple_LeftClick = 1 ELSE Quadruple_LeftClick = 0
06,090 '  IF Quadruple_LeftClick = 1 THEN PRINT "Quadruple_LeftClick detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(1)) * 1000); "ms."
06,091
06,092 '  IF (ForHowLongWasPressed1# > ShutDownDuration#) THEN
06,093 '    ShutDown_LeftClick = 1: IF ShutDown_LeftClick = 1 THEN PRINT "ShutDown_LeftClick detected.": END
06,094 '  END IF
06,095
06,096 '  _LIMIT 500
06,097 '  _DISPLAY

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,098 'LOOP
06,099
06,100 ''Note1: From above experiments I see Double-Click fits usually in DclickTime# s window.
06,101 ''Note2: Double-Click is detected when the difference between two initial clicks is below DclickTime# s i.e. Button1LOG_firstDetection#(4)-Button1LOG_firstDetection#(3) < DclickTime#
06,102 '' ForHowLongWasPressed1# = B1 - A1 i.e. duration of holding down:
06,103 '' where
06,104 '' A1=first detection of the click   B1=released A2=first detection of the click   B2=released
06,105 '' [                               ][                               ][                               ]
06,106
06,107 ''mouse mumbo-jumbo ]
06,108
06,109 ' The need for more ergonomic (that is, easily accessible) shortcuts (strictly keyboardish) got me thinking...
06,110 ' Holy hell, why no one introduced the analogues/counterparts of "double-clicks" - the first that comes to mind: the "double-hits".
06,111 ' Both, being just taps.
06,112 ' Currently, in Masakari there are:
06,113 ' - shortcuts, purely keyboardish;
06,114 ' - shortcuts, mix of keys and mouse buttons;
06,115 ' - shortcuts, purely mouseish, just mouse stuff - buttons and wheel.
06,116 ' From revision 8.1+ onward, following double-hits were implemented:
06,117 ' Double LShift; Double LCtrl; Double LAlt; Double RAlt.
06,118 ' The thing is that "legacy" shortcuts as Ctrl+F3 (even the F3) force the user to "locate" their position and to overreach - meaning hovering pass the SPACEBAR area - which is most easy to hit since the wrists/palms
have support, and are simply closer.
06,119 ' To me, the traditional key sequences are to be replaced/enriched with more convenient, more close to the first row of the keyboard, ones.
06,120 ' Currently, my main keyboard is a nasty one (laptopish, with low profile, non-tactile, half-sized F-keys, and tightly packed), when e.g. a LCtrl+F3 search is needed, spotting this halved F3 is a drag, therefore the
double-hit of e.g. LCtrl saves the situation - just double-tap it - the timings between the two pressings are handled by the same logic as the mouse double/triple/quadruple-clicks.
06,121 ' The following stand-alone etude is the same as the previously shared mouse_mumbo-jumbo.bas, except:
06,122 ' buttowndown1 = _MOUSEBUTTON(1)
06,123 ' should be replaced with
06,124 ' KeyTap_buttowndown1 = _KEYDOWN(LSHIFTkey&)
06,125
06,126 ''keyboard mumbo-jumbo [
06,127
06,128 'CONST RSHIFTkey& = 100303
06,129 'CONST LSHIFTkey& = 100304
06,130 'CONST RCTRLkey& = 100305
06,131 'CONST LCTRLkey& = 100306
06,132 'CONST RALTkey& = 100307
06,133 'CONST LALTkey& = 100308
06,134
06,135 'CONST BACKSPCkey& = 8
06,136 'CONST TABkey& = 9
06,137 'CONST SPACEkey& = 32
06,138 'CONST ESCkey& = 27
06,139 'CONST ENTERkey& = 13
06,140
06,141 'CONST HOMEkey& = 18176
06,142 'CONST ENDkey& = 20224
06,143
06,144 'CONST INSkey& = 20992

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,145 'CONST DELkey& = 21248
06,146
06,147 'CONST PGUPkey& = 18688
06,148 'CONST PGDNkey& = 20736
06,149
06,150 'CONST LEFTkey& = 19200
06,151 'CONST RIGHTkey& = 19712
06,152 'CONST UPkey& = 18432
06,153 'CONST DOWNkey& = 20480
06,154
06,155 'SCREEN _NEWIMAGE(600, 900, 256)
06,156 '_SCREENMOVE 32, 32
06,157
06,158 'ShutDownDuration# = 4 ' 4 seconds
06,159 'DclickTime# = 0.33 ' 1/3 of a second, usually it is 0.27s, so 0.33 suits even the slow clickers
06,160 'DIM Button1LOG_firstDetection#(4)
06,161 'DIM Button1LOG_ForHowLongHoled#(4)
06,162 'PrevClick1# = 0
06,163 'PRINT "You may exit with Esc or by holding left Shift key for 4 seconds..."
06,164 'DO 'WHILE INKEY$ <> CHR$(27)
06,165
06,166 '     'AsIfItIsINKEY% = _MOUSEINPUT '     Check the mouse status
06,167 '     IF AsIfItIsINKEY% THEN
06,168 '         'buttondown1 = _MOUSEBUTTON(1)
06,169 '         'buttondown2 = _MOUSEBUTTON(2)
06,170 '         'buttondown3 = _MOUSEBUTTON(3)
06,171 '         'mwheel = _MOUSEWHEEL
06,172 '     END IF
06,173
06,174 '     buttondown1 = _KEYDOWN(LSHIFTkey&)
06,175 '     buttondown2 = _KEYDOWN(LCTRLkey&)
06,176 '     buttondown3 = _KEYDOWN(LALTkey&)
06,177
06,178 '     IF (buttondown1 AND PrevClick1# = 0) THEN 'first detection of the click
06,179 '         PrevClick1# = TIMER(0.001)
06,180 '     END IF
06,181
06,182 '     IF (buttondown1 AND PrevClick1# <> 0) THEN 'already clicked
06,183 '         '         PrevClick1# = TIMER(0.001)
06,184 '         ForHowLongWasPressed1# = TIMER(0.001) - PrevClick1#
06,185 '     END IF
06,186
06,187 '     IF buttondown1 = 0 AND PrevClick1# <> 0 THEN 'write to the log
06,188 '         PRINT
06,189 '         FOR MumboJumbo = 1 TO 4
06,190 '             PRINT "OLD"; Button1LOG_firstDetection#(MumboJumbo), Button1LOG_ForHowLongHoled#(MumboJumbo)
06,191 '         NEXT
06,192
06,193 '         FOR MumboJumbo = 1 TO 3

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,194 '      Button1LOG_firstDetection#(MumboJumbo) = Button1LOG_firstDetection#(MumboJumbo + 1)
06,195 '      Button1LOG_ForHowLongHolded#(MumboJumbo) = Button1LOG_ForHowLongHolded#(MumboJumbo + 1)
06,196 '      NEXT
06,197 '      Button1LOG_firstDetection#(4) = PrevClick1#
06,198 '      Button1LOG_ForHowLongHolded#(4) = ForHowLongWasPressed1#
06,199 '      PrevClick1# = 0
06,200
06,201 '      FOR MumboJumbo = 1 TO 4
06,202 '          PRINT "NEW"; Button1LOG_firstDetection#(MumboJumbo), Button1LOG_ForHowLongHolded#(MumboJumbo)
06,203 '      NEXT
06,204
06,205 '  END IF
06,206
06,207 '  IF Button1LOG_firstDetection#(3) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3) < DclickTime#) THEN Double_LeftClick = 1 ELSE Double_LeftClick = 0
06,208 '  IF Double_LeftClick = 1 THEN PRINT "Double_LeftShift detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(3)) * 1000); "ms."
06,209
06,210 '  IF Button1LOG_firstDetection#(2) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(2) < DclickTime# * 2) THEN Triple_LeftClick = 1 ELSE Triple_LeftClick = 0
06,211 '  IF Triple_LeftClick = 1 THEN PRINT "Triple_LeftShift detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(2)) * 1000); "ms."
06,212
06,213 '  IF Button1LOG_firstDetection#(1) AND (Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(1) < DclickTime# * 3) THEN Quadruple_LeftClick = 1 ELSE Quadruple_LeftClick = 0
06,214 '  IF Quadruple_LeftClick = 1 THEN PRINT "Quadruple_LeftShift detected. Done in"; INT((Button1LOG_firstDetection#(4) - Button1LOG_firstDetection#(1)) * 1000); "ms."
06,215
06,216 '  IF (ForHowLongWasPressed1# > ShutDownDuration#) THEN
06,217 '      ShutDown_LeftClick = 1: IF ShutDown_LeftClick = 1 THEN PRINT "ShutDown_LeftShift detected.": END
06,218 '  END IF
06,219
06,220 '  _LIMIT 500
06,221 '  _DISPLAY
06,222 'LOOP
06,223
06,224 'Note1: From above experiments I see Double-Click fits usually in DclickTime# s window.
06,225 'Note2: Double-Click is detected when the difference between two initial clicks is below DclickTime# s i.e. Button1LOG_firstDetection#(4)-Button1LOG_firstDetection#(3) < DclickTime#
06,226 'ForHowLongWasPressed1# = B1 - A1 i.e. duration of holding down:
06,227 'where
06,228 'A1=first detection of the click  B1=released A2=first detection of the click  B2=released
06,229 '[           ][           ][           ]
06,230
06,231 'keyboard mumbo-jumbo ]
06,232
06,233 Function Hex2$(SOMEkey)
06,234   If Len(Hex$(SOMEkey)) = 1 Then Hex2$ = "0" + Hex$(SOMEkey) Else Hex2$ = Hex$(SOMEkey)
06,235 End Function
06,236
06,237 Function EpochTime"&&
06,238   DATEstamp$ = Date$: TIMEstamp$ = Time$
06,239   Do While DATEstamp$ <> Date$ 'redo
06,240       DATEstamp$ = Date$: TIMEstamp$ = Time$
06,241   Loop
06,242   m = Val(Left$(DATEstamp$, 2))

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,243 d~&& = Val(Mid$(DATEstamp$, 4, 2))
06,244 y~&& = Val(Right$(DATEstamp$, 4)) - 1970
06,245 For i = 1 To m
06,246     Select Case i 'Add the number of days for each previous month passed
06,247         Case 1: d~&& = d~&& 'January doesn't have any carry over days.
06,248         Case 2, 4, 6, 8, 9, 11: d~&& = d~&& + 31
06,249         Case 3: d~&& = d~&& + 28
06,250         Case 5, 7, 10, 12: d~&& = d~&& + 30
06,251     End Select
06,252 Next
06,253 d~&& = d~&& + y~&& * 365
06,254 For i = 2 To y Step 4
06,255     If m > 2 Then d~&& = d~&& + 1 'add an extra day for leap year every 4 years, starting in 1970
06,256 Next
06,257 d~&& = d~&& - 1 'for year 2000
06,258 EpochTime~&& = d~&& * 24 * 60 * 60 + Val(Left$(TIMEstamp$, 2)) * 3600 + Val(Mid$(TIMEstamp$, 4, 2)) * 60 + Val(Right$(TIMEstamp$, 2))
06,259 End Function
06,260
06,261 Sub ReportSortingLength (posit)
06,262     Shared YdimROW, XdimCOL
06,263     Shared TimeA, TimeB
06,264     crx = Pos(0)
06,265     cry = CsrLin
06,266     ' P=6
06,267     '[123456] (8-P)-1=2 ' 2021-Jul-26, ugh, the fix is without '-1' because P is not 5 but 6!
06,268     Locate YdimROW + 1, posit: Color 9, 0: Print Space$((XdimCOL - posit) - 0);
06,269     Locate YdimROW + 1, posit: Color 9, 0: Print "; Sorting ... by Length";
06,270     Locate cry, crx, 1, CursorS, CursorE
06,271 End Sub
06,272
06,273 Sub ReportSortingOffset (posit)
06,274     Shared YdimROW, XdimCOL
06,275     Shared TimeA, TimeB
06,276     crx = Pos(0)
06,277     cry = CsrLin
06,278     ' P=6
06,279     '[123456] (8-P)-1=2 ' 2021-Jul-26, ugh, the fix is without '-1' because P is not 5 but 6!
06,280     Locate YdimROW + 1, posit: Color 9, 0: Print Space$((XdimCOL - posit) - 0);
06,281     Locate YdimROW + 1, posit: Color 9, 0: Print "; Sorting ... by Offset";
06,282     Locate cry, crx, 1, CursorS, CursorE
06,283 End Sub
06,284
06,285 ' Written by Sanmayce, 2021-Oct-29
06,286 ' The indexes are signed, but the elements are unsigned.
06,287 Define A-Z As _INTEGER64
06,288 Sub Quicksort_QB64 (QWORDS~&&())
06,289     Left = LBound(QWORDS~&&)
06,290     Right = UBound(QWORDS~&&)
06,291     LeftMargin = Left

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,292 ReDim Stack&&(Left To Right)
06,293 StackPtr = 0
06,294 StackPtr = StackPtr + 1
06,295 Stack&&(StackPtr + LeftMargin) = Left
06,296 StackPtr = StackPtr + 1
06,297 Stack&&(StackPtr + LeftMargin) = Right
06,298 Do 'Until StackPtr = 0
06,299     Right = Stack&&(StackPtr + LeftMargin)
06,300     StackPtr = StackPtr - 1
06,301     Left = Stack&&(StackPtr + LeftMargin)
06,302     StackPtr = StackPtr - 1
06,303 Do 'Until Left >= Right
06,304     Pivot&& = QWORDS&&((Left + Right) \ 2)
06,305     Indx = Left
06,306     Jndx = Right
06,307     Do
06,308         Do While (QWORDS&&(Indx) < Pivot&&)
06,309             Indx = Indx + 1
06,310         Loop
06,311         Do While (QWORDS&&(Jndx) > Pivot&&)
06,312             Jndx = Jndx - 1
06,313         Loop
06,314         If Indx <= Jndx Then
06,315             If Indx < Jndx Then Swap QWORDS&&(Indx), QWORDS&&(Jndx)
06,316             Indx = Indx + 1
06,317             Jndx = Jndx - 1
06,318         End If
06,319     Loop While Indx <= Jndx
06,320     If Indx < Right Then
06,321         StackPtr = StackPtr + 1
06,322         Stack&&(StackPtr + LeftMargin) = Indx
06,323         StackPtr = StackPtr + 1
06,324         Stack&&(StackPtr + LeftMargin) = Right
06,325     End If
06,326     Right = Jndx
06,327 Loop Until Left >= Right
06,328 Loop Until StackPtr = 0
06,329 End Sub
06,330
06,331 ' This works only under Windows [::::::::::::::::::::::::::::::::::::::::::::]
06,332
06,333 'Declare CustomType Library
06,334 '    Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
06,335 '    Function memcpy% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
06,336 'End Declare
06,337 'Dim Benchm As _MEM
06,338 'Elements = 10
06,339 'Granularity = 17
06,340 'Benchm = _MemNew(Elements * Granularity)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,341 'For i = 9 To 0 Step -1
06,342 '    Pattern$ = String$(16, Chr$(Asc("0") + i)) + Chr$(0)
06,343 '    Print Pattern$
06,344 '    _MemPut Benchm, Benchm.OFFSET + Granularity * (9 - i), Pattern$
06,345 'Next
06,346 'Print: Print "Sorting...": Print
06,347 'Quicksort_QB64_Granularity Benchm, Elements, Granularity
06,348 'For i = 0 To 9
06,349 '    _MemGet Benchm, Benchm.OFFSET + Granularity * i, Pattern$
06,350 '    Print Pattern$
06,351 'Next
06,352 'End
06,353
06,354 ' Written by Sanmayce, 2021-Dec-12
06,355 ' The indexes are signed, but the elements are unsigned.
06,356 'Define A-Z As _INTEGER64
06,357 'Sub Quicksort_QB64_Granularity (MemBlock As _MEM, NumberOfElements As _Integer64, Granularity As Integer)
06,358 '    Left = 1 - 1
06,359 '    Right = NumberOfElements - 1
06,360 '    ReDim Stack&&(Left To Right)
06,361 '    SwapBuffer$ = Space$(Granularity)
06,362 '    Pivot$ = Space$(Granularity)
06,363 '    StackPtr = 0
06,364 '    StackPtr = StackPtr + 1
06,365 '    Stack&&(StackPtr) = Left
06,366 '    StackPtr = StackPtr + 1
06,367 '    Stack&&(StackPtr) = Right
06,368 '    Do 'Until StackPtr = 0
06,369 '        Right = Stack&&(StackPtr)
06,370 '        StackPtr = StackPtr - 1
06,371 '        Left = Stack&&(StackPtr)
06,372 '        StackPtr = StackPtr - 1
06,373 '        Do 'Until Left >= Right
06,374 '            'Pivot~&& = QWORDS~&&((Left + Right) \ 2)
06,375 '            memcpy _Offset(Pivot$), MemBlock.OFFSET + Granularity * ((Left + Right) \ 2), Granularity
06,376 '            Indx = Left
06,377 '            Jndx = Right
06,378 '            Do
06,379 '                'Do While (QWORDS~&&(Indx) < Pivot~&&)
06,380 '                Do While (memcmp%(MemBlock.OFFSET + Granularity * Indx, _Offset(Pivot$), Granularity) = -1)
06,381 '                    Indx = Indx + 1
06,382 '                Loop
06,383 '                'Do While (QWORDS~&&(Jndx) > Pivot~&&)
06,384 '                Do While (memcmp%(MemBlock.OFFSET + Granularity * Jndx, _Offset(Pivot$), Granularity) = 1)
06,385 '                    Jndx = Jndx - 1
06,386 '                Loop
06,387 '                If Indx <= Jndx Then
06,388 '                    If Indx < Jndx Then
06,389 '                        'Swap QWORDS~&&(Indx), QWORDS~&&(Jndx)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,390 '      memcpy _Offset(SwapBuffer$), MemBlock.OFFSET + Granularity * Jndx, Granularity
06,391 '      memcpy MemBlock.OFFSET + Granularity * Jndx, MemBlock.OFFSET + Granularity * Indx, Granularity
06,392 '      memcpy MemBlock.OFFSET + Granularity * Indx, _Offset(SwapBuffer$), Granularity
06,393 '      End If
06,394 '      Indx = Indx + 1
06,395 '      Jndx = Jndx - 1
06,396 '      End If
06,397 '      Loop While Indx <= Jndx
06,398 '      If Indx < Right Then
06,399 '          StackPtr = StackPtr + 1
06,400 '          Stack&&(StackPtr) = Indx
06,401 '          StackPtr = StackPtr + 1
06,402 '          Stack&&(StackPtr) = Right
06,403 '      End If
06,404 '      Right = Jndx
06,405 '      Loop Until Left >= Right
06,406 '      Loop Until StackPtr = 0
06,407 'End Sub
06,408
06,409 ' This works only under Windows [][][][][][][][][][][][][][][][][][][][][][][]
06,410
06,411 'Declare CustomType Library "memKAZE"
06,412 '  Sub memcpyKAZE (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
06,413 '  Function memcmpKAZE% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
06,414 'End Declare
06,415 'Dim Benchm As _MEM
06,416 'Elements = 10
06,417 'Granularity = 17
06,418 'Benchm = _MemNew(Elements * Granularity)
06,419 'For i = 9 To 0 Step -1
06,420 '  Pattern$ = String$(16, Chr$(Asc("0") + i)) + Chr$(0)
06,421 '  Print Pattern$
06,422 '  _MemPut Benchm, Benchm.OFFSET + Granularity * (9 - i), Pattern$
06,423 'Next
06,424 'Print: Print "Sorting...": Print
06,425 'Quicksort_QB64_Granularity Benchm, Elements, Granularity
06,426 'For i = 0 To 9
06,427 '  _MemGet Benchm, Benchm.OFFSET + Granularity * i, Pattern$
06,428 '  Print Pattern$
06,429 'Next
06,430 'End
06,431
06,432 ' Written by Sanmayce, 2021-Dec-12
06,433 ' The indexes are signed, but the elements are unsigned.
06,434 'Define A-Z As _INTEGER64
06,435 Sub Quicksort_QB64_Granularity (MemBlock As _MEM, NumberOfElements As _Integer64, Granularity As Integer)
06,436   Left = 1 - 1
06,437   Right = NumberOfElements - 1
06,438   ReDim Stack&&(Left To Right)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

06,439 SwapBuffer$ = Space$(Granularity)
06,440 Pivot$ = Space$(Granularity)
06,441 StackPtr = 0
06,442 StackPtr = StackPtr + 1
06,443 Stack&&(StackPtr) = Left
06,444 StackPtr = StackPtr + 1
06,445 Stack&&(StackPtr) = Right
06,446 Do 'Until StackPtr = 0
06,447     Right = Stack&&(StackPtr)
06,448     StackPtr = StackPtr - 1
06,449     Left = Stack&&(StackPtr)
06,450     StackPtr = StackPtr - 1
06,451 Do 'Until Left >= Right
06,452     'Pivot~&& = QWORDS~&&((Left + Right) \ 2)
06,453     memcpyKAZE _Offset(Pivot$), MemBlock.OFFSET + Granularity * ((Left + Right) \ 2), Granularity
06,454     Indx = Left
06,455     Jndx = Right
06,456     Do
06,457         'Do While (QWORDS~&&(Indx) < Pivot~&&)
06,458             BuggyWithLinuxNotOneOrMinusOne% = memcpyKAZE%(MemBlock.OFFSET + Granularity * Indx, _Offset(Pivot$), Granularity)
06,459         Do While (BuggyWithLinuxNotOneOrMinusOne% < 0) '= -1)
06,460             Indx = Indx + 1
06,461             BuggyWithLinuxNotOneOrMinusOne% = memcpyKAZE%(MemBlock.OFFSET + Granularity * Indx, _Offset(Pivot$), Granularity)
06,462         Loop
06,463         'Do While (QWORDS~&&(Jndx) > Pivot~&&)
06,464             BuggyWithLinuxNotOneOrMinusOne% = memcpyKAZE%(MemBlock.OFFSET + Granularity * Jndx, _Offset(Pivot$), Granularity)
06,465         Do While (BuggyWithLinuxNotOneOrMinusOne% > 0) '= 1)
06,466             Jndx = Jndx - 1
06,467             BuggyWithLinuxNotOneOrMinusOne% = memcpyKAZE%(MemBlock.OFFSET + Granularity * Jndx, _Offset(Pivot$), Granularity)
06,468         Loop
06,469         If Indx <= Jndx Then
06,470             If Indx < Jndx Then
06,471                 'Swap QWORDS~&&(Indx), QWORDS~&&(Jndx)
06,472                 memcpyKAZE _Offset(SwapBuffer$), MemBlock.OFFSET + Granularity * Jndx, Granularity
06,473                 memcpyKAZE MemBlock.OFFSET + Granularity * Jndx, MemBlock.OFFSET + Granularity * Indx, Granularity
06,474                 memcpyKAZE MemBlock.OFFSET + Granularity * Indx, _Offset(SwapBuffer$), Granularity
06,475             End If
06,476             Indx = Indx + 1
06,477             Jndx = Jndx - 1
06,478         End If
06,479     Loop While Indx <= Jndx
06,480     If Indx < Right Then
06,481         StackPtr = StackPtr + 1
06,482         Stack&&(StackPtr) = Indx
06,483         StackPtr = StackPtr + 1
06,484         Stack&&(StackPtr) = Right
06,485     End If
06,486     Right = Jndx
06,487 Loop Until Left >= Right

```

```
06,488      Loop Until StackPtr = 0
06,489 End Sub
06,490
```

# **UTF8toGesch.bas:**

00,001 REM Converting UTF8 to Gesch codepage

00,002

00,003 \$SCREENHIDE

00,004 \$CONSOLE

00,005 \_CONSOLE ON

00,006 \_CONSOLETITLE "Gesch codepage convertor, revision 2"

00,007 \_DEST \_CONSOLE

00,008

00,009 REM Microsoft\_windows\_cp1251: 'Cyrillic alphabet such as Russian, Bulgarian, Serbian Cyrillic and other languages. It is the most widely used for encoding the Bulgarian, Serbian and Macedonian languages.

00,010 REM 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143

00,011 REM DATA 1026,1027,8218,1107,8222,8230,8224,8225,8364,8240,1033,8249,1034,1036,1035,1039 '128+16\*0 to 128+16\*1-1

00,012 REM 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159

00,013 REM DATA 1106,8216,8217,8220,8221,8226,8211,8212,0 ,8482,1113,8250,1114,1116,1115,1119 '128+16\*1 to 128+16\*2-1

00,014 REM 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175

00,015 REM DATA 160 ,1038,1118,1032,164 ,1168,166 ,167 ,1025,169 ,1028,171 ,172 ,173 ,174 ,1031 '128+16\*2 to 128+16\*3-1

00,016 REM 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191

00,017 REM DATA 176 ,177 ,1030,1110,1169,181 ,182 ,183 ,1105,8470,1108,187 ,1112,1029,1109,1111 '128+16\*3 to 128+16\*4-1

00,018 REM Cyrillic:

00,019 REM DATA 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055 '128+16\*4 to 128+16\*5-1

00,020 REM DATA 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071 '128+16\*5 to 128+16\*6-1

00,021 REM DATA 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087 '128+16\*6 to 128+16\*7-1

00,022 REM DATA 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103 '128+16\*7 to 128+16\*8-1 --\

00,023 REM

00,024 REM Microsoft\_pc\_cpGESCH: 'Gesch is Sanmayce's layout, combining the MIK and 437, in this way:

00,025 REM

00,026 REM DATA 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055 '128+16\*0 to 128+16\*1-1

00,027 REM DATA 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071 '128+16\*1 to 128+16\*2-1

00,028 REM DATA 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087 '128+16\*2 to 128+16\*3-1

00,029 REM

00,030 REM DATA 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488 '128+16\*3 to 128+16\*4-1

00,031 REM DATA 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575 '128+16\*4 to 128+16\*5-1

00,032 REM DATA 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600 '128+16\*5 to 128+16\*6-1

00,033 REM

00,034 REM DATA 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103 '128+16\*6 to 128+16\*7-1 <-/  
00,035 REM 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

00,036 REM DATA 8216,8217,8218,8219,8220,8221,8222,8223,176 ,1118,1025,1105,171 ,187 ,175 ,8230 '128+16\*7 to 128+16\*8-1

00,037

00,038 'We need to replace 1251 only with the last line, above.

00,039

00,040 \_DEFINE A-Z AS \_INTEGER64

00,041

00,042 DIM byt AS STRING \* 1

00,043 DIM byt2 AS STRING \* 1

00,044 DIM byt3 AS STRING \* 1

00,045 IF COMMAND\$ = "" THEN PRINT "Usage: UTF8toGesch.exe filename": SYSTEM

00,046 OPEN COMMAND\$ FOR BINARY AS #1

00,047 OPEN COMMAND\$ + ".Gesch" FOR BINARY AS #2

00,048 qq = LOF(1)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 'PRINT qq
00,050 'Grrr, below not the same as ? qq i.e. not as it should?!
00,051 'PRINT "Converting"; LOF(1); "bytes..."
00,052 PRINT "Converting"; qq; "bytes..."
00,053 LOCATE CSRLIN, 1: PRINT "Done"; (0 * 100) \ qq; "%";
00,054 FOR iii = 1 TO qq
00,055     IF (iii * 100) \ qq > ((iii - 1) * 100) \ qq THEN LOCATE CSRLIN, 1: PRINT "Done"; (iii * 100) \ qq; "%";
00,056     GET #1, iii, byt$
00,057     SELECT CASE ASC(byt$)
00,058         CASE &HE2:
00,059             byt2$ = CHR$(32)
00,060             byt3$ = CHR$(32)
00,061             IF iii + 2 <= qq THEN
00,062                 GET #1, , byt2$
00,063                 GET #1, , byt3$
00,064             END IF
00,065             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9E THEN ' ,, lower " curved to left downwards
00,066                 iii = iii + 2
00,067                 bytNEW$ = CHR$(246)
00,068                 PUT #2, , bytNEW$
00,069             END IF
00,070             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9F THEN ' `` upper " curved to right downwards
00,071                 iii = iii + 2
00,072                 bytNEW$ = CHR$(247)
00,073                 PUT #2, , bytNEW$
00,074             END IF
00,075             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9C THEN ' upper " curved to right upwards
00,076                 iii = iii + 2
00,077                 bytNEW$ = CHR$(244)
00,078                 PUT #2, , bytNEW$
00,079             END IF
00,080             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9D THEN ' upper " curved to left downwards
00,081                 iii = iii + 2
00,082                 bytNEW$ = CHR$(245)
00,083                 PUT #2, , bytNEW$
00,084             END IF
00,085             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H98 THEN ' upper ' curved to right upwards
00,086                 iii = iii + 2
00,087                 bytNEW$ = CHR$(240)
00,088                 PUT #2, , bytNEW$
00,089             END IF
00,090             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H99 THEN ' upper ' curved to left downwards
00,091                 iii = iii + 2
00,092                 bytNEW$ = CHR$(241)
00,093                 PUT #2, , bytNEW$
00,094             END IF
00,095             IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9A THEN ' lower ' curved to left downwards
00,096                 iii = iii + 2
00,097                 bytNEW$ = CHR$(242)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,098         PUT #2, , bytNEW$
00,099     END IF
00,100     IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H9B THEN ' upper ' curved to right downwards
00,101         iii = iii + 2
00,102         bytNEW$ = CHR$(243)
00,103         PUT #2, , bytNEW$
00,104     END IF
00,105     IF ASC(byt2$) = &H80 AND (ASC(byt3$) = &H93 OR ASC(byt3$) = &H94) THEN ' -
00,106         iii = iii + 2
00,107         bytNEW$ = CHR$(196)
00,108         PUT #2, , bytNEW$
00,109     END IF
00,110     IF ASC(byt2$) = &H80 AND ASC(byt3$) = &HA6 THEN ' ...
00,111         iii = iii + 2
00,112         bytNEW$ = CHR$(255)
00,113         PUT #2, , bytNEW$
00,114     END IF
00,115     IF ASC(byt2$) = &H80 AND ASC(byt3$) = &H97 THEN ' double underscore
00,116         iii = iii + 2
00,117         bytNEW$ = CHR$(208)
00,118         PUT #2, , bytNEW$
00,119     END IF
00,120     IF ASC(byt2$) = &H96 AND ASC(byt3$) = &HAA THEN ' filled box
00,121         iii = iii + 2
00,122         bytNEW$ = CHR$(209)
00,123         PUT #2, , bytNEW$
00,124     END IF
00,125     IF ASC(byt2$) = &H96 AND ASC(byt3$) = &HAB THEN ' empty box
00,126         iii = iii + 2
00,127         bytNEW$ = CHR$(210)
00,128         PUT #2, , bytNEW$
00,129     END IF
00,130     IF ASC(byt2$) = &H96 AND ASC(byt3$) = &H80 THEN ' upper box
00,131         iii = iii + 2
00,132         bytNEW$ = CHR$(28)
00,133         PUT #2, , bytNEW$
00,134     END IF
00,135     IF ASC(byt2$) = &H96 AND ASC(byt3$) = &H81 THEN ' bold cursor
00,136         iii = iii + 2
00,137         bytNEW$ = CHR$(29)
00,138         PUT #2, , bytNEW$
00,139     END IF
00,140     IF ASC(byt2$) = &H96 AND ASC(byt3$) = &H84 THEN ' lower box
00,141         iii = iii + 2
00,142         bytNEW$ = CHR$(30)
00,143         PUT #2, , bytNEW$
00,144     END IF
00,145     IF ASC(byt2$) = &H8C AND ASC(byt3$) = &HA0 THEN ' high integral
00,146         iii = iii + 2

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,147      bytNEW$ = CHR$(209)
00,148      PUT #2, , bytNEW$
00,149      END IF
00,150      IF ASC(byt2$) = &H8C AND ASC(byt3$) = &HA1 THEN ' low integral
00,151          iii = iii + 2
00,152          bytNEW$ = CHR$(210)
00,153          PUT #2, , bytNEW$
00,154      END IF
00,155      IF ASC(byt2$) = &H89 AND ASC(byt3$) = &H88 THEN ' almost equal
00,156          iii = iii + 2
00,157          bytNEW$ = CHR$(127)
00,158          PUT #2, , bytNEW$
00,159      END IF
00,160      IF ASC(byt2$) = &H82 AND ASC(byt3$) = &H8A THEN ' under +
00,161          iii = iii + 2
00,162          bytNEW$ = CHR$(11)
00,163          PUT #2, , bytNEW$
00,164      END IF
00,165      IF ASC(byt2$) = &H82 AND ASC(byt3$) = &H8B THEN ' under -
00,166          iii = iii + 2
00,167          bytNEW$ = CHR$(12)
00,168          PUT #2, , bytNEW$
00,169      END IF
00,170
00,171      CASE &HC2:
00,172          byt2$ = CHR$(32)
00,173          IF iii + 1 <= qq THEN
00,174              GET #1, , byt2$
00,175          END IF
00,176          IF ASC(byt2$) = &HA0 THEN 'space
00,177              iii = iii + 1
00,178              bytNEW$ = CHR$(32)
00,179              PUT #2, , bytNEW$
00,180          END IF
00,181          IF ASC(byt2$) = &HAB THEN '<<
00,182              iii = iii + 1
00,183              bytNEW$ = CHR$(252)
00,184              PUT #2, , bytNEW$
00,185          END IF
00,186          IF ASC(byt2$) = &HBB THEN '>>
00,187              iii = iii + 1
00,188              bytNEW$ = CHR$(253)
00,189              PUT #2, , bytNEW$
00,190          END IF
00,191          IF ASC(byt2$) = &HB5 THEN 'micro
00,192              iii = iii + 1
00,193              bytNEW$ = CHR$(249)
00,194              PUT #2, , bytNEW$
00,195          END IF

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,196      IF ASC(byt2$) = &HA1 THEN 'r!
00,197          iii = iii + 1
00,198          bytNEW$ = CHR$(211)
00,199          PUT #2, , bytNEW$
00,200      END IF
00,201      IF ASC(byt2$) = &HBF THEN 'r?
00,202          iii = iii + 1
00,203          bytNEW$ = CHR$(212)
00,204          PUT #2, , bytNEW$
00,205      END IF
00,206
00,207
00,208      CASE &HC3:
00,209          byt2$ = CHR$(32)
00,210          IF iii + 1 <= qq THEN
00,211              GET #1, , byt2$
00,212          END IF
00,213          IF ASC(byt2$) = &H84 THEN 'A:
00,214              iii = iii + 1
00,215              bytNEW$ = CHR$(0)
00,216              PUT #2, , bytNEW$
00,217          END IF
00,218          IF ASC(byt2$) = &HA4 THEN 'a:
00,219              iii = iii + 1
00,220              bytNEW$ = CHR$(1)
00,221              PUT #2, , bytNEW$
00,222          END IF
00,223          IF ASC(byt2$) = &H96 THEN '0:
00,224              iii = iii + 1
00,225              bytNEW$ = CHR$(2)
00,226              PUT #2, , bytNEW$
00,227          END IF
00,228          IF ASC(byt2$) = &HB6 THEN 'o:
00,229              iii = iii + 1
00,230              bytNEW$ = CHR$(3)
00,231              PUT #2, , bytNEW$
00,232          END IF
00,233          IF ASC(byt2$) = &H9C THEN 'U:
00,234              iii = iii + 1
00,235              bytNEW$ = CHR$(4)
00,236              PUT #2, , bytNEW$
00,237          END IF
00,238          IF ASC(byt2$) = &HBC THEN 'u:
00,239              iii = iii + 1
00,240              bytNEW$ = CHR$(5)
00,241              PUT #2, , bytNEW$
00,242          END IF
00,243          IF ASC(byt2$) = &H9F THEN 'ss
00,244              iii = iii + 1

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,245      bytNEW$ = CHR$(6)
00,246      PUT #2, , bytNEW$
00,247      END IF
00,248      IF ASC(byt2$) = &H87 THEN 'C,
00,249          iii = iii + 1
00,250          bytNEW$ = CHR$(14)
00,251          PUT #2, , bytNEW$
00,252      END IF
00,253      IF ASC(byt2$) = &HA7 THEN 'c,
00,254          iii = iii + 1
00,255          bytNEW$ = CHR$(15)
00,256          PUT #2, , bytNEW$
00,257      END IF
00,258      IF ASC(byt2$) = &H82 THEN 'A^
00,259          iii = iii + 1
00,260          bytNEW$ = CHR$(198)
00,261          PUT #2, , bytNEW$
00,262      END IF
00,263      IF ASC(byt2$) = &HA2 THEN 'a^
00,264          iii = iii + 1
00,265          bytNEW$ = CHR$(199)
00,266          PUT #2, , bytNEW$
00,267      END IF
00,268      IF ASC(byt2$) = &H8A THEN 'E^
00,269          iii = iii + 1
00,270          bytNEW$ = CHR$(200)
00,271          PUT #2, , bytNEW$
00,272      END IF
00,273      IF ASC(byt2$) = &HAA THEN 'e^
00,274          iii = iii + 1
00,275          bytNEW$ = CHR$(201)
00,276          PUT #2, , bytNEW$
00,277      END IF
00,278      IF ASC(byt2$) = &H8E THEN 'I^
00,279          iii = iii + 1
00,280          bytNEW$ = CHR$(202)
00,281          PUT #2, , bytNEW$
00,282      END IF
00,283      IF ASC(byt2$) = &HAE THEN 'i^
00,284          iii = iii + 1
00,285          bytNEW$ = CHR$(203)
00,286          PUT #2, , bytNEW$
00,287      END IF
00,288      IF ASC(byt2$) = &H94 THEN 'O^
00,289          iii = iii + 1
00,290          bytNEW$ = CHR$(204)
00,291          PUT #2, , bytNEW$
00,292      END IF
00,293      IF ASC(byt2$) = &HB4 THEN 'o^

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,294      iii = iii + 1
00,295      bytNEW$ = CHR$(205)
00,296      PUT #2, , bytNEW$
00,297      END IF
00,298      IF ASC(byt2$) = &H9B THEN 'U^
00,299      iii = iii + 1
00,300      bytNEW$ = CHR$(206)
00,301      PUT #2, , bytNEW$
00,302      END IF
00,303      IF ASC(byt2$) = &HRB THEN 'u^
00,304      iii = iii + 1
00,305      bytNEW$ = CHR$(207)
00,306      PUT #2, , bytNEW$
00,307      END IF
00,308      IF ASC(byt2$) = &H89 THEN 'E'
00,309      iii = iii + 1
00,310      bytNEW$ = CHR$(16)
00,311      PUT #2, , bytNEW$
00,312      END IF
00,313      IF ASC(byt2$) = &HA9 THEN 'e'
00,314      iii = iii + 1
00,315      bytNEW$ = CHR$(17)
00,316      PUT #2, , bytNEW$
00,317      END IF
00,318      IF ASC(byt2$) = &H80 THEN 'A'
00,319      iii = iii + 1
00,320      bytNEW$ = CHR$(18)
00,321      PUT #2, , bytNEW$
00,322      END IF
00,323      IF ASC(byt2$) = &HA0 THEN 'a'
00,324      iii = iii + 1
00,325      bytNEW$ = CHR$(19)
00,326      PUT #2, , bytNEW$
00,327      END IF
00,328      IF ASC(byt2$) = &H88 THEN 'E'
00,329      iii = iii + 1
00,330      bytNEW$ = CHR$(20)
00,331      PUT #2, , bytNEW$
00,332      END IF
00,333      IF ASC(byt2$) = &HA8 THEN 'e'
00,334      iii = iii + 1
00,335      bytNEW$ = CHR$(21)
00,336      PUT #2, , bytNEW$
00,337      END IF
00,338      IF ASC(byt2$) = &H8C THEN 'I'
00,339      iii = iii + 1
00,340      bytNEW$ = CHR$(22)
00,341      PUT #2, , bytNEW$
00,342      END IF

```

```

00,343     IF ASC(byt2$) = &HAC THEN 'i'
00,344         iii = iii + 1
00,345         bytNEW$ = CHR$(23)
00,346         PUT #2, , bytNEW$
00,347     END IF
00,348     IF ASC(byt2$) = &H92 THEN 'O'
00,349         iii = iii + 1
00,350         bytNEW$ = CHR$(24)
00,351         PUT #2, , bytNEW$
00,352     END IF
00,353     IF ASC(byt2$) = &HB2 THEN 'o'
00,354         iii = iii + 1
00,355         bytNEW$ = CHR$(25)
00,356         PUT #2, , bytNEW$
00,357     END IF
00,358     IF ASC(byt2$) = &H99 THEN 'U'
00,359         iii = iii + 1
00,360         bytNEW$ = CHR$(26)
00,361         PUT #2, , bytNEW$
00,362     END IF
00,363     IF ASC(byt2$) = &HB9 THEN 'u'
00,364         iii = iii + 1
00,365         bytNEW$ = CHR$(27)
00,366         PUT #2, , bytNEW$
00,367     END IF
00,368     IF ASC(byt2$) = &HB7 THEN 'division'
00,369         iii = iii + 1
00,370         bytNEW$ = CHR$(31)
00,371         PUT #2, , bytNEW$
00,372     END IF
00,373     IF ASC(byt2$) = &H8F THEN 'I:
00,374         iii = iii + 1
00,375         bytNEW$ = CHR$(189)
00,376         PUT #2, , bytNEW$
00,377     END IF
00,378     IF ASC(byt2$) = &HAF THEN 'i:
00,379         iii = iii + 1
00,380         bytNEW$ = CHR$(190)
00,381         PUT #2, , bytNEW$
00,382     END IF
00,383     IF ASC(byt2$) = &H91 THEN 'N'
00,384         iii = iii + 1
00,385         bytNEW$ = CHR$(213)
00,386         PUT #2, , bytNEW$
00,387     END IF
00,388     IF ASC(byt2$) = &HB1 THEN 'n~
00,389         iii = iii + 1
00,390         bytNEW$ = CHR$(214)
00,391         PUT #2, , bytNEW$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,392      END IF
00,393      IF ASC(byt2$) = &H9D THEN 'Y'
00,394          iii = iii + 1
00,395          bytNEW$ = CHR$(215)
00,396          PUT #2, , bytNEW$
00,397      END IF
00,398      IF ASC(byt2$) = &HBD THEN 'y'
00,399          iii = iii + 1
00,400          bytNEW$ = CHR$(216)
00,401          PUT #2, , bytNEW$
00,402      END IF
00,403      IF ASC(byt2$) = &H86 THEN 'AE'
00,404          iii = iii + 1
00,405          bytNEW$ = CHR$(220)
00,406          PUT #2, , bytNEW$
00,407      END IF
00,408      IF ASC(byt2$) = &HA6 THEN 'ae'
00,409          iii = iii + 1
00,410          bytNEW$ = CHR$(221)
00,411          PUT #2, , bytNEW$
00,412      END IF
00,413      IF ASC(byt2$) = &H81 THEN 'A'
00,414          iii = iii + 1
00,415          bytNEW$ = CHR$(7)
00,416          PUT #2, , bytNEW$
00,417      END IF
00,418      IF ASC(byt2$) = &HA1 THEN 'a'
00,419          iii = iii + 1
00,420          bytNEW$ = CHR$(8)
00,421          PUT #2, , bytNEW$
00,422      END IF
00,423      IF ASC(byt2$) = &H93 THEN 'O'
00,424          iii = iii + 1
00,425          bytNEW$ = CHR$(11)
00,426          PUT #2, , bytNEW$
00,427      END IF
00,428      IF ASC(byt2$) = &HB3 THEN 'o'
00,429          iii = iii + 1
00,430          bytNEW$ = CHR$(12)
00,431          PUT #2, , bytNEW$
00,432      END IF
00,433      IF ASC(byt2$) = &H8D THEN 'I'
00,434          iii = iii + 1
00,435          bytNEW$ = CHR$(28)
00,436          PUT #2, , bytNEW$
00,437      END IF
00,438      IF ASC(byt2$) = &HAD THEN 'i'
00,439          iii = iii + 1
00,440          bytNEW$ = CHR$(29)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

00,441         PUT #2, , bytNEW$
00,442     END IF
00,443     IF ASC(byt2$) = &H9A THEN 'U'
00,444         iii = iii + 1
00,445         bytNEW$ = CHR$(30)
00,446         PUT #2, , bytNEW$
00,447     END IF
00,448     IF ASC(byt2$) = &HBA THEN 'u'
00,449         iii = iii + 1
00,450         bytNEW$ = CHR$(31)
00,451         PUT #2, , bytNEW$
00,452     END IF
00,453
00,454
00,455     CASE &HC4:
00,456         byt2$ = CHR$(32)
00,457         IF iii + 1 <= qq THEN
00,458             GET #1, , byt2$
00,459         END IF
00,460         IF ASC(byt2$) = &H82 THEN 'A kratko
00,461             iii = iii + 1
00,462             bytNEW$ = CHR$(181)
00,463             PUT #2, , bytNEW$
00,464         END IF
00,465         IF ASC(byt2$) = &H83 THEN 'a kratko
00,466             iii = iii + 1
00,467             bytNEW$ = CHR$(182)
00,468             PUT #2, , bytNEW$
00,469         END IF
00,470         IF ASC(byt2$) = &H94 THEN 'E kratko
00,471             iii = iii + 1
00,472             bytNEW$ = CHR$(183)
00,473             PUT #2, , bytNEW$
00,474         END IF
00,475         IF ASC(byt2$) = &H95 THEN 'e kratko
00,476             iii = iii + 1
00,477             bytNEW$ = CHR$(184)
00,478             PUT #2, , bytNEW$
00,479         END IF
00,480
00,481
00,482     CASE &HC5:
00,483         byt2$ = CHR$(32)
00,484         IF iii + 1 <= qq THEN
00,485             GET #1, , byt2$
00,486         END IF
00,487         IF ASC(byt2$) = &H92 THEN 'OE
00,488             iii = iii + 1
00,489             bytNEW$ = CHR$(222)

```

```

00,490         PUT #2, , bytNEW$
00,491     END IF
00,492     IF ASC(byt2$) = &H93 THEN 'oe
00,493         iii = iii + 1
00,494         bytNEW$ = CHR$(223)
00,495         PUT #2, , bytNEW$
00,496     END IF
00,497     IF ASC(byt2$) = &H8E THEN 'O kratko
00,498         iii = iii + 1
00,499         bytNEW$ = CHR$(185)
00,500         PUT #2, , bytNEW$
00,501     END IF
00,502     IF ASC(byt2$) = &H8F THEN 'o kratko
00,503         iii = iii + 1
00,504         bytNEW$ = CHR$(186)
00,505         PUT #2, , bytNEW$
00,506     END IF
00,507
00,508
00,509     CASE &HD0:
00,510         byt2$ = CHR$(32)
00,511         IF iii + 1 <= qq THEN
00,512             GET #1, , byt2$
00,513         END IF
00,514         IF ASC(byt2$) = &H81 THEN 'E:
00,515             iii = iii + 1
00,516             bytNEW$ = CHR$(250)
00,517             PUT #2, , bytNEW$
00,518         END IF
00,519         IF ASC(byt2$) = &H8E THEN 'Y kratko
00,520             iii = iii + 1
00,521             bytNEW$ = CHR$(187)
00,522             PUT #2, , bytNEW$
00,523         END IF
00,524         IF ASC(byt2$) >= &H90 AND ASC(byt2$) <= &HBF THEN
00,525             iii = iii + 1
00,526             bytNEW$ = CHR$(128 + (ASC(byt2$) - &H90))
00,527             PUT #2, , bytNEW$
00,528         END IF
00,529
00,530     CASE &HD1:
00,531         byt2$ = CHR$(32)
00,532         IF iii + 1 <= qq THEN
00,533             GET #1, , byt2$
00,534         END IF
00,535         IF ASC(byt2$) = &H91 THEN 'e:
00,536             iii = iii + 1
00,537             bytNEW$ = CHR$(251)
00,538             PUT #2, , bytNEW$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,539      END IF
00,540      IF ASC(byt2$) = &H9E THEN 'y(
00,541          iii = iii + 1
00,542          bytNEW$ = CHR$(188)
00,543          PUT #2, , bytNEW$
00,544      END IF
00,545      IF ASC(byt2$) >= &H80 AND ASC(byt2$) <= &H8F THEN 'A..ya
00,546          iii = iii + 1
00,547          bytNEW$ = CHR$(224 + (ASC(byt2$) - &H80))
00,548          PUT #2, , bytNEW$
00,549      END IF
00,550
00,551      CASE ELSE
00,552          PUT #2, , byt$
00,553  END SELECT
00,554
00,555
00,556  'SELECT CASE ASC(byt$)
00,557  '    CASE 130:
00,558  '        bytNEW$ = CHR$(242)
00,559  '        PUT #2, , bytNEW$
00,560
00,561  '    CASE 132:
00,562  '        bytNEW$ = CHR$(246)
00,563  '        PUT #2, , bytNEW$
00,564
00,565  '    CASE 133:
00,566  '        bytNEW$ = CHR$(255)
00,567  '        PUT #2, , bytNEW$
00,568
00,569  '    CASE 145:
00,570  '        bytNEW$ = CHR$(240)
00,571  '        PUT #2, , bytNEW$
00,572
00,573  '    CASE 146:
00,574  '        bytNEW$ = CHR$(241)
00,575  '        PUT #2, , bytNEW$
00,576
00,577  '    CASE 147:
00,578  '        bytNEW$ = CHR$(244)
00,579  '        PUT #2, , bytNEW$
00,580
00,581  '    CASE 148:
00,582  '        bytNEW$ = CHR$(245)
00,583  '        PUT #2, , bytNEW$
00,584
00,585  '    CASE 152:
00,586  '        bytNEW$ = CHR$(0)
00,587  '        PUT #2, , bytNEW$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,588
00,589 ' CASE 160:
00,590 '     bytNEW$ = CHR$(0)
00,591 '     PUT #2, , bytNEW$
00,592
00,593 ' CASE 162:
00,594 '     bytNEW$ = CHR$(249)
00,595 '     PUT #2, , bytNEW$
00,596
00,597 ' CASE 168:
00,598 '     bytNEW$ = CHR$(250)
00,599 '     PUT #2, , bytNEW$
00,600
00,601 ' CASE 184:
00,602 '     bytNEW$ = CHR$(251)
00,603 '     PUT #2, , bytNEW$
00,604
00,605 '     'Not in the tables above, but should be changed [
00,606 ' CASE 171:
00,607 '     bytNEW$ = CHR$(252)
00,608 '     PUT #2, , bytNEW$
00,609 ' CASE 187:
00,610 '     bytNEW$ = CHR$(253)
00,611 '     PUT #2, , bytNEW$
00,612 ' CASE 150:
00,613 '     bytNEW$ = CHR$(196)
00,614 '     PUT #2, , bytNEW$
00,615 ' CASE 151:
00,616 '     bytNEW$ = CHR$(196)
00,617 '     PUT #2, , bytNEW$
00,618 ' CASE 173:
00,619 '     bytNEW$ = CHR$(196)
00,620 '     PUT #2, , bytNEW$
00,621 '     'Not in the tables above, but should be changed ]
00,622
00,623 ' CASE 128 + 16 * 4 TO 128 + 16 * 5 - 1:
00,624 '     bytNEW$ = CHR$(ASC(byt$) - (4 - 0) * 16)
00,625 '     PUT #2, , bytNEW$
00,626 ' CASE 128 + 16 * 5 TO 128 + 16 * 6 - 1:
00,627 '     bytNEW$ = CHR$(ASC(byt$) - (5 - 1) * 16)
00,628 '     PUT #2, , bytNEW$
00,629 ' CASE 128 + 16 * 6 TO 128 + 16 * 7 - 1:
00,630 '     bytNEW$ = CHR$(ASC(byt$) - (6 - 2) * 16)
00,631 '     PUT #2, , bytNEW$
00,632 ' CASE 128 + 16 * 7 TO 128 + 16 * 8 - 1:
00,633 '     bytNEW$ = CHR$(ASC(byt$) - (7 - 6) * 16)
00,634 '     PUT #2, , bytNEW$
00,635 ' CASE ELSE:
00,636 '     PUT #2, , byt$

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,637 'END SELECT
00,638 NEXT
00,639 LOCATE CSRLIN, 1: PRINT "Done"; (qq * 100) \ qq; "%"
00,640 CLOSE #1, #2
00,641 SYSTEM
00,642
00,643
00,644 'Microsoft_pc_cpGESCH: 'Schpitz/Gesch (a.k.a. Georgievica a.k.a. Geschovica) is Sanmayce's layout, combining the MIK and 437, in this way: (NOT RECODING only 009,010,013):
00,645 '' shpitz
00,646
00,647 '' DEFINITIONS
00,648 '' top or extreme part
00,649 '' the ultimate, the best of, "cool"
00,650 '' typical, classic, a perfect example of; definitively
00,651
00,652 '' LANGUAGES OF ORIGIN
00,653 '' Yiddish
00,654
00,655 '' ETYMOLOGY
00,656 '' ????? shpits 'tip, peak'
00,657
00,658 '' ALTERNATIVE SPELLINGS
00,659 '' schpitz, shpits, shpitzzy, shpitsy
00,660
00,661 ''NOTES
00,662 ''Steinmetz, "Yiddish and English" has an entry for shpits, but the meaning is simply 'tip.'
00,663
00,664 '' ASCII 000..031 have to accomodate German and French ' &H203e = 8254 is upperscore ' &H00DC=220
00,665 'DATA 196,228,214,246,&H00DC,&H00FC,223,&H00c1,&H00e1,&h00C7,&H00E7
00,666 ''A:a:O:o:U:u:ss A' a' C'c'
00,667 'DATA &H0c9,&H00e9,&H0c0,&H00e0,&H00c8,&H00e8,&H00cc,&H00ec,&H00d2,&H00f2,&H00d9,&H00f9
00,668 '' E'e' A'a'E'e'I'i'O'o'U'u'
00,669
00,670 ''DATA &H00ch,&H00eb,&H00cf,&H00ef
00,671 '' E:e: I:i: ' E:e: are present in Russian, so remove them
00,672
00,673 ''First half of big Cyrillic letters:
00,674 'DATA 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,675 ''Second half of big Cyrillic letters:
00,676 'DATA 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,677 ''First half of small Cyrillic letters:
00,678 'DATA 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,679 ''1st third of CP437 drawing symbols:
00,680 'DATA 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,681 ''2nd third of CP437 drawing symbols:
00,682 'DATA 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,683 ''3rd third of CP437 drawing symbols:
00,684 'DATA 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,685 ''Second half of small Cyrillic letters:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

00,686 'DATA 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103

00,687 ''Last 16 of CP437 symbols:

00,688 '' u-kr E: e: << >> Up\_ ...

00,689 ''DATA 8216,8217,8218,8219,8220,8221,8222,8223,176,1118,1025,1105,171,187,175,8230

00,690 '' MICRO E: e: << >> Up\_ ...

00,691 'DATA 8216,8217,8218,8219,8220,8221,8222,8223,176,&h00b5,1025,1105,171,187,175,8230

00,692

00,693 'DATA &H00c6,&H00E6,&H0152,&H0153

00,694 '' AE ae OEoe

00,695

00,696 'DATA &h2248

00,697 ''almost equal to

00,698 ''DATA &H221a

00,699 ''square root

00,700

00,701 'DATA &H0102,&H0103,&H0114,&H0115,&H014e,&H014f,&H040e,&H045e,&H00cf,&H00ef

00,702 ''Cyrillic short vowels: AaEeOoYy I:i:

00,703 'DATA &H00c2,&H00e2,&H00ca,&H00ea,&H00ce,&H00ee,&H00d4,&H00f4,&H00db,&H00fb

00,704 '' A^a^ E^e^ I^i^ O^o^ U^u^

00,705

00,706 'DATA &h2017,&H2320,&H2321,&h00a1,&h00bf,&h00d1,&h00f1,&h00dd,&h00fd

00,707 ''doubleunderline IntegralH IntegralL, Spanish: r! r? N~ n~ Y^ y'

00,708

00,709 'DATA &h00cd,&h00ed,&h00da,&h00fa

00,710 '' I^i^ U^u^

00,711

00,712 'DATA &h00D3,&h00F3

00,713 '' O' o'

00,714

00,715 '' Portuguese makes use of five diacritics: the cedilla (c,), acute accent (a' e' i' o' u'), circumflex accent (a^, e^, o^), tilde (a~, o~), and grave accent (a', and rarely e', i', o', and u').

00,716

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

# **1251toGesch.bas:**

```
00,001 REM Converting windows_cp1251 to Gesch codpage
00,002
00,003 $SCREENHIDE
00,004 $CONSOLE
00,005 _CONSOLE ON
00,006 _CONSOLETITLE "Gesch codepage convertor"
00,007 _DEST _CONSOLE
00,008
00,009 REM Microsoft_windows_cp1251: 'Cyrillic alphabet such as Russian, Bulgarian, Serbian Cyrillic and other languages. It is the most widely used for encoding the Bulgarian, Serbian and Macedonian languages.
00,010 REM      128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
00,011 REM DATA 1026,1027,8218,1107,8222,8230,8224,8225,8364,8240,1033,8249,1034,1036,1035,1039 '128+16*0 to 128+16*1-1
00,012 REM      144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
00,013 REM DATA 1106,8216,8217,8220,8221,8226,8211,8212,0 ,8482,1113,8250,1114,1116,1115,1119 '128+16*1 to 128+16*2-1
00,014 REM      160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
00,015 REM DATA 160 ,1038,1118,1032,164 ,1168,166 ,167 ,1025,169 ,1028,171 ,172 ,173 ,174 ,1031 '128+16*2 to 128+16*3-1
00,016 REM      176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191
00,017 REM DATA 176 ,177 ,1030,1110,1169,181 ,182 ,183 ,1105,8470,1108,187 ,1112,1029,1109,1111 '128+16*3 to 128+16*4-1
00,018 REM Cyrillic:
00,019 REM DATA 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055 '128+16*4 to 128+16*5-1
00,020 REM DATA 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071 '128+16*5 to 128+16*6-1
00,021 REM DATA 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087 '128+16*6 to 128+16*7-1
00,022 REM DATA 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103 '128+16*7 to 128+16*8-1 --\
00,023 REM |
00,024 REM Microsoft_pc_cpGESCH: 'Gesch is Sanmayce's layout, combining the MIK and 437, in this way: |
00,025 REM |
00,026 REM DATA 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055 '128+16*0 to 128+16*1-1 |
00,027 REM DATA 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071 '128+16*1 to 128+16*2-1 |
00,028 REM DATA 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087 '128+16*2 to 128+16*3-1 |
00,029 REM |
00,030 REM DATA 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488 '128+16*3 to 128+16*4-1 |
00,031 REM DATA 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575 '128+16*4 to 128+16*5-1 |
00,032 REM DATA 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600 '128+16*5 to 128+16*6-1 |
00,033 REM |
00,034 REM DATA 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103 '128+16*6 to 128+16*7-1 <-/
00,035 REM      240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
00,036 REM DATA 8216,8217,8218,8219,8220,8221,8222,8223,176 ,1118,1025,1105,171 ,187 ,175 ,8230 '128+16*7 to 128+16*8-1
00,037
00,038 'We need to replace 1251 only with the last line, above.
00,039
00,040 _DEFINE A-Z AS _INTEGER64
00,041
00,042 DIM byt AS STRING * 1
00,043 IF COMMAND$ = "" THEN PRINT "Usage: 1251toGesch.exe filename": SYSTEM
00,044 OPEN COMMAND$ FOR BINARY AS #1
00,045 OPEN COMMAND$ + ".Gesch" FOR BINARY AS #2
00,046 qq = LOF(1)
00,047 'PRINT qq
00,048 'Grrr, below not the same as ? qq i.e. not as it should?!
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 'PRINT "Converting"; LOF(1); "bytes..."
00,050 PRINT "Converting"; qq; "bytes..."
00,051 FOR iii = 1 TO qq
00,052     GET #1, iii, byt$
00,053     SELECT CASE ASC(byt$)
00,054         CASE 130:
00,055             bytNEW$ = CHR$(242)
00,056             PUT #2, , bytNEW$
00,057
00,058         CASE 132:
00,059             bytNEW$ = CHR$(246)
00,060             PUT #2, , bytNEW$
00,061
00,062         CASE 133:
00,063             bytNEW$ = CHR$(255)
00,064             PUT #2, , bytNEW$
00,065
00,066         CASE 145:
00,067             bytNEW$ = CHR$(240)
00,068             PUT #2, , bytNEW$
00,069
00,070         CASE 146:
00,071             bytNEW$ = CHR$(241)
00,072             PUT #2, , bytNEW$
00,073
00,074         CASE 147:
00,075             bytNEW$ = CHR$(244)
00,076             PUT #2, , bytNEW$
00,077
00,078         CASE 148:
00,079             bytNEW$ = CHR$(245)
00,080             PUT #2, , bytNEW$
00,081
00,082         CASE 152:
00,083             bytNEW$ = CHR$(0)
00,084             PUT #2, , bytNEW$
00,085
00,086         CASE 160:
00,087             bytNEW$ = CHR$(0)
00,088             PUT #2, , bytNEW$
00,089
00,090         CASE 162:
00,091             bytNEW$ = CHR$(249)
00,092             PUT #2, , bytNEW$
00,093
00,094         CASE 168:
00,095             bytNEW$ = CHR$(250)
00,096             PUT #2, , bytNEW$
00,097

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

00,098      CASE 184:
00,099          bytNEW$ = CHR$(251)
00,100          PUT #2, , bytNEW$
00,101
00,102          'Not in the tables above, but should be changed [
00,103      CASE 171:
00,104          bytNEW$ = CHR$(252)
00,105          PUT #2, , bytNEW$
00,106      CASE 187:
00,107          bytNEW$ = CHR$(253)
00,108          PUT #2, , bytNEW$
00,109      CASE 150:
00,110          bytNEW$ = CHR$(196)
00,111          PUT #2, , bytNEW$
00,112      CASE 151:
00,113          bytNEW$ = CHR$(196)
00,114          PUT #2, , bytNEW$
00,115      CASE 173:
00,116          bytNEW$ = CHR$(196)
00,117          PUT #2, , bytNEW$
00,118          'Not in the tables above, but should be changed ]
00,119
00,120      CASE 128 + 16 * 4 TO 128 + 16 * 5 - 1:
00,121          bytNEW$ = CHR$(ASC(byt$) - (4 - 0) * 16)
00,122          PUT #2, , bytNEW$
00,123      CASE 128 + 16 * 5 TO 128 + 16 * 6 - 1:
00,124          bytNEW$ = CHR$(ASC(byt$) - (5 - 1) * 16)
00,125          PUT #2, , bytNEW$
00,126      CASE 128 + 16 * 6 TO 128 + 16 * 7 - 1:
00,127          bytNEW$ = CHR$(ASC(byt$) - (6 - 2) * 16)
00,128          PUT #2, , bytNEW$
00,129      CASE 128 + 16 * 7 TO 128 + 16 * 8 - 1:
00,130          bytNEW$ = CHR$(ASC(byt$) - (7 - 6) * 16)
00,131          PUT #2, , bytNEW$
00,132      CASE ELSE:
00,133          PUT #2, , byt$
00,134      END SELECT
00,135 NEXT
00,136 CLOSE #1, #2
00,137 SYSTEM
00,138

```

**MEM.H:**

```
00,001 #include<windows.h>
00,002 #include<stdio.h>
00,003 #include<tchar.h>
00,004
00,005
00,006 uint64 MemInUsePercent();
00,007 uint64 TotalPhysicalMem ();
00,008 uint64 FreePhysicalMem ();
00,009 uint64 TotalPagingFile ();
00,010 uint64 FreePagingFile ();
00,011 uint64 TotalVirtualMem ();
00,012 uint64 FreeVirtualMem ();
00,013 uint64 FreeExtendedMem ();
00,014
00,015 static float CalculateCPULoad();
00,016 static unsigned long long FileTimeToInt64();
00,017 float GetCPULoad();
00,018
00,019
00,020 uint64 MemInUsePercent () {
00,021     MEMORYSTATUSEX statex;
00,022     statex.dwLength = sizeof (statex);
00,023     GlobalMemoryStatusEx (&statex);
00,024     return statex.dwMemoryLoad;
00,025 }
00,026
00,027 uint64 TotalPhysicalMem () {
00,028     MEMORYSTATUSEX statex;
00,029     statex.dwLength = sizeof (statex);
00,030     GlobalMemoryStatusEx (&statex);
00,031     return statex.ullTotalPhys;
00,032 }
00,033
00,034 uint64 FreePhysicalMem () {
00,035     MEMORYSTATUSEX statex;
00,036     statex.dwLength = sizeof (statex);
00,037     GlobalMemoryStatusEx (&statex);
00,038     return statex.ullAvailPhys;
00,039 }
00,040
00,041 uint64 TotalPagingFile () {
00,042     MEMORYSTATUSEX statex;
00,043     statex.dwLength = sizeof (statex);
00,044     GlobalMemoryStatusEx (&statex);
00,045     return statex.ullTotalPageFile;
00,046 }
00,047
00,048 uint64 FreePagingFile () {
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 MEMORYSTATUSEX statex;
00,050 statex.dwLength = sizeof (statex);
00,051 GlobalMemoryStatusEx (&statex);
00,052 return statex.ullAvailPageFile;
00,053 }
00,054
00,055 uint64 TotalVirtualMem () {
00,056     MEMORYSTATUSEX statex;
00,057     statex.dwLength = sizeof (statex);
00,058     GlobalMemoryStatusEx (&statex);
00,059     return statex.ullTotalVirtual;
00,060 }
00,061
00,062 uint64 FreeVirtualMem () {
00,063     MEMORYSTATUSEX statex;
00,064     statex.dwLength = sizeof (statex);
00,065     GlobalMemoryStatusEx (&statex);
00,066     return statex.ullAvailVirtual;
00,067 }
00,068
00,069 uint64 FreeExtendedMem () {
00,070     MEMORYSTATUSEX statex;
00,071     statex.dwLength = sizeof (statex);
00,072     GlobalMemoryStatusEx (&statex);
00,073     return statex.ullAvailExtendedVirtual;
00,074 }
00,075
00,076 static float CalculateCPULoad(unsigned long long idleTicks, unsigned long long totalTicks)
00,077 {
00,078     static unsigned long long _previousTotalTicks = 0;
00,079     static unsigned long long _previousIdleTicks = 0;
00,080
00,081     unsigned long long totalTicksSinceLastTime = totalTicks - _previousTotalTicks;
00,082     unsigned long long idleTicksSinceLastTime = idleTicks - _previousIdleTicks;
00,083
00,084
00,085     float ret = 1.0f - ((totalTicksSinceLastTime > 0) ? ((float)idleTicksSinceLastTime) / totalTicksSinceLastTime : 0);
00,086
00,087     _previousTotalTicks = totalTicks;
00,088     _previousIdleTicks = idleTicks;
00,089     return ret;
00,090 }
00,091
00,092 static unsigned long long FileTimeToInt64(const FILETIME & ft)
00,093 {
00,094     return (((unsigned long long)(ft.dwHighDateTime)) << 32) | ((unsigned long long)ft.dwLowDateTime);
00,095 }
00,096
00,097 // Returns 1.0f for "CPU fully pinned", 0.0f for "CPU idle", or somewhere in between

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,098 // You'll need to call this at regular intervals, since it measures the load between
```

```
00,099 // the previous call and the current one. Returns -1.0 on error.
```

```
00,100 float GetCPULoad()
```

```
00,101 {
```

```
00,102     FILETIME idleTime, kernelTime, userTime;
```

```
00,103     return GetSystemTimes(&idleTime, &kernelTime, &userTime) ? CalculateCPULoad(FileTimeToInt64(idleTime), FileTimeToInt64(kernelTime) + FileTimeToInt64(userTime)) : -1.0f;
```

```
00,104 }
```

**XGRAM PAGODA5 64bit.bat:**

```
00,001 @if '%1' == '' goto Usage
00,002 @if '%2' == '' goto Usage
00,003 @goto Doit
00,004 :Usage
00,005 @echo Usage: PAGODA5.bat lowercasedword corpusname
00,006 @goto Finish
00,007 :Doit
00,008
00,009 if exist %2.01.txt.sorted goto OK1
00,010 if not exist %2.01.txt.sorted.7z goto RIPanyway
00,011 7za_x86_v1900.exe e -y %2.01.txt.sorted.7z
00,012 :OK1
00,013 if exist %2.02.txt.sorted goto OK2
00,014 if not exist %2.02.txt.sorted.7z goto RIPanyway
00,015 7za_x86_v1900.exe e -y %2.02.txt.sorted.7z
00,016 :OK2
00,017 if exist %2.03.txt.sorted goto OK3
00,018 if not exist %2.03.txt.sorted.7z goto RIPanyway
00,019 7za_x86_v1900.exe e -y %2.03.txt.sorted.7z
00,020 :OK3
00,021 if exist %2.04.txt.sorted goto OK4
00,022 if not exist %2.04.txt.sorted.7z goto RIPanyway
00,023 7za_x86_v1900.exe e -y %2.04.txt.sorted.7z
00,024 :OK4
00,025 if exist %2.05.txt.sorted goto OK5
00,026 if not exist %2.05.txt.sorted.7z goto RIPanyway
00,027 7za_x86_v1900.exe e -y %2.05.txt.sorted.7z
00,028 :OK5
00,029
00,030 goto SkipRIP
00,031
00,032 :RIPanyway
00,033
00,034 @dir %2/b >%2.lst
00,035
00,036 Leprechaun_x-leton_64bit_Intel_01_001p.exe %2.lst %2.01 14123456 Y
00,037 @if %errorlevel% == 0 goto OKay1
00,038 Leprechaun_x-leton_64bit_Intel_01_008p.exe %2.lst %2.01 14123456 Y
00,039 @if %errorlevel% == 0 goto OKay1
00,040 Leprechaun_x-leton_64bit_Intel_01_512p.exe %2.lst %2.01 14123456 Y
00,041 @if %errorlevel% == 0 goto OKay1
00,042 del %2.01
00,043 goto Finish
00,044 :OKay1
00,045
00,046 Leprechaun_x-leton_64bit_Intel_02_001p.exe %2.lst %2.02 14123456 Y
00,047 @if %errorlevel% == 0 goto OKay2
00,048 Leprechaun_x-leton_64bit_Intel_02_008p.exe %2.lst %2.02 14123456 Y
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 @if %errorlevel% == 0 goto Okay2
00,050 Leprechaun_x-leton_64bit_Intel_02_512p.exe %2.1st %2.02 14123456 Y
00,051 @if %errorlevel% == 0 goto Okay2
00,052 del %2.02
00,053 goto Finish
00,054 :Okay2
00,055
00,056 Leprechaun_x-leton_64bit_Intel_03_001p.exe %2.1st %2.03 14123456 Y
00,057 @if %errorlevel% == 0 goto Okay3
00,058 Leprechaun_x-leton_64bit_Intel_03_008p.exe %2.1st %2.03 14123456 Y
00,059 @if %errorlevel% == 0 goto Okay3
00,060 Leprechaun_x-leton_64bit_Intel_03_512p.exe %2.1st %2.03 14123456 Y
00,061 @if %errorlevel% == 0 goto Okay3
00,062 del %2.03
00,063 goto Finish
00,064 :Okay3
00,065
00,066 Leprechaun_x-leton_64bit_Intel_04_001p.exe %2.1st %2.04 14123456 Y
00,067 @if %errorlevel% == 0 goto Okay4
00,068 Leprechaun_x-leton_64bit_Intel_04_008p.exe %2.1st %2.04 14123456 Y
00,069 @if %errorlevel% == 0 goto Okay4
00,070 Leprechaun_x-leton_64bit_Intel_04_512p.exe %2.1st %2.04 14123456 Y
00,071 @if %errorlevel% == 0 goto Okay4
00,072 del %2.04
00,073 goto Finish
00,074 :Okay4
00,075
00,076 Leprechaun_x-leton_64bit_Intel_05_001p.exe %2.1st %2.05 14123456 Y
00,077 @if %errorlevel% == 0 goto Okay5
00,078 Leprechaun_x-leton_64bit_Intel_05_008p.exe %2.1st %2.05 14123456 Y
00,079 @if %errorlevel% == 0 goto Okay5
00,080 Leprechaun_x-leton_64bit_Intel_05_512p.exe %2.1st %2.05 14123456 Y
00,081 @if %errorlevel% == 0 goto Okay5
00,082 del %2.05
00,083 goto Finish
00,084 :Okay5
00,085
00,086 sort /+10 /M 1012012 %2.01 /O %2.01.txt.sorted
00,087 sort /+10 /M 1012012 %2.02 /O %2.02.txt.sorted
00,088 sort /+10 /M 1012012 %2.03 /O %2.03.txt.sorted
00,089 sort /+10 /M 1012012 %2.04 /O %2.04.txt.sorted
00,090 sort /+10 /M 1012012 %2.05 /O %2.05.txt.sorted
00,091
00,092 @rem To avoid 'C:\Users\GOLDEN~1\AppData\Local\Temp\srt6CD8.tmp There is not enough space on the disk.' failures:
00,093 @rem set TMP=d:\
00,094 @rem set TEMP=d:\
00,095 @rem Of course, SSD if possible.
00,096
00,097 if exist %2.01.txt.sorted.7z del %2.01.txt.sorted.7z

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,098 if exist %2.02.txt.sorted.7z del %2.02.txt.sorted.7z
00,099 if exist %2.03.txt.sorted.7z del %2.03.txt.sorted.7z
00,100 if exist %2.04.txt.sorted.7z del %2.04.txt.sorted.7z
00,101 if exist %2.05.txt.sorted.7z del %2.05.txt.sorted.7z
00,102
00,103 7za_x86_v1900.exe a %2.01.txt.sorted.7z %2.01.txt.sorted
00,104 7za_x86_v1900.exe a %2.02.txt.sorted.7z %2.02.txt.sorted
00,105 7za_x86_v1900.exe a %2.03.txt.sorted.7z %2.03.txt.sorted
00,106 7za_x86_v1900.exe a %2.04.txt.sorted.7z %2.04.txt.sorted
00,107 7za_x86_v1900.exe a %2.05.txt.sorted.7z %2.05.txt.sorted
00,108
00,109 @del %2.lst
00,110 @del %2.01
00,111 @del %2.02
00,112 @del %2.03
00,113 @del %2.04
00,114 @del %2.05
00,115 @del Leprechaun.LOG
00,116
00,117 @echo.
00,118 @dir %2.07.txt.sorted*
00,119
00,120 :SkipRIP
00,121
00,122 copy %2.01.txt.sorted Gallowwalker.1.txt.sorted /y
00,123 copy %2.02.txt.sorted Gallowwalker.2.txt.sorted /y
00,124 copy %2.03.txt.sorted Gallowwalker.3.txt.sorted /y
00,125 copy %2.04.txt.sorted Gallowwalker.4.txt.sorted /y
00,126 copy %2.05.txt.sorted Gallowwalker.5.txt.sorted /y
00,127
00,128 Kazahana_Hexadecad_GCC_102_32bit.exe "%1" Gallowwalker.1.txt.sorted 1023
00,129 sort /R Kazahana.txt /O Kazahana_%1.1-1.txt
00,130 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_" Gallowwalker.2.txt.sorted 1023
00,131 sort /R Kazahana.txt /O Kazahana_%1.2-1.txt
00,132 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_" Gallowwalker.2.txt.sorted 1023
00,133 sort /R Kazahana.txt /O Kazahana_%1.2-2.txt
00,134 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.3.txt.sorted 1023
00,135 sort /R Kazahana.txt /O Kazahana_%1.3-1.txt
00,136 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.3.txt.sorted 1023
00,137 sort /R Kazahana.txt /O Kazahana_%1.3-2.txt
00,138 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.3.txt.sorted 1023
00,139 sort /R Kazahana.txt /O Kazahana_%1.3-3.txt
00,140 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.4.txt.sorted 1023
00,141 sort /R Kazahana.txt /O Kazahana_%1.4-1.txt
00,142 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.4.txt.sorted 1023
00,143 sort /R Kazahana.txt /O Kazahana_%1.4-2.txt
00,144 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.4.txt.sorted 1023
00,145 sort /R Kazahana.txt /O Kazahana_%1.4-3.txt
00,146 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_." Gallowwalker.4.txt.sorted 1023

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,147 sort /R Kazahana.txt /O Kazahana_%1.4-4.txt
00,148 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_._._." Gallowwalker.5.txt.sorted 1023
00,149 sort /R Kazahana.txt /O Kazahana_%1.5-1.txt
00,150 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_._._." Gallowwalker.5.txt.sorted 1023
00,151 sort /R Kazahana.txt /O Kazahana_%1.5-2.txt
00,152 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_._._." Gallowwalker.5.txt.sorted 1023
00,153 sort /R Kazahana.txt /O Kazahana_%1.5-3.txt
00,154 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_._._." Gallowwalker.5.txt.sorted 1023
00,155 sort /R Kazahana.txt /O Kazahana_%1.5-4.txt
00,156 Kazahana_Hexadecad_GCC_102_32bit.exe "%1_._._." Gallowwalker.5.txt.sorted 1023
00,157 sort /R Kazahana.txt /O Kazahana_%1.5-5.txt
00,158
00,159 dir Kazahana_%1.*.txt/b/on>q
00,160 LineJustify_PAGODAo5.exe q
00,161 copy Kazahana_%1.*.txt.PAD %2_%1.PAGODA-order-5.txt/b/y
00,162
00,163 Leprechaun_x-leton_64bit_Intel_01_001p.exe q q.wrd 14123456 Y
00,164 @if %errorlevel% == 0 goto OKwrd
00,165 Leprechaun_x-leton_64bit_Intel_01_008p.exe q q.wrd 14123456 Y
00,166 @if %errorlevel% == 0 goto OKwrd
00,167 Leprechaun_x-leton_64bit_Intel_01_512p.exe q q.wrd 14123456 Y
00,168 :OKwrd
00,169
00,170 sort /R q.wrd /O %2_%1.PAGODA-order-5.wrd
00,171
00,172 @del q
00,173 @del *.PAD
00,174 @del q.wrd
00,175 @del Kazahana.txt
00,176 @del Leprechaun.LOG
00,177 @del Kazahana_%1.*.txt
00,178
00,179 :Finish

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

qsm.h:
00,001 // File: qsm.h
00,002
00,003 // And for all QB64 fellow members who need maximum speed in their sorting, here comes the usage of the "header" written in C, ready to go in QB64:
00,004
00,005 //Declare CustomType Library "qsm"
00,006 // Sub Quicksort_QB64_v7 (ByVal QWORDS As _Offset, ByVal Left As _Integer64, ByVal Right As _Integer64)
00,007 //End Declare
00,008 //Quicksort_QB64_v7 _Offset(QWORDS"&&(LBound(QWORDS"&&))) , 0, UBound(QWORDS"&&) - LBound(QWORDS"&&)
00,009
00,010 /*
00,011 Declare CustomType Library
00,012 Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,013 End Declare
00,014
00,015 a$ = "1234567890"
00,016 b$ = "ABCDEFGHJIJ"
00,017
00,018 memcpy _Offset(a$) + 5, _Offset(b$) + 5, 5
00,019 Print a$
00,020 */
00,021
00,022 #include <stdint.h> // Needed for uint64_t
00,023
00,024 // Quicksort 'Magnetica' r.4, unbreakable (never overflowing the stack) when '#define FinishWithInsertionSort' is uncommented [
00,025 // Roughly speaking:
00,026 // - with (median of 1) nastiest quadratic  $O((N/1)^2)$  we have  $2(N/1)$  (due to Left+Right=2, a pair) stack usage;
00,027 // - with (median of 3) less nastier quadratic  $O((N/2)^2)$  we have  $2(N/2)$  (due to Left+Right=2, a pair) stack usage;
00,028 // - with (median of 5) even less nastier quadratic  $O((N/3)^2)$  we have  $2(N/3)$  (due to Left+Right=2, a pair) stack usage;
00,029 // - with (median of 7) even more less nastier quadratic  $O((N/4)^2)$  we have  $2(N/4)$  (due to Left+Right=2, a pair) stack usage;
00,030 // Value below allows  $2(N/4) = 2((\text{Right}-\text{Left}+1)/4) = (99999-1)$  i.e.  $N=2(99999-1)$  or ~200,000 elements
00,031 #define StackEntries 99999
00,032 // If below define is commented then we have "middle Pivot standalone Magnetica", otherwise "median of 3/7 Pivot InsertionSort Magnetica".
00,033 #define FinishWithInsertionSort
00,034 // Uncomment either 4 (fastest so far) or 3 (for reference only):
00,035 #define revision4
00,036 // #define revision3
00,037 // Thus we have 4 variants:
00,038 // Variant #1: "middle Pivot standalone Magnetica r.4"
00,039 // Variant #2: "middle Pivot standalone Magnetica r.3"
00,040 // Variant #3: "median of 3/7 Pivot InsertionSort Magnetica r.4"
00,041 // Variant #4: "median of 3/7 Pivot InsertionSort Magnetica r.3"
00,042 void swapUnconditional(uint64_t *a, uint64_t *b) { uint64_t t = *a; *a = *b; *b = t; }
00,043 // Many thanks go to all the coders at: https://stackoverflow.com/questions/2786899/fastest-sort-of-fixed-length-6-int-array
00,044 #define min50(x, y) (y ^ ((x ^ y) & -(x < y)))
00,045 #define max50(x, y) (x ^ ((x ^ y) & -(x < y)))
00,046 //
00,047 //
00,048 //

```

Listing: MASAKARI Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MzPlus ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,097      x5 = QWORDS[TheMiddleOfMiddle+5];
00,098      x6 = QWORDS[TheMiddleOfMiddle+6];
00,099      o0 = (x0>x1)+(x0>x2)+(x0>x3)+(x0>x4)+(x0>x5)+(x0>x6);
00,100      o1 = (x1>x0)+(x1>x2)+(x1>x3)+(x1>x4)+(x1>x5)+(x1>x6);
00,101      o2 = (x2>x0)+(x2>x1)+(x2>x3)+(x2>x4)+(x2>x5)+(x2>x6);
00,102      o3 = (x3>x0)+(x3>x1)+(x3>x2)+(x3>x4)+(x3>x5)+(x3>x6);
00,103      o4 = (x4>x0)+(x4>x1)+(x4>x2)+(x4>x3)+(x4>x5)+(x4>x6);
00,104      o5 = (x5>x0)+(x5>x1)+(x5>x2)+(x5>x3)+(x5>x4)+(x5>x6);
00,105      o6 = (0+1+2+3+4+5+6)-(o0+o1+o2+o3+o4+o5);
00,106      QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2; QWORDS[TheMiddleOfMiddle+o3]=x3; QWORDS[TheMiddleOfMiddle+o4]=x4; QWORDS[TheMiddleOfMiddle+o5]=x5;
QWORDS[TheMiddleOfMiddle+o6]=x6;
00,107      swapUnconditional (&QWORDS[TheMiddleOfMiddle+3], &QWORDS[PR]); //4th
00,108 }
00,109      Pivot = QWORDS[PR];
00,110      #else
00,111      swapUnconditional (&QWORDS[Left + Right]>>1], &QWORDS[PR]);
00,112      Pivot = QWORDS[PR];
00,113      #endif
00,114      #ifdef revision4
00,115      for (;PR < Jndx;) {
00,116          PR = PR + 1;
00,117          if (Pivot > QWORDS[PR]) {
00,118              swapUnconditional (&QWORDS[PL], &QWORDS[PR]);
00,119              PL = PL + 1;
00,120              //} else if (Pivot == QWORDS[PR]) {
00,121              } else if (Pivot < QWORDS[PR]) {
00,122                  for (;Pivot < QWORDS[Jndx];) {
00,123                      Jndx = Jndx - 1;
00,124                  }
00,125                  if (PR < Jndx) swapUnconditional (&QWORDS[PR], &QWORDS[Jndx]);
00,126                  Jndx = Jndx - 1;
00,127                  PR = PR - 1;
00,128              }
00,129          }
00,130      #endif
00,131 /*
00,132 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,133 ; ae-64+2=76 bytes i.e. (94-76=18 less), 5/1 conditional/unconditional jumps i.e. 2-1=1 less unconditional jump than r.1
00,134 ; 'Magnetica' partitioning, mainloop rev.2[
00,135 .B4.5::
00,136 00064 48 ff c5      inc rbp
00,137 00067 48 8b 14 e9    mov rdx, QWORD PTR [rcx+rbp*8]
00,138 0006b 4c 3b ea      cmp r13, rdx
00,139 0006e 77 2c        ja .B4.14
00,140 .B4.6::
00,141 00070 73 39        jae .B4.15
00,142 .B4.7::
00,143 00072 4e 8b 3c d9    mov r15, QWORD PTR [rcx+r11*8]
00,144 00076 4d 3b ef      cmp r13, r15

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,145 00079 73 0c      jae .B4.11
00,146 .B4.9::
00,147 0007b 49 ff cb      dec r11
00,148 0007e 4e 8b 3c d9    mov r15, QWORD PTR [rcx+r11*8]
00,149 00082 4d 3b ef      cmp r13, r15
00,150 00085 72 f4          jb .B4.9
00,151 .B4.11::
00,152 00087 49 3b eb      cmp rbp, r11
00,153 0008a 7d 08        jge .B4.13
00,154 .B4.12::
00,155 0008c 4c 89 3c e9    mov QWORD PTR [rcx+rbp*8], r15
00,156 00090 4a 89 14 d9    mov QWORD PTR [rcx+r11*8], rdx
00,157 .B4.13::
00,158 00094 49 ff cb      dec r11
00,159 00097 48 ff cd      dec rbp
00,160 0009a eb 0f        jmp .B4.15
00,161 .B4.14::
00,162 0009c 4e 8b 3c f1    mov r15, QWORD PTR [rcx+r14*8]
00,163 000a0 4a 89 14 f1    mov QWORD PTR [rcx+r14*8], rdx
00,164 000a4 49 ff c6      inc r14
00,165 000a7 4c 89 3c e9    mov QWORD PTR [rcx+rbp*8], r15
00,166 .B4.15::
00,167 000ab 49 3b eb      cmp rbp, r11
00,168 000ae 7c b4        jl .B4.5
00,169 ; 'Magnetica' partitioning, mainloop rev.2]
00,170 */
00,171 #ifdef revision3
00,172     for (;PR < Jndx;) {
00,173         // Many thanks go to Orson Peters for sharing his Pattern-defeating quicksort (pdqsort) at GitHub.
00,174         // This is due to a new technique described in "BlockQuicksort: How Branch Mispredictions don't affect Quicksort" by Stefan Edelkamp and Armin Weiss.
00,175         Stefan_Edelkamp_Armin_Weiss1 = (Pivot > QWORDS[PR + 1]);
00,176         Stefan_Edelkamp_Armin_Weiss2 = (Pivot == QWORDS[PR + 1]);
00,177         Stefan_Edelkamp_Armin_Weiss3 = (Pivot < QWORDS[PR + 1]);
00,178         tmp = minSO(QWORDS[PL], QWORDS[PR + 1]); QWORDS[PR + 1] = maxSO(QWORDS[PL], QWORDS[PR + 1]); QWORDS[PL] = tmp; // Since Pivot is equal to QWORDS[PL]
00,179         PL = PL + Stefan_Edelkamp_Armin_Weiss1;
00,180         PR = PR + Stefan_Edelkamp_Armin_Weiss1;
00,181         PR = PR + Stefan_Edelkamp_Armin_Weiss2;
00,182         if (Stefan_Edelkamp_Armin_Weiss3) {
00,183             for (;Pivot < QWORDS[Jndx];) {
00,184                 Jndx = Jndx - 1;
00,185             }
00,186             if (PR + 1 < Jndx) swapUnconditional (&QWORDS[PR + 1], &QWORDS[Jndx]);
00,187             Jndx = Jndx - 1;
00,188         }
00,189     }
00,190 #endif
00,191 /*
00,192 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,193 ; f9-73*6=140 bytes, 4/0 conditional/unconditional jumps

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,194 ; 'Magnetica' partitioning, mainloop rev.3[
00,195 B7.5::
00,196 00073 4a 8b 5c f1 08 mov rbx, QWORD PTR [8+rcx+r14*8]
00,197 00078 33 d2 xor edx, edx
00,198 0007a 4c 3b eb cmp r13, rbx
00,199 0007d 4a 8b 2c c1 mov rbp, QWORD PTR [rcx+r8*8]
00,200 00081 49 0f 47 d4 cmova rdx, r12
00,201 00085 45 33 ff xor r15d, r15d
00,202 00088 48 3b eb cmp rbp, rbx
00,203 0008b 48 89 ee mov rsi, rbp
00,204 0008e 41 0f 92 c7 setb r15b
00,205 00092 48 33 f3 xor rsi, rbx
00,206 00095 41 f7 df neg r15d
00,207 00098 4d 63 ff movsxd r15, r15d
00,208 0009b 49 23 f7 and rsi, r15
00,209 0009e 48 33 ee xor rbp, rsi
00,210 000a1 4c 3b eb cmp r13, rbx
00,211 000a4 4a 89 6c f1 08 mov QWORD PTR [8+rcx+r14*8], rbp
00,212 000a9 bd 00 00 00 00 mov ebp, 0
00,213 000ae 49 0f 44 ec cmove rbp, r12
00,214 000b2 48 33 f3 xor rsi, rbx
00,215 000b5 4a 89 34 c1 mov QWORD PTR [rcx+r8*8], rsi
00,216 000b9 4c 03 c2 add r8, rdx
00,217 000bc 48 03 d5 add rdx, rbp
00,218 000bf 4c 03 f2 add r14, rdx
00,219 000c2 4c 3b eb cmp r13, rbx
00,220 000c5 73 2f jae .B7.13
00,221 B7.6::
00,222 000c7 4a 8b 14 c9 mov rdx, QWORD PTR [rcx+r9*8]
00,223 000cb 4c 3b ea cmp r13, rdx
00,224 000ce 73 0c jae .B7.10
00,225 B7.8::
00,226 000d0 49 ff c9 dec r9
00,227 000d3 4a 8b 14 c9 mov rdx, QWORD PTR [rcx+r9*8]
00,228 000d7 4c 3b ea cmp r13, rdx
00,229 000da 72 f4 jb .B7.8
00,230 B7.10::
00,231 000dc 49 8d 5e 01 lea rbx, QWORD PTR [1+r14]
00,232 000e0 4c 3b cb cmp r9, rbx
00,233 000e3 7e 0e jle .B7.12
00,234 B7.11::
00,235 000e5 4a 8b 5c f1 08 mov rbx, QWORD PTR [8+rcx+r14*8]
00,236 000ea 4a 89 54 f1 08 mov QWORD PTR [8+rcx+r14*8], rdx
00,237 000ef 4a 89 1c c9 mov QWORD PTR [rcx+r9*8], rbx
00,238 B7.12:: ; Preds .B7.10 .B7.11
00,239 000f3 49 ff c9 dec r9
00,240 B7.13::
00,241 000f6 4d 3b f1 cmp r14, r9
00,242 000f9 0f 8c 74 ff ff

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,243      ff      jl .B7.5
00,244 ; 'Magnetica' partitioning, mainloop rev.3]
00,245 */
00,246      Jndx = PL - 1;
00,247      Indx = PR + 1;
00,248 /*
00,249      // 'Magnetica' partitioning [
00,250      Jndx = Right;
00,251      PL = Left;
00,252      PR = Left;
00,253      swap (&QWORDS[(Left + Right)>>1], &QWORDS[PR]);
00,254      Pivot = QWORDS[PR];
00,255      for (;PR < Jndx;) {
00,256          if (Pivot > QWORDS[PR + 1]) {
00,257              swap (&QWORDS[PL], &QWORDS[PR + 1]);
00,258              PL = PL + 1;
00,259              PR = PR + 1;
00,260          } else if (Pivot == QWORDS[PR + 1]) {
00,261              PR = PR + 1;
00,262          } else {
00,263              for (;Pivot < QWORDS[Jndx];) {
00,264                  Jndx = Jndx - 1;
00,265              }
00,266              if (PR + 1 < Jndx) swap (&QWORDS[PR + 1], &QWORDS[Jndx]);
00,267              Jndx = Jndx - 1;
00,268          }
00,269      }
00,270      Jndx = PL - 1;
00,271      Indx = PR + 1;
00,272      // 'Magnetica' partitioning ]
00,273 */
00,274 /*
00,275 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,276 ; c6-6a+2=94 bytes, 5/2 conditional/unconditional jumps
00,277 ; 'Magnetica' partitioning, mainloop [
00,278 .B4.5::
00,279 0006a 4e 3b 5c e9 08    cmp r11, QWORD PTR [8+rcx+r13*8]
00,280 0006f 77 37            ja .B4.15
00,281 .B4.6::
00,282 00071 75 08            jne .B4.8
00,283 .B4.7::
00,284 00073 49 89 d5          mov r13, rdx
00,285 00076 48 ff c2          inc rdx
00,286 00079 eb 48            jmp .B4.16
00,287 .B4.8::
00,288 0007b 4e 8b 34 c1        mov r14, QWORD PTR [rcx+r8*8]
00,289 0007f 4d 3b de          cmp r11, r14
00,290 00082 73 0c            jae .B4.12
00,291 .B4.10::

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,292 00004 49 ff c8      dec r8
00,293 00007 4e 8b 34 c1    mov r14, QWORD PTR [rcx+r8*8]
00,294 0000b 4d 3b de      cmp r11, r14
00,295 0000e 72 f4        jb .B4.10
00,296 .B4.12::
00,297 00009 4c 3b c2      cmp r8, rdx
00,298 00003 7e 0e        jle .B4.14
00,299 .B4.13::
00,300 00005 4e 8b 7c e9 08    mov r15, QWORD PTR [8+rcx+r13*8]
00,301 0000a 4e 89 74 e9 08    mov QWORD PTR [8+rcx+r13*8], r14
00,302 0000f 4e 89 3c c1      mov QWORD PTR [rcx+r8*8], r15
00,303 .B4.14::
00,304 000a3 49 ff c8      dec r8
00,305 000a6 eb 1b        jmp .B4.16
00,306 .B4.15::
00,307 000a8 4e 8b 74 e9 08    mov r14, QWORD PTR [8+rcx+r13*8]
00,308 000ad 4e 8b 3c e1      mov r15, QWORD PTR [rcx+r12*8]
00,309 000b1 4e 89 34 e1      mov QWORD PTR [rcx+r12*8], r14
00,310 000b5 49 ff c4        inc r12
00,311 000b8 4e 89 7c e9 08    mov QWORD PTR [8+rcx+r13*8], r15
00,312 000bd 49 89 d5      mov r13, rdx
00,313 000c0 48 ff c2      inc rdx
00,314 .B4.16::
00,315 000c3 4d 3b e8      cmp r13, r8
00,316 000c6 7c a2        jl .B4.5
00,317 ; 'Magnetica' partitioning, mainloop ]
00,318 */
00,319 #ifdef FinishWithInsertionSort
00,320     GearBox = ( max50((Right-Indx), (Jndx-Left)) > min50((Right-Indx), (Jndx-Left))<<6 ); // same as MAX/MIN > 64, i.e. Gear=1 if we need Median of 7
00,321 #endif
00,322     if (Indx + InsertionsortTHRESHOLD < Right) { // still refusing to go (always) with the smaller partition first...
00,323         StackPtr = StackPtr + 2;
00,324         Stack[StackPtr - 1] = Indx;
00,325         Stack[StackPtr] = Right;
00,326 #ifdef FinishWithInsertionSort
00,327         // Stack is full, bail out and finalize with Insertionsort [
00,328         StackPtr = StackPtr*(StackPtr + 2 <= StackEntries-1); // StackPtr should be enforced FALSE; also, ensure the next 'push' above is sanctioned - the max index of Stack[] being 'StackEntries-1'
00,329         Right = Right*(StackPtr + 2 <= StackEntries-1); // Left + InsertionsortTHRESHOLD < Right should be enforced FALSE:
00,330         // Stack is full, bail out and finalize with Insertionsort ]
00,331 #endif
00,332     }
00,333     Right = Jndx;
00,334     } //while (Left + InsertionsortTHRESHOLD < Right);
00,335 } while ( (StackPtr != 0) );
00,336 #ifdef FinishWithInsertionSort
00,337 for (Indx=LeftBackup+1; Indx <= RightBackup; Indx++) {
00,338     Jndx = Indx;
00,339     for (;Jndx >= 1;) {
00,340         if (QWORDS[Jndx-1] > QWORDS[Jndx]) swapUnconditional (&QWORDS[Jndx-1], &QWORDS[Jndx]); else break;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,341      Jndx = Jndx - 1;
00,342    }
00,343  }
00,344  #endif
00,345 }
00,346 // My threads with Magnetica's source and binaries (Linux and Windows):
00,347 // https://www.overclock.net/threads/benchmark-quicksort-says-sorting-2-billion-qwords.1794855/
00,348 // https://www.qb64.org/forum/index.php?topic=3518.msg137645#msg137645
00,349 // https://www.linuxquestions.org/questions/programming-9/qsrt-vs-%27magnetica%27-quicksort-4175703333/#post6299782
00,350 // Quicksort 'Magnetica' r.4, unbreakable (never overflowing the stack) when '#define FinishWithInsertionSort' is uncommented ]
00,351
00,352 // 113 lines of scalar C:
00,353 /*
00,354 00,001 #define StackEntries 99999
00,355 00,002 #define FinishWithInsertionSort
00,356 00,003 #define revision4
00,357 00,004 void swapUnconditional(uint64_t *a, uint64_t *b) { uint64_t t = *a; *a = *b; *b = t; }
00,358 00,005 #define min50(x, y) (y ^ ((x ^ y) & -(x < y)))
00,359 00,006 #define max50(x, y) (x ^ ((x ^ y) & -(x < y)))
00,360 00,007 //
00,361 00,008 // 
00,362 00,009 //
00,363 00,010 //
00,364 00,011 //
00,365 00,012 //
00,366 00,013 // Written by Sanmayce, 2021-Dec-02
00,367 00,014 void Quicksort_QB64_v7(uint64_t QWORDS[], int64_t Left, int64_t Right) {
00,368 00,015     #ifdef FinishWithInsertionSort
00,369 00,016         int InsertionsortTHRESHOLD = 17;
00,370 00,017     #else
00,371 00,018         int InsertionsortTHRESHOLD = 0;
00,372 00,019     #endif
00,373 00,020     int64_t Indx, Jndx, PL, PR;
00,374 00,021     int64_t Stack[StackEntries];
00,375 00,022     int64_t StackPtr = 0;
00,376 00,023     uint64_t Pivot;
00,377 00,024     int64_t LeftBackup = Left;
00,378 00,025     int64_t RightBackup = Right;
00,379 00,026     register uint64_t x0,x1,x2,x3,x4,x5,x6;
00,380 00,027     int o0,o1,o2,o3,o4,o5,o6;
00,381 00,028     int64_t TheMiddleOfMiddle;
00,382 00,029     int GearBox = 0; // 0 is Median of 3; 1 is Median of 7; Shifting to higher gear if BiggerPartition:SmallerPartition ratio is > 64:1
00,383 00,030     StackPtr++; Stack[StackPtr] = Left;
00,384 00,031     StackPtr++; Stack[StackPtr] = Right;
00,385 00,032     do {
00,386 00,033         Right = Stack[StackPtr];
00,387 00,034         Left = Stack[StackPtr - 1];
00,388 00,035         StackPtr = StackPtr - 2;
00,389 00,036         for(;(Left + InsertionsortTHRESHOLD < Right);) { // For instance, 1 + 17 < 19 i.e. (19-1)/4+6 < 19

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

00,390 00,037      Jndx = Right;
00,391 00,038      PL = Left;
00,392 00,039      PR = Left;
00,393 00,040      #ifndef FinishWithInsertionSort
00,394 00,041          TheMiddleOfMiddle = Left + ((Right-Left)>>2); // (4Left + Right - Left)/4; Caution: [TheMiddleOfMiddle+6] should be <= [Right] i.e. 6*(4Left + Right - Left)/4 <= Right or 4*6*(4Left + Right - Left) <=
4Right or 4*6 <= 3Right-3Left or (4*6)/3 <= Right-Left i.e. 8 <= Right-Left as a minimum condition to enter the 'for'.
00,395 00,042      x0 = QWORDS[TheMiddleOfMiddle+0];
00,396 00,043      x1 = QWORDS[TheMiddleOfMiddle+1];
00,397 00,044      x2 = QWORDS[TheMiddleOfMiddle+2];
00,398 00,045      if (GearBox == 0) { // Median of 5 proves more effective than 7, however 7 betters better the nastiest N^2
00,399 00,046          o0 = (x0>x1)+(x0>x2);
00,400 00,047          o1 = (x1>x0)+(x1>x2);
00,401 00,048          o2 = (0+1+2)-(o0+o1);
00,402 00,049          QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2;
00,403 00,050          swapUnconditional (&QWORDS[TheMiddleOfMiddle+1], &QWORDS[PR]); //2nd
00,404 00,051      } else {
00,405 00,052          x3 = QWORDS[TheMiddleOfMiddle+3];
00,406 00,053          x4 = QWORDS[TheMiddleOfMiddle+4];
00,407 00,054          x5 = QWORDS[TheMiddleOfMiddle+5];
00,408 00,055          x6 = QWORDS[TheMiddleOfMiddle+6];
00,409 00,056          o0 = (x0>x1)+(x0>x2)+(x0>x3)+(x0>x4)+(x0>x5)+(x0>x6);
00,410 00,057          o1 = (x1>x0)+(x1>x2)+(x1>x3)+(x1>x4)+(x1>x5)+(x1>x6);
00,411 00,058          o2 = (x2>x0)+(x2>x1)+(x2>x3)+(x2>x4)+(x2>x5)+(x2>x6);
00,412 00,059          o3 = (x3>x0)+(x3>x1)+(x3>x2)+(x3>x4)+(x3>x5)+(x3>x6);
00,413 00,060          o4 = (x4>x0)+(x4>x1)+(x4>x2)+(x4>x3)+(x4>x5)+(x4>x6);
00,414 00,061          o5 = (x5>x0)+(x5>x1)+(x5>x2)+(x5>x3)+(x5>x4)+(x5>x6);
00,415 00,062          o6 = (0+1+2+3+4+5+6)-(o0+o1+o2+o3+o4+o5);
00,416 00,063          QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2; QWORDS[TheMiddleOfMiddle+o3]=x3; QWORDS[TheMiddleOfMiddle+o4]=x4; QWORDS[TheMiddleOfMiddle+o5]=x5;
QWORDS[TheMiddleOfMiddle+o6]=x6;
00,417 00,064          swapUnconditional (&QWORDS[TheMiddleOfMiddle+3], &QWORDS[PR]); //4th
00,418 00,065      }
00,419 00,066      Pivot = QWORDS[PR];
00,420 00,067      #else
00,421 00,068          swapUnconditional (&QWORDS[(Left + Right)>>1], &QWORDS[PR]);
00,422 00,069          Pivot = QWORDS[PR];
00,423 00,070      #endif
00,424 00,071      #ifndef revision4
00,425 00,072          for (;PR < Jndx;) {
00,426 00,073              PR = PR + 1;
00,427 00,074              if (Pivot > QWORDS[PR]) {
00,428 00,075                  swapUnconditional (&QWORDS[PL], &QWORDS[PR]);
00,429 00,076                  PL = PL + 1;
00,430 00,077              } else if (Pivot < QWORDS[PR]) {
00,431 00,078                  for (;Pivot < QWORDS[Jndx];) {
00,432 00,079                      Jndx = Jndx - 1;
00,433 00,080                  }
00,434 00,081                  if (PR < Jndx) swapUnconditional (&QWORDS[PR], &QWORDS[Jndx]);
00,435 00,082                  Jndx = Jndx - 1;
00,436 00,083                  PR = PR - 1;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,437 00,084      }
00,438 00,085      }
00,439 00,086      #endif
00,440 00,087      Jndx = PL - 1;
00,441 00,088      Indx = PR + 1;
00,442 00,089      #ifdef FinishWithInsertionSort
00,443 00,090      GearBox = ( max50((Right-Indx), (Jndx-Left)) > min50((Right-Indx), (Jndx-Left))<<6 ); // same as MAX/MIN > 64, i.e. Gear=1 if we need Median of 7
00,444 00,091      #endif
00,445 00,092      if (Indx + InsertionsortTHRESHOLD < Right) { // still refusing to go (always) with the smaller partition first...
00,446 00,093          StackPtr = StackPtr + 2;
00,447 00,094          Stack[StackPtr - 1] = Indx;
00,448 00,095          Stack[StackPtr] = Right;
00,449 00,096      #ifdef FinishWithInsertionSort
00,450 00,097          StackPtr = StackPtr*(StackPtr + 2 <= StackEntries-1); // StackPtr should be enforced FALSE; also, ensure the next 'push' above is sanctioned - the max index of Stack[] being 'StackEntries-1'
00,451 00,098          Right = Right*(StackPtr + 2 <= StackEntries-1); // Left + InsertionsortTHRESHOLD < Right should be enforced FALSE;
00,452 00,099      #endif
00,453 00,100      }
00,454 00,101      Right = Jndx;
00,455 00,102      } //while (Left + InsertionsortTHRESHOLD < Right);
00,456 00,103      } while ( (StackPtr != 0) );
00,457 00,104      #ifdef FinishWithInsertionSort
00,458 00,105      for (Indx=LeftBackup+1; Indx <= RightBackup; Indx++) {
00,459 00,106          Jndx = Indx;
00,460 00,107          for (;Jndx >= 1;) {
00,461 00,108              if (QWORDS[Jndx-1] > QWORDS[Jndx]) swapUnconditional (&QWORDS[Jndx-1], &QWORDS[Jndx]); else break;
00,462 00,109              Jndx = Jndx - 1;
00,463 00,110          }
00,464 00,111      }
00,465 00,112      #endif
00,466 00,113 }
00,467 */
00,468
00,469 // 380 lines scalar Assembly:
00,470 /*
00,471 00,001 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,472 00,002 ; mark_description "-S -Os";
00,473 00,003
00,474 00,004 Quicksort_QB64_v7PROC
00,475 00,005 .B4.1::
00,476 00,006     push    rbx
00,477 00,007     push    rsi
00,478 00,008     push    rdi
00,479 00,009     push    r12
00,480 00,010     push    r13
00,481 00,011     push    r14
00,482 00,012     push    r15
00,483 00,013     push    rbp
00,484 00,014     mov     eax, 800152
00,485 00,015     call    __chkstk

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,486 00,016      sub      rsp, 800152
00,487 00,017      mov      r13, rcx
00,488 00,018      mov      QWORD PTR [800064+rsp], r8
00,489 00,019      xor      r14d, r14d
00,490 00,020      mov      QWORD PTR [40+rsp], rdx
00,491 00,021      mov      QWORD PTR [56+rsp], rdx
00,492 00,022      mov      QWORD PTR [32+rsp], 2
00,493 00,023      mov      QWORD PTR [64+rsp], r8
00,494 00,024      .B4.2::
00,495 00,025      mov      rax, QWORD PTR [32+rsp]
00,496 00,026      mov      rdx, QWORD PTR [40+rsp+rax*8]
00,497 00,027      mov      r12, QWORD PTR [48+rsp+rax*8]
00,498 00,028      add      rax, -2
00,499 00,029      mov      QWORD PTR [800040+rsp], rdx
00,500 00,030      mov      QWORD PTR [32+rsp], rax
00,501 00,031      lea      rcx, QWORD PTR [17+rdx]
00,502 00,032      mov      QWORD PTR [800072+rsp], rcx
00,503 00,033      cmp      r12, rcx
00,504 00,034      jle      .B4.27
00,505 00,035      .B4.3::
00,506 00,036      mov      QWORD PTR [800056+rsp], rdx
00,507 00,037      lea      rax, QWORD PTR [r13+rdx*8]
00,508 00,038      mov      QWORD PTR [800048+rsp], rax
00,509 00,039      .B4.4::
00,510 00,040      mov      r8, r12
00,511 00,041      xor      eax, eax
00,512 00,042      mov      rbp, QWORD PTR [800040+rsp]
00,513 00,043      sub      r8, rbp
00,514 00,044      sar      r8, 2
00,515 00,045      xor      edx, edx
00,516 00,046      add      r8, rbp
00,517 00,047      xor      r15d, r15d
00,518 00,048      mov      rbx, QWORD PTR [800056+rsp]
00,519 00,049      mov      rdi, r12
00,520 00,050      mov      QWORD PTR [800056+rsp], rbp
00,521 00,051      lea      rsi, QWORD PTR [rbp*8]
00,522 00,052      lea      rcx, QWORD PTR [r13+r8*8]
00,523 00,053      mov      r10, QWORD PTR [rcx]
00,524 00,054      mov      r11, QWORD PTR [8+rcx]
00,525 00,055      cmp      r10, r11
00,526 00,056      mov      r9, QWORD PTR [16+rcx]
00,527 00,057      seta      al
00,528 00,058      cmp      r10, r9
00,529 00,059      seta      dl
00,530 00,060      add      eax, edx
00,531 00,061      xor      edx, edx
00,532 00,062      cmp      r11, r10
00,533 00,063      setae     dl
00,534 00,064      cmp      r11, r9

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,535 00,065      seta      r15b
00,536 00,066      add       edx, r15d
00,537 00,067      test      r14d, r14d
00,538 00,068      jne       .B4.6
00,539 00,069 .B4.5::
00,540 00,070      add       rcx, 8
00,541 00,071      lea       r14, QWORD PTR [r8+rax]
00,542 00,072      mov      QWORD PTR [r13+r14*8], r10
00,543 00,073      lea       r10, QWORD PTR [r8+rdx]
00,544 00,074      sub       r8, rax
00,545 00,075      sub       r8, rdx
00,546 00,076      mov      QWORD PTR [r13+r10*8], r11
00,547 00,077      mov      QWORD PTR [24+r13+r8*8], r9
00,548 00,078      jmp       .B4.7
00,549 00,079 .B4.6::
00,550 00,080      mov      QWORD PTR [800104+rsp], rsi
00,551 00,081      xor      r14d, r14d
00,552 00,082      mov      rsi, QWORD PTR [24+rcx]
00,553 00,083      cmp      r10, rsi
00,554 00,084      mov      QWORD PTR [800096+rsp], rdi
00,555 00,085      mov      rdi, QWORD PTR [32+rcx]
00,556 00,086      seta     r14b
00,557 00,087      xor      r15d, r15d
00,558 00,088      cmp      r10, rdi
00,559 00,089      mov      QWORD PTR [800088+rsp], r12
00,560 00,090      push     0
00,561 00,091      pop      r12
00,562 00,092      seta     r12b
00,563 00,093      mov      QWORD PTR [800112+rsp], rbp
00,564 00,094      add      r14d, r12d
00,565 00,095      mov      rbp, QWORD PTR [40+rcx]
00,566 00,096      xor      r12d, r12d
00,567 00,097      cmp      r10, rbp
00,568 00,098      mov      QWORD PTR [800120+rsp], rbx
00,569 00,099      seta     r12b
00,570 00,100      mov      rbx, QWORD PTR [48+rcx]
00,571 00,101      cmp      r10, rbx
00,572 00,102      mov      QWORD PTR [800080+rsp], rcx
00,573 00,103      push     0
00,574 00,104      pop      rcx
00,575 00,105      seta     cl
00,576 00,106      add      eax, r14d
00,577 00,107      add      r12d, ecx
00,578 00,108      xor      ecx, ecx
00,579 00,109      cmp      r11, rsi
00,580 00,110      seta     cl
00,581 00,111      xor      r14d, r14d
00,582 00,112      cmp      r11, rdi
00,583 00,113      seta     r15b

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,584 00,114    cmp    r11, rbp
00,585 00,115    seta    r14b
00,586 00,116    add     eax, r12d
00,587 00,117    xor     r12d, r12d
00,588 00,118    cmp     r11, rbx
00,589 00,119    seta    r12b
00,590 00,120    add     ecx, r15d
00,591 00,121    add     edx, ecx
00,592 00,122    xor     ecx, ecx
00,593 00,123    cmp     r9, r10
00,594 00,124    setae   cl
00,595 00,125    xor     r15d, r15d
00,596 00,126    add     r14d, r12d
00,597 00,127    xor     r12d, r12d
00,598 00,128    cmp     r9, r11
00,599 00,129    setae   r12b
00,600 00,130    add     edx, r14d
00,601 00,131    add     ecx, r12d
00,602 00,132    xor     r12d, r12d
00,603 00,133    cmp     r9, rsi
00,604 00,134    seta    r12b
00,605 00,135    xor     r14d, r14d
00,606 00,136    cmp     r9, rdi
00,607 00,137    seta    r15b
00,608 00,138    add     r12d, r15d
00,609 00,139    xor     r15d, r15d
00,610 00,140    add     ecx, r12d
00,611 00,141    xor     r12d, r12d
00,612 00,142    cmp     r9, rbp
00,613 00,143    seta    r12b
00,614 00,144    cmp     r9, rbx
00,615 00,145    seta    r14b
00,616 00,146    cmp     rsi, r10
00,617 00,147    setae   r15b
00,618 00,148    add     r12d, r14d
00,619 00,149    xor     r14d, r14d
00,620 00,150    cmp     rsi, r11
00,621 00,151    setae   r14b
00,622 00,152    add     ecx, r12d
00,623 00,153    add     r15d, r14d
00,624 00,154    xor     r14d, r14d
00,625 00,155    cmp     rsi, r9
00,626 00,156    setae   r14b
00,627 00,157    xor     r12d, r12d
00,628 00,158    cmp     rsi, rdi
00,629 00,159    seta    r12b
00,630 00,160    add     r14d, r12d
00,631 00,161    xor     r12d, r12d
00,632 00,162    add     r15d, r14d

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,633 00,163    xor     r14d, r14d
00,634 00,164    cmp     rsi, rbp
00,635 00,165    seta    r14b
00,636 00,166    cmp     rsi, rbx
00,637 00,167    seta    r12b
00,638 00,168    add     r14d, r12d
00,639 00,169    xor     r12d, r12d
00,640 00,170    add     r15d, r14d
00,641 00,171    xor     r14d, r14d
00,642 00,172    cmp     rdi, r10
00,643 00,173    setae   r14b
00,644 00,174    cmp     rdi, r11
00,645 00,175    setae   r12b
00,646 00,176    add     r14d, r12d
00,647 00,177    xor     r12d, r12d
00,648 00,178    cmp     rdi, r9
00,649 00,179    setae   r12b
00,650 00,180    cmp     rdi, rsi
00,651 00,181    mov     QWORD PTR [800128+rsp], r15
00,652 00,182    push    0
00,653 00,183    pop     r15
00,654 00,184    setae   r15b
00,655 00,185    add     r12d, r15d
00,656 00,186    xor     r15d, r15d
00,657 00,187    add     r14d, r12d
00,658 00,188    xor     r12d, r12d
00,659 00,189    cmp     rdi, rbp
00,660 00,190    seta    r12b
00,661 00,191    cmp     rdi, rbx
00,662 00,192    seta    r15b
00,663 00,193    add     r12d, r15d
00,664 00,194    xor     r15d, r15d
00,665 00,195    add     r14d, r12d
00,666 00,196    xor     r12d, r12d
00,667 00,197    cmp     rbp, r10
00,668 00,198    setae   r12b
00,669 00,199    cmp     rbp, r11
00,670 00,200    setae   r15b
00,671 00,201    add     r12d, r15d
00,672 00,202    xor     r15d, r15d
00,673 00,203    cmp     rbp, r9
00,674 00,204    setae   r15b
00,675 00,205    cmp     rbp, rsi
00,676 00,206    mov     QWORD PTR [800136+rsp], r14
00,677 00,207    push    0
00,678 00,208    pop     r14
00,679 00,209    setae   r14b
00,680 00,210    add     r15d, r14d
00,681 00,211    xor     r14d, r14d

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,682 00,212      add     r12d, r15d
00,683 00,213      xor     r15d, r15d
00,684 00,214      cmp     rbp, rdi
00,685 00,215      setae  r15b
00,686 00,216      cmp     rbp, rbx
00,687 00,217      seta   r14b
00,688 00,218      add     r15d, r14d
00,689 00,219      lea     r14, QWORD PTR [r8+rax]
00,690 00,220      mov     QWORD PTR [r13+r14*8], r10
00,691 00,221      lea     r10, QWORD PTR [r8+rdx]
00,692 00,222      mov     QWORD PTR [r13+r10*8], r11
00,693 00,223      lea     r11, QWORD PTR [r8+rcx]
00,694 00,224      mov     r10, QWORD PTR [800128+rsp]
00,695 00,225      add     r12d, r15d
00,696 00,226      mov     QWORD PTR [r13+r11*8], r9
00,697 00,227      add     rcx, r10
00,698 00,228      mov     r11, QWORD PTR [800136+rsp]
00,699 00,229      lea     r9, QWORD PTR [r8+r10]
00,700 00,230      mov     QWORD PTR [r13+r9*8], rsi
00,701 00,231      lea     rsi, QWORD PTR [r8+r11]
00,702 00,232      add     r11, r12
00,703 00,233      mov     QWORD PTR [r13+rsi*8], rdi
00,704 00,234      add     rcx, r11
00,705 00,235      mov     rsi, QWORD PTR [800104+rsp]
00,706 00,236      lea     rdi, QWORD PTR [r8+r12]
00,707 00,237      sub     r8, rax
00,708 00,238      sub     r8, rdx
00,709 00,239      sub     r8, rcx
00,710 00,240      mov     QWORD PTR [r13+rdi*8], rbp
00,711 00,241      mov     rcx, QWORD PTR [800080+rsp]
00,712 00,242      mov     rbp, QWORD PTR [800112+rsp]
00,713 00,243      add     rcx, 24
00,714 00,244      mov     QWORD PTR [168+r13+r8*8], rbx
00,715 00,245      mov     rbx, QWORD PTR [800120+rsp]
00,716 00,246      mov     rdi, QWORD PTR [800096+rsp]
00,717 00,247      mov     r12, QWORD PTR [800088+rsp]
00,718 00,248      .B4.7::
00,719 00,249      mov     rdx, QWORD PTR [800048+rsp]
00,720 00,250      call   swapUnconditional
00,721 00,251      .B4.8::
00,722 00,252      mov     rax, QWORD PTR [800048+rsp]
00,723 00,253      cmp     r12, QWORD PTR [800040+rsp]
00,724 00,254      mov     r14, QWORD PTR [rax]
00,725 00,255      jle     .B4.23
00,726 00,256      .B4.10::
00,727 00,257      inc     rbp
00,728 00,258      lea     rdx, QWORD PTR [r13+rbp*8]
00,729 00,259      cmp     r14, QWORD PTR [rdx]
00,730 00,260      jbe     .B4.13

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,731 00,261 .B4.11::
00,732 00,262     lea     rcx, QWORD PTR [r13+rsi]
00,733 00,263     call   swapUnconditional
00,734 00,264 .B4.12::
00,735 00,265     add     rsi, 8
00,736 00,266     inc     rbx
00,737 00,267     jmp     .B4.21
00,738 00,268 .B4.13::
00,739 00,269     jae     .B4.21
00,740 00,270 .B4.14::
00,741 00,271     lea     rax, QWORD PTR [r13+rdi*8]
00,742 00,272     cmp     r14, QWORD PTR [rax]
00,743 00,273     jae     .B4.18
00,744 00,274 .B4.16::
00,745 00,275     dec     rdi
00,746 00,276     lea     rax, QWORD PTR [r13+rdi*8]
00,747 00,277     cmp     r14, QWORD PTR [rax]
00,748 00,278     jb     .B4.16
00,749 00,279 .B4.18::
00,750 00,280     cmp     rbp, rdi
00,751 00,281     jge     .B4.20
00,752 00,282 .B4.19::
00,753 00,283     mov     rcx, rdx
00,754 00,284     mov     rdx, rax
00,755 00,285     call   swapUnconditional
00,756 00,286 .B4.20::
00,757 00,287     dec     rdi
00,758 00,288     dec     rbp
00,759 00,289 .B4.21::
00,760 00,290     cmp     rbp, rdi
00,761 00,291     jl     .B4.10
00,762 00,292 .B4.23::
00,763 00,293     dec     rbx
00,764 00,294     lea     r9, QWORD PTR [1+rbp]
00,765 00,295     mov     rdi, r12
00,766 00,296     mov     r8, rbx
00,767 00,297     sub     r8, QWORD PTR [800040+rsp]
00,768 00,298     sub     rdi, r9
00,769 00,299     xor     edx, edx
00,770 00,300     cmp     rdi, r8
00,771 00,301     mov     rcx, rdi
00,772 00,302     setl    dl
00,773 00,303     xor     r14d, r14d
00,774 00,304     xor     rcx, r8
00,775 00,305     neg     edx
00,776 00,306     add     rbp, 18
00,777 00,307     movsxd  rdx, edx
00,778 00,308     and     rcx, rdx
00,779 00,309     xor     r8, rcx

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,780 00,310      xor      rdi, rcx
00,781 00,311      shl      r8, 6
00,782 00,312      cmp      rdi, r8
00,783 00,313      setg     r14b
00,784 00,314      cmp      r12, rbp
00,785 00,315      jle      .B4.25
00,786 00,316 .B4.24::
00,787 00,317      mov      rax, QWORD PTR [32+rsp]
00,788 00,318      cmp      rax, 99994
00,789 00,319      mov      QWORD PTR [64+rsp+rax*8], r12
00,790 00,320      push     0
00,791 00,321      pop      r12
00,792 00,322      settle  r12b
00,793 00,323      mov      QWORD PTR [56+rsp+rax*8], r9
00,794 00,324      add      rax, 2
00,795 00,325      imul     rax, r12
00,796 00,326      mov      QWORD PTR [32+rsp], rax
00,797 00,327 .B4.25::
00,798 00,328      mov      r12, rbx
00,799 00,329      cmp      rbx, QWORD PTR [800072+rsp]
00,800 00,330      jg       .B4.4
00,801 00,331 .B4.27::
00,802 00,332      cmp      QWORD PTR [32+rsp], 0
00,803 00,333      jne      .B4.2
00,804 00,334 .B4.28::
00,805 00,335      mov      rax, QWORD PTR [40+rsp]
00,806 00,336      inc      rax
00,807 00,337      mov      QWORD PTR [40+rsp], rax
00,808 00,338      cmp      rax, QWORD PTR [800064+rsp]
00,809 00,339      jg       .B4.39
00,810 00,340 .B4.30::
00,811 00,341      mov      rbx, QWORD PTR [40+rsp]
00,812 00,342      test     rbx, rbx
00,813 00,343      jle      .B4.28
00,814 00,344 .B4.32::
00,815 00,345      mov      rax, QWORD PTR [-8+r13+rbx*8]
00,816 00,346      cmp      rax, QWORD PTR [r13+rbx*8]
00,817 00,347      jbe      .B4.28
00,818 00,348 .B4.33::
00,819 00,349      lea      rdx, QWORD PTR [r13+rbx*8]
00,820 00,350      lea      rcx, QWORD PTR [-8+rdx]
00,821 00,351      call     swapUnconditional
00,822 00,352 .B4.34::
00,823 00,353      dec      rbx
00,824 00,354      test     rbx, rbx
00,825 00,355      jg       .B4.32
00,826 00,356      jmp      .B4.28
00,827 00,357 .B4.39::
00,828 00,358      add      rsp, 800152

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,829 00,359      pop      rbp
00,830 00,360      pop      r15
00,831 00,361      pop      r14
00,832 00,362      pop      r13
00,833 00,363      pop      r12
00,834 00,364      pop      rdi
00,835 00,365      pop      rsi
00,836 00,366      pop      rbx
00,837 00,367      ret
00,838 00,368 .B4.40::
00,839 00,369 Quicksort_QB64_v7 ENDP
00,840 00,370
00,841 00,371 swapUnconditionalPROC
00,842 00,372 .B5.1::
00,843 00,373      mov      rax, QWORD PTR [rdx]
00,844 00,374      mov      r8, QWORD PTR [rcx]
00,845 00,375      mov      QWORD PTR [rcx], rax
00,846 00,376      mov      QWORD PTR [rdx], r8
00,847 00,377 .B5.4::
00,848 00,378      ret
00,849 00,379 .B5.2::
00,850 00,380 swapUnconditional ENDP
00,851 */
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

### qsm\_linesize.h:

```
00,001 // File: qsm_linesize.h
00,002
00,003 // And for all QB64 fellow members who need maximum speed in their sorting, here comes the usage of the "header" written in C, ready to go in QB64:
00,004
00,005 // Testfile:
00,006 // https://dumps.wikimedia.org/enwiki/20211201/enwiki-20211201-pages-articles.xml.bz2 (17.9 GB)
00,007
00,008 //Declare CustomType Library "qsm_linesize"
00,009 // Sub Quicksort_QB64_v7_linesize (ByVal QWORDSoFF As _Offset, ByVal QWORDSLen As _Offset, Byval Left As _Integer64, Byval Right As _Integer64)
00,010 //End Declare
00,011 //Quicksort_QB64_v7_linesize MhandleOFF.OFFSET, MhandleLEN.OFFSET, 0, ElementsMinusOne
00,012
00,013 /*
00,014 Declare CustomType Library
00,015 Sub memcpy (ByVal dest As _Offset, Byval source As _Offset, Byval bytes As Long)
00,016 End Declare
00,017
00,018 a$ = "1234567890"
00,019 b$ = "ABCDEFGHJIJ"
00,020
00,021 memcpy _Offset(a$) + 5, _Offset(b$) + 5, 5
00,022 Print a$
00,023 */
00,024
00,025 #include <stdint.h> // Needed for uint64_t
00,026
00,027 // Quicksort 'Magnetica' r.4, unbreakable (never overflowing the stack) when '#define FinishWithInsertionSort' is uncommented [
00,028 // Roughly speaking:
00,029 // - with (median of 1) nastiest quadratic  $O((N/1)^2)$  we have  $2(N/1)$  (due to Left+Right=2, a pair) stack usage;
00,030 // - with (median of 3) less nastier quadratic  $O((N/2)^2)$  we have  $2(N/2)$  (due to Left+Right=2, a pair) stack usage;
00,031 // - with (median of 5) even less nastier quadratic  $O((N/3)^2)$  we have  $2(N/3)$  (due to Left+Right=2, a pair) stack usage;
00,032 // - with (median of 7) even more less nastier quadratic  $O((N/4)^2)$  we have  $2(N/4)$  (due to Left+Right=2, a pair) stack usage;
00,033 // Value below allows  $2(N/4) = 2((\text{Right}-\text{Left}+1)/4) = (99999-1)$  i.e.  $N=2(99999-1)$  or ~200,000 elements
00,034 #define StackEntries 99999
00,035 // If below define is commented then we have "middle Pivot standalone Magnetica", otherwise "median of 3/7 Pivot InsertionSort Magnetica".
00,036 #define FinishWithInsertionSort
00,037 // Uncomment either 4 (fastest so far) or 3 (for reference only):
00,038 #define revision4
00,039 // #define revision3
00,040 // Thus we have 4 variants:
00,041 // Variant #1: "middle Pivot standalone Magnetica r.4"
00,042 // Variant #2: "middle Pivot standalone Magnetica r.3"
00,043 // Variant #3: "median of 3/7 Pivot InsertionSort Magnetica r.4"
00,044 // Variant #4: "median of 3/7 Pivot InsertionSort Magnetica r.3"
00,045 //void swapUnconditional(uint64_t *a, uint64_t *b) { uint64_t t = *a; *a = *b; *b = t; }
00,046 // Many thanks go to all the coders at: https://stackoverflow.com/questions/2786899/fastest-sort-of-fixed-length-6-int-array
00,047 #define min50(x, y) (y ^ ((x ^ y) & -(x < y)))
00,048 #define max50(x, y) (x ^ ((x ^ y) & -(x < y)))
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 //
00,050 //
00,051 //
00,052 //
00,053 //
00,054 //
00,055 // Written by Sanmayce, 2021-Dec-02
00,056 // The indexes are signed, but the elements are unsigned, targeting 64bit pointers mostly.
00,057 void Quicksort_QB64_v7_linesize(uint64_t QWORDSoFF[], uint64_t QWORDS[], int64_t Left, int64_t Right) {
00,058     #ifdef FinishWithInsertionSort
00,059         int InsertionsortTHRESHOLD = 17;
00,060     #else
00,061         int InsertionsortTHRESHOLD = 0;
00,062     #endif
00,063     int64_t Indx, Jndx, PL, PR;
00,064     int64_t Stack[StackEntries];
00,065     int64_t StackPtr = 0;
00,066     uint64_t Pivot;
00,067     int64_t LeftBackup = Left;
00,068     int64_t RightBackup = Right;
00,069     register uint64_t x0,x1,x2,x3,x4,x5,x6;
00,070     register uint64_t x0off,x1off,x2off,x3off,x4off,x5off,x6off;
00,071     int o0,o1,o2,o3,o4,o5,o6;
00,072     int64_t TheMiddleOfMiddle;
00,073     int Stefan_Edelkamp_Armin_Weiss1, Stefan_Edelkamp_Armin_Weiss2, Stefan_Edelkamp_Armin_Weiss3;
00,074     uint64_t tmp;
00,075     int GearBox = 0; // 0 is Median of 3; 1 is Median of 7; Shifting to higher gear if BiggerPartition:SmallerPartition ratio is > 64:1
00,076     StackPtr++; Stack[StackPtr] = Left;
00,077     StackPtr++; Stack[StackPtr] = Right;
00,078     do {
00,079         Right = Stack[StackPtr];
00,080         Left = Stack[StackPtr - 1];
00,081         StackPtr = StackPtr - 2;
00,082         //do {
00,083             for(;(Left + InsertionsortTHRESHOLD < Right);) { // For instance, 1 + 17 < 19 i.e. (19-1)/4+6 < 19
00,084                 Jndx = Right;
00,085                 PL = Left;
00,086                 PR = Left;
00,087             #ifdef FinishWithInsertionSort
00,088                 TheMiddleOfMiddle = Left + ((Right-Left)>>2); // (4Left + Right - Left)/4; Caution: [TheMiddleOfMiddle+6] should be <= [Right] i.e. 6*(4Left + Right - Left)/4 <= Right or 4*6*(4Left + Right - Left) <= 4Right
00,089                 x0 = QWORDS[TheMiddleOfMiddle+0];
00,090                 x1 = QWORDS[TheMiddleOfMiddle+1];
00,091                 x2 = QWORDS[TheMiddleOfMiddle+2];
00,092                 x0off = QWORDSoFF[TheMiddleOfMiddle+0];
00,093                 x1off = QWORDSoFF[TheMiddleOfMiddle+1];
00,094                 x2off = QWORDSoFF[TheMiddleOfMiddle+2];
00,095             if (GearBox == 0) { // Median of 5 proves more effective than 7, however 7 betters better the nastiest N^2
00,096                 o0 = (x0>x1)+(x0>x2);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,097      o1 = (x1>=x0)+(x1>x2);
00,098      o2 = (0+1+2)-(o0+o1);
00,099      QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2;
00,100      QWORDSOFF[TheMiddleOfMiddle+o0]=x0off; QWORDSOFF[TheMiddleOfMiddle+o1]=x1off; QWORDSOFF[TheMiddleOfMiddle+o2]=x2off;
00,101      swapUnconditional (&QWORDS[TheMiddleOfMiddle+1], &QWORDS[PR]); //2nd
00,102      swapUnconditional (&QWORDSOFF[TheMiddleOfMiddle+1], &QWORDSOFF[PR]); //2nd
00,103 } else {
00,104      x3 = QWORDS[TheMiddleOfMiddle+3];
00,105      x4 = QWORDS[TheMiddleOfMiddle+4];
00,106      x5 = QWORDS[TheMiddleOfMiddle+5];
00,107      x6 = QWORDS[TheMiddleOfMiddle+6];
00,108      x3off = QWORDSOFF[TheMiddleOfMiddle+3];
00,109      x4off = QWORDSOFF[TheMiddleOfMiddle+4];
00,110      x5off = QWORDSOFF[TheMiddleOfMiddle+5];
00,111      x6off = QWORDSOFF[TheMiddleOfMiddle+6];
00,112      o0 = (x0>x1)+(x0>x2)+(x0>x3)+(x0>x4)+(x0>x5)+(x0>x6);
00,113      o1 = (x1>=x0)+(x1>x2)+(x1>x3)+(x1>x4)+(x1>x5)+(x1>x6);
00,114      o2 = (x2>=x0)+(x2>=x1)+(x2>x3)+(x2>x4)+(x2>x5)+(x2>x6);
00,115      o3 = (x3>=x0)+(x3>=x1)+(x3>=x2)+(x3>x4)+(x3>x5)+(x3>x6);
00,116      o4 = (x4>=x0)+(x4>=x1)+(x4>=x2)+(x4>=x3)+(x4>x5)+(x4>x6);
00,117      o5 = (x5>=x0)+(x5>=x1)+(x5>=x2)+(x5>=x3)+(x5>=x4)+(x5>x6);
00,118      o6 = (0+1+2+3+4+5+6)-(o0+o1+o2+o3+o4+o5);
00,119      QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2; QWORDS[TheMiddleOfMiddle+o3]=x3; QWORDS[TheMiddleOfMiddle+o4]=x4; QWORDS[TheMiddleOfMiddle+o5]=x5;
QWORDS[TheMiddleOfMiddle+o6]=x6;
00,120      QWORDSOFF[TheMiddleOfMiddle+o0]=x0off; QWORDSOFF[TheMiddleOfMiddle+o1]=x1off; QWORDSOFF[TheMiddleOfMiddle+o2]=x2off; QWORDSOFF[TheMiddleOfMiddle+o3]=x3off; QWORDSOFF[TheMiddleOfMiddle+o4]=x4off;
QWORDSOFF[TheMiddleOfMiddle+o5]=x5off; QWORDSOFF[TheMiddleOfMiddle+o6]=x6off;
00,121      swapUnconditional (&QWORDS[TheMiddleOfMiddle+3], &QWORDS[PR]); //4th
00,122      swapUnconditional (&QWORDSOFF[TheMiddleOfMiddle+3], &QWORDSOFF[PR]); //4th
00,123 }
00,124      Pivot = QWORDS[PR];
00,125 #else
00,126      swapUnconditional (&QWORDS[(Left + Right)>>1], &QWORDS[PR]);
00,127      swapUnconditional (&QWORDSOFF[(Left + Right)>>1], &QWORDSOFF[PR]);
00,128      Pivot = QWORDS[PR];
00,129 #endif
00,130 #ifdef revision4
00,131      for (;PR < Jndx;) {
00,132          PR = PR + 1;
00,133          if (Pivot > QWORDS[PR]) {
00,134              swapUnconditional (&QWORDS[PL], &QWORDS[PR]);
00,135              swapUnconditional (&QWORDSOFF[PL], &QWORDSOFF[PR]);
00,136              PL = PL + 1;
00,137          } else if (Pivot == QWORDS[PR]) {
00,138          } else if (Pivot < QWORDS[PR]) {
00,139              for (;Pivot < QWORDS[Jndx];) {
00,140                  Jndx = Jndx - 1;
00,141              }
00,142              if (PR < Jndx) { swapUnconditional (&QWORDS[PR], &QWORDS[Jndx]); swapUnconditional (&QWORDSOFF[PR], &QWORDSOFF[Jndx]); }
00,143              Jndx = Jndx - 1;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,144      PR = PR - 1;
00,145      }
00,146      }
00,147      #endif
00,148 /*
00,149 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,150 ; ae-64+2=76 bytes i.e. (94-76=18 less), 5/1 conditional/unconditional jumps i.e. 2-1=1 less unconditional jump than r.1
00,151 ; 'Magnetica' partitioning, mainloop rev.2]
00,152 .B4.5::
00,153 00064 48 ff c5      inc rbp
00,154 00067 48 8b 14 e9    mov rdx, QWORD PTR [rcx+rbp*8]
00,155 0006b 4c 3b ea      cmp r13, rdx
00,156 0006e 77 2c        ja .B4.14
00,157 .B4.6::
00,158 00070 73 39        jae .B4.15
00,159 .B4.7::
00,160 00072 4e 8b 3c d9    mov r15, QWORD PTR [rcx+r11*8]
00,161 00076 4d 3b ef      cmp r13, r15
00,162 00079 73 0c        jae .B4.11
00,163 .B4.9::
00,164 0007b 49 ff cb      dec r11
00,165 0007e 4e 8b 3c d9    mov r15, QWORD PTR [rcx+r11*8]
00,166 00082 4d 3b ef      cmp r13, r15
00,167 00085 72 f4        jb .B4.9
00,168 .B4.11::
00,169 00087 49 3b eb      cmp rbp, r11
00,170 0008a 7d 08        jge .B4.13
00,171 .B4.12::
00,172 0008c 4c 89 3c e9    mov QWORD PTR [rcx+rbp*8], r15
00,173 00090 4a 89 14 d9    mov QWORD PTR [rcx+r11*8], rdx
00,174 .B4.13::
00,175 00094 49 ff cb      dec r11
00,176 00097 48 ff cd      dec rbp
00,177 0009a eb 0f        jmp .B4.15
00,178 .B4.14::
00,179 0009c 4e 8b 3c f1    mov r15, QWORD PTR [rcx+r14*8]
00,180 000a0 4a 89 14 f1    mov QWORD PTR [rcx+r14*8], rdx
00,181 000a4 49 ff c6      inc r14
00,182 000a7 4c 89 3c e9    mov QWORD PTR [rcx+rbp*8], r15
00,183 .B4.15::
00,184 000ab 49 3b eb      cmp rbp, r11
00,185 000ae 7c b4        jl .B4.5
00,186 ; 'Magnetica' partitioning, mainloop rev.2]
00,187 */
00,188 #ifdef revision3
00,189     for (;PR < Jndx;) {
00,190         // Many thanks go to Orson Peters for sharing his Pattern-defeating quicksort (pdqsort) at GitHub.
00,191         // This is due to a new technique described in "BlockQuicksort: How Branch Mispredictions don't affect Quicksort" by Stefan Edelkamp and Armin Weiss.
00,192         Stefan_Edelkamp_Armin_Weiss1 = (Pivot > QWORDS[PR + 1]);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,193      Stefan_Edelkamp_Armin_Weiss2 = (Pivot == QWORDS[PR + 1]);
00,194      Stefan_Edelkamp_Armin_Weiss3 = (Pivot < QWORDS[PR + 1]);
00,195      tmp = minSO(QWORDS[PL], QWORDS[PR + 1]); QWORDS[PR + 1] = maxSO(QWORDS[PL], QWORDS[PR + 1]); QWORDS[PL] = tmp; // Since Pivot is equal to QWORDS[PL]
00,196      PL = PL + Stefan_Edelkamp_Armin_Weiss1;
00,197      PR = PR + Stefan_Edelkamp_Armin_Weiss1;
00,198      PR = PR + Stefan_Edelkamp_Armin_Weiss2;
00,199      if (Stefan_Edelkamp_Armin_Weiss3) {
00,200          for (;Pivot < QWORDS[Jndx];) {
00,201              Jndx = Jndx - 1;
00,202          }
00,203          if (PR + 1 < Jndx) swapUnconditional (&QWORDS[PR + 1], &QWORDS[Jndx]);
00,204          Jndx = Jndx - 1;
00,205      }
00,206  }
00,207  #endif
00,208 /*
00,209 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,210 ; f9-73+6=140 bytes, 4/0 conditional/unconditional jumps
00,211 ; 'Magnetica' partitioning, mainloop rev.3[
00,212 .B7.5::
00,213 00073 4a 8b 5c f1 08  mov rbx, QWORD PTR [8+rcx+r14*8]
00,214 00078 33 d2           xor edx, edx
00,215 0007a 4c 3b eb        cmp r13, rbx
00,216 0007d 4a 8b 2c c1     mov rbp, QWORD PTR [rcx+r8*8]
00,217 00081 49 0f 47 d4      cmova rdx, r12
00,218 00085 45 33 ff       xor r15d, r15d
00,219 00088 48 3b eb        cmp rbp, rbx
00,220 0008b 48 89 ee        mov rsi, rbp
00,221 0008e 41 0f 92 c7      setb r15b
00,222 00092 48 33 f3        xor rsi, rbx
00,223 00095 41 f7 df        neg r15d
00,224 00098 4d 63 ff        movsxd r15, r15d
00,225 0009b 49 23 f7          and rsi, r15
00,226 0009e 48 33 ee          xor rbp, rsi
00,227 000a1 4c 3b eb        cmp r13, rbx
00,228 000a4 4a 89 6c f1 08  mov QWORD PTR [8+rcx+r14*8], rbp
00,229 000a9 bd 00 00 00 00  mov ebp, 0
00,230 000ae 49 0f 44 ec      cmove rbp, r12
00,231 000b2 48 33 f3        xor rsi, rbx
00,232 000b5 4a 89 34 c1     mov QWORD PTR [rcx+r8*8], rsi
00,233 000b9 4c 03 c2        add r8, rdx
00,234 000bc 48 03 d5        add rdx, rbp
00,235 000bf 4c 03 f2        add r14, rdx
00,236 000c2 4c 3b eb        cmp r13, rbx
00,237 000c5 73 2f         jae .B7.13
00,238 .B7.6::
00,239 000c7 4a 8b 14 c9      mov rdx, QWORD PTR [rcx+r9*8]
00,240 000cb 4c 3b ea        cmp r13, rdx
00,241 000ce 73 0c         jae .B7.10

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,242 .B7.8::
00,243 000d0 49 ff c9      dec r9
00,244 000d3 4a 8b 14 c9  mov rdx, QWORD PTR [rcx+r9*8]
00,245 000d7 4c 3b ea      cmp r13, rdx
00,246 000da 72 f4        jb .B7.8
00,247 .B7.10::
00,248 000dc 49 8d 5e 01    lea rbx, QWORD PTR [1+r14]
00,249 000e0 4c 3b cb      cmp r9, rbx
00,250 000e3 7e 0e        jle .B7.12
00,251 .B7.11::
00,252 000e5 4a 8b 5c f1 08  mov rbx, QWORD PTR [8+rcx+r14*8]
00,253 000ea 4a 89 54 f1 08  mov QWORD PTR [8+rcx+r14*8], rdx
00,254 000ef 4a 89 1c c9    mov QWORD PTR [rcx+r9*8], rbx
00,255 .B7.12::          ; Preds .B7.10 .B7.11
00,256 000f3 49 ff c9      dec r9
00,257 .B7.13::
00,258 000f6 4d 3b f1      cmp r14, r9
00,259 000f9 0f 8c 74 ff ff
00,260 ff                jl .B7.5
00,261 ; 'Magnetica' partitioning, mainloop rev.3]
00,262 */
00,263         Jndx = PL - 1;
00,264         Indx = PR + 1;
00,265 /*
00,266         // 'Magnetica' partitioning [
00,267         Jndx = Right;
00,268         PL = Left;
00,269         PR = Left;
00,270         swap (&QWORDS[(Left + Right)>>1], &QWORDS[PR]);
00,271         Pivot = QWORDS[PR];
00,272         for (;PR < Jndx;) {
00,273             if (Pivot > QWORDS[PR + 1]) {
00,274                 swap (&QWORDS[PL], &QWORDS[PR + 1]);
00,275                 PL = PL + 1;
00,276                 PR = PR + 1;
00,277             } else if (Pivot == QWORDS[PR + 1]) {
00,278                 PR = PR + 1;
00,279             } else {
00,280                 for (;Pivot < QWORDS[Jndx];) {
00,281                     Jndx = Jndx - 1;
00,282                 }
00,283                 if (PR + 1 < Jndx) swap (&QWORDS[PR + 1], &QWORDS[Jndx]);
00,284                 Jndx = Jndx - 1;
00,285             }
00,286         }
00,287         Jndx = PL - 1;
00,288         Indx = PR + 1;
00,289         // 'Magnetica' partitioning ]
00,290 */

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,291 /*
00,292 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,293 ; c6-6a+2=94 bytes, 5/2 conditional/unconditional jumps
00,294 ; 'Magnetica' partitioning, mainloop [
00,295 .B4.5::
00,296 0006a 4e 3b 5c e9 08    cmp r11, QWORD PTR [8+rcx+r13*8]
00,297 0006f 77 37             ja .B4.15
00,298 .B4.6::
00,299 00071 75 08             jne .B4.8
00,300 .B4.7::
00,301 00073 49 89 d5          mov r13, rdx
00,302 00076 48 ff c2          inc rdx
00,303 00079 eb 48             jmp .B4.16
00,304 .B4.8::
00,305 0007b 4e 8b 34 c1       mov r14, QWORD PTR [rcx+r8*8]
00,306 0007f 4d 3b de          cmp r11, r14
00,307 00082 73 0c            jae .B4.12
00,308 .B4.10::
00,309 00084 49 ff c8          dec r8
00,310 00087 4e 8b 34 c1       mov r14, QWORD PTR [rcx+r8*8]
00,311 0008b 4d 3b de          cmp r11, r14
00,312 0008e 72 f4            jb .B4.10
00,313 .B4.12::
00,314 00090 4c 3b c2          cmp r8, rdx
00,315 00093 7e 0e            jle .B4.14
00,316 .B4.13::
00,317 00095 4e 8b 7c e9 08    mov r15, QWORD PTR [8+rcx+r13*8]
00,318 0009a 4e 89 74 e9 08    mov QWORD PTR [8+rcx+r13*8], r14
00,319 0009f 4e 89 3c c1       mov QWORD PTR [rcx+r8*8], r15
00,320 .B4.14::
00,321 000a3 49 ff c8          dec r8
00,322 000a6 eb 1b            jmp .B4.16
00,323 .B4.15::
00,324 000a8 4e 8b 74 e9 08    mov r14, QWORD PTR [8+rcx+r13*8]
00,325 000ad 4e 8b 3c e1       mov r15, QWORD PTR [rcx+r12*8]
00,326 000b1 4e 89 34 e1       mov QWORD PTR [rcx+r12*8], r14
00,327 000b5 49 ff c4          inc r12
00,328 000b8 4e 89 7c e9 08    mov QWORD PTR [8+rcx+r13*8], r15
00,329 000bd 49 89 d5          mov r13, rdx
00,330 000c0 48 ff c2          inc rdx
00,331 .B4.16::
00,332 000c3 4d 3b e8          cmp r13, r8
00,333 000c6 7c a2            jl .B4.5
00,334 ; 'Magnetica' partitioning, mainloop ]
00,335 */
00,336 #ifdef FinishWithInsertionSort
00,337     GearBox = ( max50((Right-Indx), (Jndx-Left)) > min50((Right-Indx), (Jndx-Left))<<6 ); // same as MAX/MIN > 64, i.e. Gear=1 if we need Median of 7
00,338 #endif
00,339     if (Indx + InsertionsortTHRESHOLD < Right) { // still refusing to go (always) with the smaller partition first...

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,340         StackPtr = StackPtr + 2;
00,341         Stack[StackPtr - 1] = Indx;
00,342         Stack[StackPtr] = Right;
00,343 #ifdef FinishWithInsertionSort
00,344         // Stack is full, bail out and finalize with Insertionsort [
00,345         StackPtr = StackPtr*(StackPtr + 2 <= StackEntries-1); // StackPtr should be enforced FALSE; also, ensure the next 'push' above is sanctioned - the max index of Stack[] being 'StackEntries-1'
00,346         Right = Right*(StackPtr + 2 <= StackEntries-1); // Left + InsertionsortTHRESHOLD < Right should be enforced FALSE:
00,347         // Stack is full, bail out and finalize with Insertionsort ]
00,348 #endif
00,349     }
00,350     Right = Jndx;
00,351 } //while (Left + InsertionsortTHRESHOLD < Right);
00,352 } while ( (StackPtr != 0) );
00,353 #ifdef FinishWithInsertionSort
00,354 for (Indx=LeftBackup+1; Indx <= RightBackup; Indx++) {
00,355     Jndx = Indx;
00,356     for (;Jndx >= 1;) {
00,357         if (QWORDS[Jndx-1] > QWORDS[Jndx]) { swapUnconditional (&QWORDS[Jndx-1], &QWORDS[Jndx]); swapUnconditional (&QWORDSoFF[Jndx-1], &QWORDSoFF[Jndx]);}
00,358         else break;
00,359         Jndx = Jndx - 1;
00,360     }
00,361 }
00,362 #endif
00,363 }
00,364 // My threads with Magnetica's source and binaries (Linux and Windows):
00,365 // https://www.overclock.net/threads/benchmark-quicksort-says-sorting-2-billion-qwords.1794855/
00,366 // https://www.qb64.org/forum/index.php?topic=3518.msg137645#msg137645
00,367 // https://www.linuxquestions.org/questions/programming-9/qsrt-vs-%27magnetica%27-quicksort-4175703333/#post6299782
00,368 // Quicksort 'Magnetica' r.4, unbreakable (never overflowing the stack) when '#define FinishWithInsertionSort' is uncommented ]
00,369
00,370 // 113 lines of scalar C:
00,371 /*
00,372 00,001 #define StackEntries 99999
00,373 00,002 #define FinishWithInsertionSort
00,374 00,003 #define revision4
00,375 00,004 void swapUnconditional(uint64_t *a, uint64_t *b) { uint64_t t = *a; *a = *b; *b = t; }
00,376 00,005 #define min50(x, y) (y ^ ((x ^ y) & -(x < y)))
00,377 00,006 #define max50(x, y) (x ^ ((x ^ y) & -(x < y)))
00,378 00,007 //
00,379 00,008 //
00,380 00,009 //
00,381 00,010 //
00,382 00,011 //
00,383 00,012 //
00,384 00,013 // Written by Sanmayce, 2021-Dec-02
00,385 00,014 void Quicksort_QB64_v7(uint64_t QWORDS[], int64_t Left, int64_t Right) {
00,386 00,015     #ifdef FinishWithInsertionSort
00,387 00,016     int InsertionsortTHRESHOLD = 17;
00,388 00,017     #else

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,389 00,018   int InsertionsortTHRESHOLD = 0;
00,390 00,019   #endif
00,391 00,020   int64_t Indx, Jndx, PL, PR;
00,392 00,021   int64_t Stack[StackEntries];
00,393 00,022   int64_t StackPtr = 0;
00,394 00,023   uint64_t Pivot;
00,395 00,024   int64_t LeftBackup = Left;
00,396 00,025   int64_t RightBackup = Right;
00,397 00,026   register uint64_t x0,x1,x2,x3,x4,x5,x6;
00,398 00,027   int o0,o1,o2,o3,o4,o5,o6;
00,399 00,028   int64_t TheMiddleOfMiddle;
00,400 00,029   int GearBox = 0; // 0 is Median of 3; 1 is Median of 7; Shifting to higher gear if BiggerPartition:SmallerPartition ratio is > 64:1
00,401 00,030   StackPtr++; Stack[StackPtr] = Left;
00,402 00,031   StackPtr++; Stack[StackPtr] = Right;
00,403 00,032   do {
00,404 00,033       Right = Stack[StackPtr];
00,405 00,034       Left = Stack[StackPtr - 1];
00,406 00,035       StackPtr = StackPtr - 2;
00,407 00,036       for(;(Left + InsertionsortTHRESHOLD < Right);) { // For instance, 1 + 17 < 19 i.e. (19-1)/4+6 < 19
00,408 00,037           Jndx = Right;
00,409 00,038           PL = Left;
00,410 00,039           PR = Left;
00,411 00,040   #ifdef FinishWithInsertionSort
00,412 00,041       TheMiddleOfMiddle = Left + ((Right-Left)>>2); // (4Left + Right - Left)/4; Caution: [TheMiddleOfMiddle+6] should be <= [Right] i.e. 6*(4Left + Right - Left)/4 <= Right or 4*6+(4Left + Right - Left) <=
4Right or 4*6 <= 3Right-3Left or (4*6)/3 <= Right-Left i.e. 8 <= Right-Left as a minimum condition to enter the 'for'.
00,413 00,042       x0 = QWORDS[TheMiddleOfMiddle+0];
00,414 00,043       x1 = QWORDS[TheMiddleOfMiddle+1];
00,415 00,044       x2 = QWORDS[TheMiddleOfMiddle+2];
00,416 00,045   if (GearBox == 0) { // Median of 5 proves more effective than 7, however 7 betters better the nastiest N^2
00,417 00,046       o0 = (x0>x1)+(x0>x2);
00,418 00,047       o1 = (x1>x0)+(x1>x2);
00,419 00,048       o2 = (0+1+2)-(o0+o1);
00,420 00,049       QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2;
00,421 00,050       swapUnconditional (&QWORDS[TheMiddleOfMiddle+1], &QWORDS[PR]); //2nd
00,422 00,051   } else {
00,423 00,052       x3 = QWORDS[TheMiddleOfMiddle+3];
00,424 00,053       x4 = QWORDS[TheMiddleOfMiddle+4];
00,425 00,054       x5 = QWORDS[TheMiddleOfMiddle+5];
00,426 00,055       x6 = QWORDS[TheMiddleOfMiddle+6];
00,427 00,056       o0 = (x0>x1)+(x0>x2)+(x0>x3)+(x0>x4)+(x0>x5)+(x0>x6);
00,428 00,057       o1 = (x1>x0)+(x1>x2)+(x1>x3)+(x1>x4)+(x1>x5)+(x1>x6);
00,429 00,058       o2 = (x2>x0)+(x2>x1)+(x2>x3)+(x2>x4)+(x2>x5)+(x2>x6);
00,430 00,059       o3 = (x3>x0)+(x3>x1)+(x3>x2)+(x3>x4)+(x3>x5)+(x3>x6);
00,431 00,060       o4 = (x4>x0)+(x4>x1)+(x4>x2)+(x4>x3)+(x4>x5)+(x4>x6);
00,432 00,061       o5 = (x5>x0)+(x5>x1)+(x5>x2)+(x5>x3)+(x5>x4)+(x5>x6);
00,433 00,062       o6 = (0+1+2+3+4+5+6)-(o0+o1+o2+o3+o4+o5);
00,434 00,063       QWORDS[TheMiddleOfMiddle+o0]=x0; QWORDS[TheMiddleOfMiddle+o1]=x1; QWORDS[TheMiddleOfMiddle+o2]=x2; QWORDS[TheMiddleOfMiddle+o3]=x3; QWORDS[TheMiddleOfMiddle+o4]=x4; QWORDS[TheMiddleOfMiddle+o5]=x5;
QWORDS[TheMiddleOfMiddle+o6]=x6;
00,435 00,064       swapUnconditional (&QWORDS[TheMiddleOfMiddle+3], &QWORDS[PR]); //4th

```

```

00,436 00,065 }
00,437 00,066     Pivot = QWORDS[PR];
00,438 00,067 #else
00,439 00,068     swapUnconditional (&QWORDS[(Left + Right)>>1], &QWORDS[PR]);
00,440 00,069     Pivot = QWORDS[PR];
00,441 00,070 #endif
00,442 00,071 #ifndef revision4
00,443 00,072     for (;PR < Jndx;) {
00,444 00,073         PR = PR + 1;
00,445 00,074         if (Pivot > QWORDS[PR]) {
00,446 00,075             swapUnconditional (&QWORDS[PL], &QWORDS[PR]);
00,447 00,076             PL = PL + 1;
00,448 00,077         } else if (Pivot < QWORDS[PR]) {
00,449 00,078             for (;Pivot < QWORDS[Jndx];) {
00,450 00,079                 Jndx = Jndx - 1;
00,451 00,080             }
00,452 00,081             if (PR < Jndx) swapUnconditional (&QWORDS[PR], &QWORDS[Jndx]);
00,453 00,082             Jndx = Jndx - 1;
00,454 00,083             PR = PR - 1;
00,455 00,084         }
00,456 00,085     }
00,457 00,086 #endif
00,458 00,087     Jndx = PL - 1;
00,459 00,088     Indx = PR + 1;
00,460 00,089 #ifndef FinishWithInsertionSort
00,461 00,090     GearBox = ( max50((Right-Indx), (Jndx-Left)) > min50((Right-Indx), (Jndx-Left))<<6 ); // same as MAX/MIN > 64, i.e. Gear=1 if we need Median of 7
00,462 00,091 #endif
00,463 00,092     if (Indx + InsertionsortTHRESHOLD < Right) { // still refusing to go (always) with the smaller partition first...
00,464 00,093         StackPtr = StackPtr + 2;
00,465 00,094         Stack[StackPtr - 1] = Indx;
00,466 00,095         Stack[StackPtr] = Right;
00,467 00,096 #ifndef FinishWithInsertionSort
00,468 00,097         StackPtr = StackPtr*(StackPtr + 2 <= StackEntries-1); // StackPtr should be enforced FALSE; also, ensure the next 'push' above is sanctioned - the max index of Stack[] being 'StackEntries-1'
00,469 00,098         Right = Right*(StackPtr + 2 <= StackEntries-1); // Left + InsertionsortTHRESHOLD < Right should be enforced FALSE:
00,470 00,099 #endif
00,471 00,100     }
00,472 00,101     Right = Jndx;
00,473 00,102 } //while (Left + InsertionsortTHRESHOLD < Right);
00,474 00,103 } while ( (StackPtr != 0) );
00,475 00,104 #ifndef FinishWithInsertionSort
00,476 00,105 for (Indx=LeftBackup+1; Indx <= RightBackup; Indx++) {
00,477 00,106     Jndx = Indx;
00,478 00,107     for (;Jndx >= 1;) {
00,479 00,108         if (QWORDS[Jndx-1] > QWORDS[Jndx]) swapUnconditional (&QWORDS[Jndx-1], &QWORDS[Jndx]); else break;
00,480 00,109         Jndx = Jndx - 1;
00,481 00,110     }
00,482 00,111 }
00,483 00,112 #endif
00,484 00,113 }

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,485 */
00,486
00,487 // 380 lines scalar Assembly:
00,488 /*
00,489 00,001 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
00,490 00,002 ; mark_description "-S -Os";
00,491 00,003
00,492 00,004 Quicksort_QB64_v7PROC
00,493 00,005 .B4.1::
00,494 00,006     push    rbx
00,495 00,007     push    rsi
00,496 00,008     push    rdi
00,497 00,009     push    r12
00,498 00,010     push    r13
00,499 00,011     push    r14
00,500 00,012     push    r15
00,501 00,013     push    rbp
00,502 00,014     mov     eax, 800152
00,503 00,015     call   __chkstk
00,504 00,016     sub     rsp, 800152
00,505 00,017     mov     r13, rcx
00,506 00,018     mov     QWORD PTR [800064+rsp], r8
00,507 00,019     xor     r14d, r14d
00,508 00,020     mov     QWORD PTR [40+rsp], rdx
00,509 00,021     mov     QWORD PTR [56+rsp], rdx
00,510 00,022     mov     QWORD PTR [32+rsp], 2
00,511 00,023     mov     QWORD PTR [64+rsp], r8
00,512 00,024 .B4.2::
00,513 00,025     mov     rax, QWORD PTR [32+rsp]
00,514 00,026     mov     rdx, QWORD PTR [40+rsp+rax*8]
00,515 00,027     mov     r12, QWORD PTR [48+rsp+rax*8]
00,516 00,028     add     rax, -2
00,517 00,029     mov     QWORD PTR [800040+rsp], rdx
00,518 00,030     mov     QWORD PTR [32+rsp], rax
00,519 00,031     lea     rcx, QWORD PTR [17+rdx]
00,520 00,032     mov     QWORD PTR [800072+rsp], rcx
00,521 00,033     cmp     r12, rcx
00,522 00,034     jle     .B4.27
00,523 00,035 .B4.3::
00,524 00,036     mov     QWORD PTR [800056+rsp], rdx
00,525 00,037     lea     rax, QWORD PTR [r13+rdx*8]
00,526 00,038     mov     QWORD PTR [800048+rsp], rax
00,527 00,039 .B4.4::
00,528 00,040     mov     r8, r12
00,529 00,041     xor     eax, eax
00,530 00,042     mov     rbp, QWORD PTR [800040+rsp]
00,531 00,043     sub     r8, rbp
00,532 00,044     sar     r8, 2
00,533 00,045     xor     edx, edx

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,534 00,046      add     r8, rbp
00,535 00,047      xor     r15d, r15d
00,536 00,048      mov     rbx, QWORD PTR [800056+rsp]
00,537 00,049      mov     rdi, r12
00,538 00,050      mov     QWORD PTR [800056+rsp], rbp
00,539 00,051      lea     rsi, QWORD PTR [rbp*8]
00,540 00,052      lea     rcx, QWORD PTR [r13+r8*8]
00,541 00,053      mov     r10, QWORD PTR [rcx]
00,542 00,054      mov     r11, QWORD PTR [8+rcx]
00,543 00,055      cmp     r10, r11
00,544 00,056      mov     r9, QWORD PTR [16+rcx]
00,545 00,057      seta    al
00,546 00,058      cmp     r10, r9
00,547 00,059      seta    dl
00,548 00,060      add     eax, edx
00,549 00,061      xor     edx, edx
00,550 00,062      cmp     r11, r10
00,551 00,063      setae   dl
00,552 00,064      cmp     r11, r9
00,553 00,065      seta    r15b
00,554 00,066      add     edx, r15d
00,555 00,067      test    r14d, r14d
00,556 00,068      jne     .B4.6
00,557 00,069      .B4.5::
00,558 00,070      add     rcx, 8
00,559 00,071      lea     r14, QWORD PTR [r8+rax]
00,560 00,072      mov     QWORD PTR [r13+r14*8], r10
00,561 00,073      lea     r10, QWORD PTR [r8+rdx]
00,562 00,074      sub     r8, rax
00,563 00,075      sub     r8, rdx
00,564 00,076      mov     QWORD PTR [r13+r10*8], r11
00,565 00,077      mov     QWORD PTR [24+r13+r8*8], r9
00,566 00,078      jmp     .B4.7
00,567 00,079      .B4.6::
00,568 00,080      mov     QWORD PTR [800104+rsp], rsi
00,569 00,081      xor     r14d, r14d
00,570 00,082      mov     rsi, QWORD PTR [24+rcx]
00,571 00,083      cmp     r10, rsi
00,572 00,084      mov     QWORD PTR [800096+rsp], rdi
00,573 00,085      mov     rdi, QWORD PTR [32+rcx]
00,574 00,086      seta    r14b
00,575 00,087      xor     r15d, r15d
00,576 00,088      cmp     r10, rdi
00,577 00,089      mov     QWORD PTR [800088+rsp], r12
00,578 00,090      push    0
00,579 00,091      pop     r12
00,580 00,092      seta    r12b
00,581 00,093      mov     QWORD PTR [800112+rsp], rbp
00,582 00,094      add     r14d, r12d

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,583 00,095      mov     rbp, QWORD PTR [40+rcx]
00,584 00,096      xor     r12d, r12d
00,585 00,097      cmp     r10, rbp
00,586 00,098      mov     QWORD PTR [800120+rsp], rbx
00,587 00,099      seta    r12b
00,588 00,100      mov     rbx, QWORD PTR [48+rcx]
00,589 00,101      cmp     r10, rbx
00,590 00,102      mov     QWORD PTR [800080+rsp], rcx
00,591 00,103      push   0
00,592 00,104      pop     rcx
00,593 00,105      seta    cl
00,594 00,106      add     eax, r14d
00,595 00,107      add     r12d, ecx
00,596 00,108      xor     ecx, ecx
00,597 00,109      cmp     r11, rsi
00,598 00,110      seta    cl
00,599 00,111      xor     r14d, r14d
00,600 00,112      cmp     r11, rdi
00,601 00,113      seta    r15b
00,602 00,114      cmp     r11, rbp
00,603 00,115      seta    r14b
00,604 00,116      add     eax, r12d
00,605 00,117      xor     r12d, r12d
00,606 00,118      cmp     r11, rbx
00,607 00,119      seta    r12b
00,608 00,120      add     ecx, r15d
00,609 00,121      add     edx, ecx
00,610 00,122      xor     ecx, ecx
00,611 00,123      cmp     r9, r10
00,612 00,124      setae   cl
00,613 00,125      xor     r15d, r15d
00,614 00,126      add     r14d, r12d
00,615 00,127      xor     r12d, r12d
00,616 00,128      cmp     r9, r11
00,617 00,129      setae   r12b
00,618 00,130      add     edx, r14d
00,619 00,131      add     ecx, r12d
00,620 00,132      xor     r12d, r12d
00,621 00,133      cmp     r9, rsi
00,622 00,134      seta    r12b
00,623 00,135      xor     r14d, r14d
00,624 00,136      cmp     r9, rdi
00,625 00,137      seta    r15b
00,626 00,138      add     r12d, r15d
00,627 00,139      xor     r15d, r15d
00,628 00,140      add     ecx, r12d
00,629 00,141      xor     r12d, r12d
00,630 00,142      cmp     r9, rbp
00,631 00,143      seta    r12b

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,632 00,144    cmp     r9, rbx
00,633 00,145    seta     r14b
00,634 00,146    cmp     rsi, r10
00,635 00,147    setae    r15b
00,636 00,148    add     r12d, r14d
00,637 00,149    xor     r14d, r14d
00,638 00,150    cmp     rsi, r11
00,639 00,151    setae    r14b
00,640 00,152    add     ecx, r12d
00,641 00,153    add     r15d, r14d
00,642 00,154    xor     r14d, r14d
00,643 00,155    cmp     rsi, r9
00,644 00,156    setae    r14b
00,645 00,157    xor     r12d, r12d
00,646 00,158    cmp     rsi, rdi
00,647 00,159    seta     r12b
00,648 00,160    add     r14d, r12d
00,649 00,161    xor     r12d, r12d
00,650 00,162    add     r15d, r14d
00,651 00,163    xor     r14d, r14d
00,652 00,164    cmp     rsi, rbp
00,653 00,165    seta     r14b
00,654 00,166    cmp     rsi, rbx
00,655 00,167    seta     r12b
00,656 00,168    add     r14d, r12d
00,657 00,169    xor     r12d, r12d
00,658 00,170    add     r15d, r14d
00,659 00,171    xor     r14d, r14d
00,660 00,172    cmp     rdi, r10
00,661 00,173    setae    r14b
00,662 00,174    cmp     rdi, r11
00,663 00,175    setae    r12b
00,664 00,176    add     r14d, r12d
00,665 00,177    xor     r12d, r12d
00,666 00,178    cmp     rdi, r9
00,667 00,179    setae    r12b
00,668 00,180    cmp     rdi, rsi
00,669 00,181    mov     QWORD PTR [800128+rsp], r15
00,670 00,182    push    0
00,671 00,183    pop     r15
00,672 00,184    setae    r15b
00,673 00,185    add     r12d, r15d
00,674 00,186    xor     r15d, r15d
00,675 00,187    add     r14d, r12d
00,676 00,188    xor     r12d, r12d
00,677 00,189    cmp     rdi, rbp
00,678 00,190    seta     r12b
00,679 00,191    cmp     rdi, rbx
00,680 00,192    seta     r15b

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,681 00,193      add     r12d, r15d
00,682 00,194      xor     r15d, r15d
00,683 00,195      add     r14d, r12d
00,684 00,196      xor     r12d, r12d
00,685 00,197      cmp     rbp, r10
00,686 00,198      setae   r12b
00,687 00,199      cmp     rbp, r11
00,688 00,200      setae   r15b
00,689 00,201      add     r12d, r15d
00,690 00,202      xor     r15d, r15d
00,691 00,203      cmp     rbp, r9
00,692 00,204      setae   r15b
00,693 00,205      cmp     rbp, rsi
00,694 00,206      mov     QWORD PTR [800136+rsp], r14
00,695 00,207      push    0
00,696 00,208      pop     r14
00,697 00,209      setae   r14b
00,698 00,210      add     r15d, r14d
00,699 00,211      xor     r14d, r14d
00,700 00,212      add     r12d, r15d
00,701 00,213      xor     r15d, r15d
00,702 00,214      cmp     rbp, rdi
00,703 00,215      setae   r15b
00,704 00,216      cmp     rbp, rbx
00,705 00,217      seta    r14b
00,706 00,218      add     r15d, r14d
00,707 00,219      lea     r14, QWORD PTR [r8+rax]
00,708 00,220      mov     QWORD PTR [r13+r14*8], r10
00,709 00,221      lea     r10, QWORD PTR [r8+rdx]
00,710 00,222      mov     QWORD PTR [r13+r10*8], r11
00,711 00,223      lea     r11, QWORD PTR [r8+rcx]
00,712 00,224      mov     r10, QWORD PTR [800128+rsp]
00,713 00,225      add     r12d, r15d
00,714 00,226      mov     QWORD PTR [r13+r11*8], r9
00,715 00,227      add     rcx, r10
00,716 00,228      mov     r11, QWORD PTR [800136+rsp]
00,717 00,229      lea     r9, QWORD PTR [r8+r10]
00,718 00,230      mov     QWORD PTR [r13+r9*8], rsi
00,719 00,231      lea     rsi, QWORD PTR [r8+r11]
00,720 00,232      add     r11, r12
00,721 00,233      mov     QWORD PTR [r13+rsi*8], rdi
00,722 00,234      add     rcx, r11
00,723 00,235      mov     rsi, QWORD PTR [800104+rsp]
00,724 00,236      lea     rdi, QWORD PTR [r8+r12]
00,725 00,237      sub     r8, rax
00,726 00,238      sub     r8, rdx
00,727 00,239      sub     r8, rcx
00,728 00,240      mov     QWORD PTR [r13+rdi*8], rbp
00,729 00,241      mov     rcx, QWORD PTR [800080+rsp]

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,730 00,242      mov     rbp, QWORD PTR [800112+rsp]
00,731 00,243      add     rcx, 24
00,732 00,244      mov     QWORD PTR [168+r13+r8*8], rbx
00,733 00,245      mov     rbx, QWORD PTR [800120+rsp]
00,734 00,246      mov     rdi, QWORD PTR [800096+rsp]
00,735 00,247      mov     r12, QWORD PTR [800088+rsp]
00,736 00,248 .B4.7::
00,737 00,249      mov     rdx, QWORD PTR [800048+rsp]
00,738 00,250      call    swapUnconditional
00,739 00,251 .B4.8::
00,740 00,252      mov     rax, QWORD PTR [800048+rsp]
00,741 00,253      cmp     r12, QWORD PTR [800040+rsp]
00,742 00,254      mov     r14, QWORD PTR [rax]
00,743 00,255      jle     .B4.23
00,744 00,256 .B4.10::
00,745 00,257      inc     rbp
00,746 00,258      lea     rdx, QWORD PTR [r13+rbp*8]
00,747 00,259      cmp     r14, QWORD PTR [rdx]
00,748 00,260      jbe     .B4.13
00,749 00,261 .B4.11::
00,750 00,262      lea     rcx, QWORD PTR [r13+rsi]
00,751 00,263      call    swapUnconditional
00,752 00,264 .B4.12::
00,753 00,265      add     rsi, 8
00,754 00,266      inc     rbx
00,755 00,267      jmp     .B4.21
00,756 00,268 .B4.13::
00,757 00,269      jae     .B4.21
00,758 00,270 .B4.14::
00,759 00,271      lea     rax, QWORD PTR [r13+rdi*8]
00,760 00,272      cmp     r14, QWORD PTR [rax]
00,761 00,273      jae     .B4.18
00,762 00,274 .B4.16::
00,763 00,275      dec     rdi
00,764 00,276      lea     rax, QWORD PTR [r13+rdi*8]
00,765 00,277      cmp     r14, QWORD PTR [rax]
00,766 00,278      jb     .B4.16
00,767 00,279 .B4.18::
00,768 00,280      cmp     rbp, rdi
00,769 00,281      jge     .B4.20
00,770 00,282 .B4.19::
00,771 00,283      mov     rcx, rdx
00,772 00,284      mov     rdx, rax
00,773 00,285      call    swapUnconditional
00,774 00,286 .B4.20::
00,775 00,287      dec     rdi
00,776 00,288      dec     rbp
00,777 00,289 .B4.21::
00,778 00,290      cmp     rbp, rdi

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,779 00,291      jl      .B4.10
00,780 00,292 .B4.23::
00,781 00,293      dec     rbx
00,782 00,294      lea     r9, QWORD PTR [1+rbp]
00,783 00,295      mov     rdi, r12
00,784 00,296      mov     r8, rbx
00,785 00,297      sub     r8, QWORD PTR [800040+rsp]
00,786 00,298      sub     rdi, r9
00,787 00,299      xor     edx, edx
00,788 00,300      cmp     rdi, r8
00,789 00,301      mov     rcx, rdi
00,790 00,302      setl    dl
00,791 00,303      xor     r14d, r14d
00,792 00,304      xor     rcx, r8
00,793 00,305      neg     edx
00,794 00,306      add     rbp, 18
00,795 00,307      movsxd  rdx, edx
00,796 00,308      and     rcx, rdx
00,797 00,309      xor     r8, rcx
00,798 00,310      xor     rdi, rcx
00,799 00,311      shl     r8, 6
00,800 00,312      cmp     rdi, r8
00,801 00,313      setg    r14b
00,802 00,314      cmp     r12, rbp
00,803 00,315      jle     .B4.25
00,804 00,316 .B4.24::
00,805 00,317      mov     rax, QWORD PTR [32+rsp]
00,806 00,318      cmp     rax, 99994
00,807 00,319      mov     QWORD PTR [64+rsp+rax*8], r12
00,808 00,320      push    0
00,809 00,321      pop     r12
00,810 00,322      setle   r12b
00,811 00,323      mov     QWORD PTR [56+rsp+rax*8], r9
00,812 00,324      add     rax, 2
00,813 00,325      imul    rax, r12
00,814 00,326      mov     QWORD PTR [32+rsp], rax
00,815 00,327 .B4.25::
00,816 00,328      mov     r12, rbx
00,817 00,329      cmp     rbx, QWORD PTR [800072+rsp]
00,818 00,330      jg      .B4.4
00,819 00,331 .B4.27::
00,820 00,332      cmp     QWORD PTR [32+rsp], 0
00,821 00,333      jne     .B4.2
00,822 00,334 .B4.28::
00,823 00,335      mov     rax, QWORD PTR [40+rsp]
00,824 00,336      inc     rax
00,825 00,337      mov     QWORD PTR [40+rsp], rax
00,826 00,338      cmp     rax, QWORD PTR [800064+rsp]
00,827 00,339      jg      .B4.39

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,828 00,340 .B4.30::
00,829 00,341      mov     rbx, QWORD PTR [40+rsp]
00,830 00,342      test    rbx, rbx
00,831 00,343      jle     .B4.28
00,832 00,344 .B4.32::
00,833 00,345      mov     rax, QWORD PTR [-8+r13+rbx*8]
00,834 00,346      cmp     rax, QWORD PTR [r13+rbx*8]
00,835 00,347      jbe     .B4.28
00,836 00,348 .B4.33::
00,837 00,349      lea     rdx, QWORD PTR [r13+rbx*8]
00,838 00,350      lea     rcx, QWORD PTR [-8+rdx]
00,839 00,351      call    swapUnconditional
00,840 00,352 .B4.34::
00,841 00,353      dec     rbx
00,842 00,354      test    rbx, rbx
00,843 00,355      jg      .B4.32
00,844 00,356      jmp     .B4.28
00,845 00,357 .B4.39::
00,846 00,358      add     rsp, 800152
00,847 00,359      pop     rbp
00,848 00,360      pop     r15
00,849 00,361      pop     r14
00,850 00,362      pop     r13
00,851 00,363      pop     r12
00,852 00,364      pop     rdi
00,853 00,365      pop     rsi
00,854 00,366      pop     rbx
00,855 00,367      ret
00,856 00,368 .B4.40::
00,857 00,369 Quicksort_QB64_v7 ENDP
00,858 00,370
00,859 00,371 swapUnconditionalPROC
00,860 00,372 .B5.1::
00,861 00,373      mov     rax, QWORD PTR [rdx]
00,862 00,374      mov     r8, QWORD PTR [rcx]
00,863 00,375      mov     QWORD PTR [rcx], rax
00,864 00,376      mov     QWORD PTR [rdx], r8
00,865 00,377 .B5.4::
00,866 00,378      ret
00,867 00,379 .B5.2::
00,868 00,380 swapUnconditional ENDP
00,869 */

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

# **memKAZE.h:**

```
00,001 #include <stdint.h> // Needed for uint32_t
00,002 #include <string.h> // Needed for memcmp, memcpy
00,003
00,004 //Function memcmpKAZE% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,005 int memcmpKAZE (char * buf1, char * buf2, uint32_t bytes)
00,006 {
00,007     return memcmp(buf1, buf2, bytes);
00,008 }
00,009
00,010 //int __cdecl memcmp (const void * buf1, const void * buf2, size_t count)
00,011 //int memcmp(buf1, buf2, count) - compare memory for lexical order
00,012 //
00,013 //Purpose:
00,014 //     Compares count bytes of memory starting at buf1 and buf2
00,015 //     and find if equal or which one is first in lexical order.
00,016 //
00,017 //Entry:
00,018 //     void *buf1, *buf2 - pointers to memory sections to compare
00,019 //     size_t count - length of sections to compare
00,020 //
00,021 //Exit:
00,022 //     returns < 0 if buf1 < buf2
00,023 //     returns 0 if buf1 == buf2
00,024 //     returns > 0 if buf1 > buf2
00,025
00,026 //Sub memcpyKAZE (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,027 void memcpyKAZE (char * dest, char * source, uint32_t bytes)
00,028 {
00,029     memcpy(dest, source, bytes);
00,030 }
00,031 //void * __cdecl memcpy (void * dst, const void * src, size_t count)
00,032 //memcpy - Copy source buffer to destination buffer
00,033 //
00,034 //Purpose:
00,035 //     memcpy() copies a source memory buffer to a destination memory buffer.
00,036 //     This routine does NOT recognize overlapping buffers, and thus can lead
00,037 //     to propogation.
00,038 //
00,039 //     For cases where propogation must be avoided, memmove() must be used.
00,040 //
00,041 //Entry:
00,042 //     void *dst = pointer to destination buffer
00,043 //     const void *src = pointer to source buffer
00,044 //     size_t count = number of bytes to copy
00,045 //
00,046 //Exit:
00,047 //     Returns a pointer to the destination buffer
```

**manatarka.h:**

```

00,001 // Last change: 2021-Dec-25
00,002
00,003 // manatarka.h:
00,004 // In order to compile properly, edit the next line from E:\qb64\internal\temp\recompile_win.bat:
00,005 // c_compiler\bin\g++ -s -Wfatal-errors ...
00,006 // Insert "-O3 -msse4.1 -maes", or even better "-O3 -mavx2 -maes" for YMM variants, so:
00,007 // c_compiler\bin\g++ -O3 -mavx2 -maes -s -Wfatal-errors ...
00,008
00,009 /*
00,010 Declare CustomType Library
00,011     Sub memcpy (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,012 End Declare
00,013
00,014 a$ = "1234567890"
00,015 b$ = "ABCDEFGHJIJ"
00,016
00,017 memcpy _Offset(a$) + 5, _Offset(b$) + 5, 5
00,018 Print a$
00,019 */
00,020
00,021 /*
00,022 Rem In folder 'E:\qb64\internal\temp' in 'recompile_win.bat' file had to add '-O2 -msse4.1' into compile line in order to get XMM going:
00,023
00,024 Rem @echo off
00,025 Rem cd %0\..\
00,026 Rem echo Recompiling...
00,027 Rem cd ../c
00,028 Rem c_compiler\bin\g++ -O2 -msse4.1 -s -Wfatal-errors -w -Wall qbx.cpp libqb\os\win\libqb_2.0.1_000000000000.o -D DEPENDENCY_NO_SOCKETS -D DEPENDENCY_NO_PRINTER -D DEPENDENCY_NO_ICON -D DEPENDENCY_NO_SCREENIMAGE
parts\core\os\win\src.a -lopengl32 -lglu32 -mwindows -static-libgcc -static-libstdc++ -D GLEW_STATIC -D FREEGLUT_STATIC -lwinmm -o "..\..\UCASE_showdown.exe"
00,029 Rem pause
00,030 */
00,031
00,032 /*
00,033 Declare CustomType Library "manatarka"
00,034     Sub UCASE_XMM (ByVal QWORDSrc As _Offset, ByVal QWORDdst As _Offset, ByVal bytesto As Integer64)
00,035     Sub UCASE_XMM_inplace (ByVal QWORDSrc As _Offset, ByVal bytesto As Integer64)
00,036     Function Railgun_Doublet%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer arithmetic) is
to define HaystackLEN as _Offset instead of LONG
00,037     Function Railgun_Trolldom_64%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer arithmetic)
is to define HaystackLEN as _Offset instead of Integer64
00,038     Function Railgun_Nyotengu_XMM_YMM_ZMM%& (ByVal HaystackADDR As _Offset, ByVal NeedleADDR As _Offset, ByVal HaystackLEN As _Offset, ByVal NeedleLEN As Long) ' the workaround in QB64 for lacking casting (in pointer
arithmetic) is to define HaystackLEN as _Offset instead of Integer64
00,039     'char * Railgun_Doublet (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern)
00,040     'char * Railgun_Trolldom_64 (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern)
00,041     'char * Railgun_Nyotengu_XMM_YMM_ZMM (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern) // Last change: 2020-Nov-30
00,042     Sub DoubleDeuceAES_Gumbottron (ByVal buffer As _Offset, ByVal bytesto As Integer64, ByVal LoPart As _Offset, ByVal HiPart As _Offset) 'Note: This hasher uses AES extension, the CPU has to support it!
00,043     Sub DoubleDeuceAES_Gumbottron_YMM (ByVal buffer As _Offset, ByVal bytesto As Integer64, ByVal LoPart As _Offset, ByVal HiPart As _Offset) 'Note: This hasher uses AES extension, the CPU has to support it!
00,044     'void DoubleDeuceAES_Gumbottron(const uint8_t *buffer, size_t length, uint64_t *LoPart, uint64_t *HiPart) {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,045
00,046 Sub memcpyKAZE (ByVal dest As _Offset, ByVal source As _Offset, ByVal bytes As Long)
00,047 Function memcpyKAZE% (ByVal buf1 As _Offset, ByVal buf2 As _Offset, ByVal bytes As Long)
00,048 End Declare
00,049 */
00,050 //UCASE_XMM MhandleSRC.OFFSET, MhandleDST.OFFSET, bytesto
00,051
00,052 /*
00,053 #include <emmintrin.h> // SSE2 intrinsics
00,054 #include <smintrin.h> // SSE4.1 intrinsics
00,055 #include <immintrin.h> // AVX intrinsics
00,056 #include <zmmmintrin.h> // AVX2 intrinsics, definitions and declarations for use with 512-bit compiler intrinsics.
00,057 */
00,058 #include <immintrin.h>
00,059 #include <stdint.h> // Needed for uint32_t
00,060
00,061 void UCASE_XMM(uint64_t QWORD5src[], uint64_t QWORD5dst[], int64_t bytesto) {
00,062
00,063     int64_t i = 0;
00,064     __m128i maska;
00,065     __m128i maskz;
00,066     __m128i mask32;
00,067     __m128i maskaz;
00,068     __m128i r0;
00,069     char *QWORD5s = (char *)QWORD5src;
00,070     char *QWORD5d = (char *)QWORD5dst;
00,071
00,072     maska = _mm_set1_epi8( 'a' );
00,073     maskz = _mm_set1_epi8( 'z' );
00,074     mask32 = _mm_set1_epi8( 32 );
00,075
00,076     for( ; i < bytesto/16 * 16 ; i+=16 )
00,077     {
00,078         r0 = _mm_loadu_si128( ( __m128i * )&QWORD5s[ i ] );
00,079
00,080         // maskaz contains 0x00 where character between 'a' and 'z', 0xff otherwise.
00,081         maskaz = _mm_or_si128( _mm_cmplt_epi8( r0, maska ), _mm_cmpgt_epi8( r0, maskz ) );
00,082
00,083         // Set the 6th bit to 0 only for lowercase characters.
00,084         r0 = _mm_andnot_si128( _mm_andnot_si128( maskaz, mask32 ), r0 );
00,085         _mm_storeu_si128( ( __m128i * )&QWORD5d[ i ], r0 );
00,086     }
00,087 }
00,088
00,089 void UCASE_XMM_inplace(uint64_t QWORD5src[], int64_t bytesto) {
00,090
00,091     int64_t i = 0;
00,092     __m128i maska;
00,093     __m128i maskz;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,094   __m128i mask32;
00,095   __m128i maskz;
00,096   __m128i r0;
00,097   char *QWORDSs = (char *)QWORDSsrc;
00,098
00,099   maska = _mm_set1_epi8( 'a' );
00,100   maskz = _mm_set1_epi8( 'z' );
00,101   mask32 = _mm_set1_epi8( 32 );
00,102
00,103   for( ; i < bytesto/16 * 16 ; i+=16 )
00,104   {
00,105       r0 = _mm_loadu_si128( ( __m128i *) &QWORDSs[ i ] );
00,106
00,107       // maskz contains 0x00 where character between 'a' and 'z', 0xff otherwise.
00,108       maskz = _mm_or_si128( _mm_cmplt_epi8( r0, maska ), _mm_cmpgt_epi8( r0, maskz ) );
00,109
00,110       // Set the 6th bit to 0 only for lowercase characters.
00,111       r0 = _mm_andnot_si128( _mm_andnot_si128( maskz, mask32 ), r0 );
00,112       _mm_storeu_si128( ( __m128i *) &QWORDSs[ i ], r0 );
00,113   }
00,114 }
00,115
00,116 // Railgun_Trolldom (the successor of Railgun_Swampshine_BailOut - avoiding second pattern comparison in BMH2 and pseudo-BMH4), copyleft 2016-Aug-19, Kaze.
00,117 // Railgun_Swampshine_BailOut, copyleft 2016-Aug-10, Kaze.
00,118 // Internet "home" page: http://www.codeproject.com/Articles/250566/Fastest-strstr-like-function-in-C
00,119 // My homepage (homeserver, often down): http://www.sanmayce.com/Railgun/
00,120 /*
00,121 !!!!!!!!!!!!!!!!!!!!!!! BENCHMARKING GNU's memmem vs Railgun !!!!!!!!!!!!!!!!!!!!!!! [
00,122 Add-on: 2016-Aug-22
00,123
00,124 Two things.
00,125
00,126 First, the fix from the last time was buggy, my apologies, now fixed, quite embarrassing since it is a simple left/right boundary check. It doesn't affect the speed, it appears as rare pattern hit misses.
00,127 Since I don't believe in saying "sorry" but in making things right, here my attempt to further disgrace my amateurish work follows:
00,128 Two years ago, I didn't pay due attention to adding 'Swampwalker' heuristic to the Railgun_Ennearch, I mean, only quick test was done and no real proofing - this was due not to a blunder of mine, nor carelessness, but
overconfidence in my ability to write "on the fly". Stupid, indeed, however, when a coder gets momentum in writing simple etudes he starts gaining false confidence of mastering the subject, not good for sure!
00,129 Hopefully, other coders will learn to avoid such full of neglect style.
00,130
00,131 Second, wanted to present the heaviest testbed for search i.e. memmem() functions: it benefits the benchmarking (speed in real application) as well as bug-control.
00,132
00,133 The benchmark is downloadable at my INTERNET drive:
00,134 https://1drv.ms/u/s!AmWWFXGmZDmEglwj1UtrMjrfhosK
00,135
00,136 The speed showdown has three facets:
00,137 - compares the 64bit code generated from GCC 5.10 versus Intel 15.0 compilers;
00,138 - compares four types of datasets - search speed through English texts versus genome ACGT-type data versus binary versus UTF8;
00,139 - compares the tweaked Two-Way algorithm (implemented by Eric Blake) and adopted by GLIBC as memmem() versus my Railgun_Swampshine.
00,140
00,141 Note1: The GLIBC memmem() was taken from latest (2016-08-05) glibc 2.24 tar:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

00,142 <https://www.gnu.org/software/libc/>

00,143 Note2: Eric Blake says that he enhanced the linearity of Two-Way by adding some sublinear paths, well, Railgun is all about sublinearity, so feel free to experiment with your own testfiles (worst-case-scenarios), just make such a file feed the compressor with it, then we will see how the LINEAR Two-Way behaves versus Railgun\_Swampshine.

00,144 Note3: Just copy-and-paste 'Railgun\_Swampshine' or 'Railgun\_Ennearch' from the benchmark's source.

00,145

00,146 So the result on Core 2 Q9550s @2.83GHz DDR2 @666MHz / i5-2430M @3.00GHz DDR3 @666MHz:

00,147	-----						
00,148	Searcher	GNU/GLIBC memmem()	Railgun_Swampshine	Railgun_Trolldom			
00,149	-----						
00,150	Testfile\Compiler	Intel 15.0   GCC 5.10	Intel 15.0   GCC 5.10	Intel 15.0   GCC 5.10			
00,151	-----						
00,152	Size: 27,703 bytes	4506/-   5330/14725	13198/-   11581/15171	19105/22449	15493/21642		
00,153	Name: An_Interview_with_Carlos_Castaneda.TXT						
00,154	LATENCY-WISE: Number of 'memmem()' Invocations: 308,062						
00,155	THROUGHPUT-WISE: Number of Total bytes Traversed: 3,242,492,648						
00,156	-----						
00,157	Size: 2,347,772 bytes	190/-   226/244	1654/-   1729/1806	1794/1822	1743/1809		
00,158	Name: Gutenberg_EBook_Don_Quixote_996_(ANSI).txt						
00,159	LATENCY-WISE: Number of 'memmem()' Invocations: 14,316,954						
00,160	THROUGHPUT-WISE: Number of Total bytes Traversed: 6,663,594,719,173						
00,161	-----						
00,162	Size: 899,425 bytes	582/-   760/816	3094/-   2898/3088	3255/3289	2915/3322		
00,163	Name: Gutenberg_EBook_Dokoe_by_Hakucho_Masamune_(Japanese_UTF8).txt						
00,164	LATENCY-WISE: Number of 'memmem()' Invocations: 3,465,806						
00,165	THROUGHPUT-WISE: Number of Total bytes Traversed: 848,276,034,315						
00,166	-----						
00,167	Size: 4,487,433 bytes	104/-   109/116	445/-   458/417	450/411	467/425		
00,168	Name: Dragonfly_genome_shotgun_sequence_(ACGT_alphabet).fasta						
00,169	LATENCY-WISE: Number of 'memmem()' Invocations: 20,540,375						
00,170	THROUGHPUT-WISE: Number of Total bytes Traversed: 13,592,530,857,131						
00,171	-----						
00,172	Size: 954,035 bytes	99/-   144/144	629/-   580/682	634/807	585/725		
00,173	Name: LAOTZU_Wu_Wei_(BINARY).pdf						
00,174	LATENCY-WISE: Number of 'memmem()' Invocations: 27,594,933						
00,175	THROUGHPUT-WISE: Number of Total bytes Traversed: 8,702,455,122,519						
00,176	-----						
00,177	Size: 15,583,440 bytes	-/-   -/-	-/-   663/771	675/778	663/757		
00,178	Name: Arabian_Nights_complete.html						
00,179	LATENCY-WISE: Number of 'memmem()' Invocations: 72,313,262						
00,180	THROUGHPUT-WISE: Number of Total bytes Traversed: 105,631,163,854,099						

00,182

00,183 Note0: Railgun\_Trolldom is slightly faster (both with Intel & GCC) than Railgun\_Swampshine, this is mostly due to adding a bitwise BMH order 2 (8KB table overhead instead of 64KB) path - for haystacks <77777 bytes long - the warm-up is faster.

00,184 Note1: The numbers represent the rate (bytes/s) at which patterns/needles 4,5,6,8,9,10,12,13,14,16,17,18,24 bytes long are memmemed into 4KB, 256KB, 1MB, 256MB long haystacks.

00,185 in fact, these numbers are the compression speed using LZSS and memmem() as matchfinder.

00,186 Note2: The Arabian Nights is downloadable at:

00,187 <https://ebooks.adelaide.edu.au/b/burton/richard/b97b/complete.html>

00,188 Note3: On i5-2430M, TW is catching up since this CPU crunches instructions faster while the RAM speed is almost the same, Railgun suffers from the slow RAM fetches - the prefetcher and such suck.

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,287 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):
00,288     if ( ((signed int)(i-(PRIMALposition-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)]+(PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4) ) {
00,289         if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)(&pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,290             count = PRIMALlengthCANDIDATE-4+1;
00,291             while ( count > 0 && *(uint32_t *)(&pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)(&pbTarget[i-(PRIMALposition-1)]+(count-1)) )
00,292                 count = count-4;
00,293             if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
00,294         }
00,295     }
00,296 }
00,297 ...
00,298 */
00,299 // Railgun_Swampshine_BailOut, copyleft 2014-Jan-31, Kaze.
00,300 // Caution: For better speed the case 'if (cbPattern==1)' was removed, so Pattern must be longer than 1 char.
00,301 #define NeedleThreshold2vs4swampLITE 9+10 // Should be bigger than 9. BMH2 works up to this value (inclusive), if bigger then BMH4 takes over. Should be <=255 otherwise the 0!1 BMH2 should be used.
00,302 #define QB64_32bit
00,303 // With QB64 x32 had to be 'uint32_t cbTarget' otherwise doesn't work - _Offset is 32bit and cannot handle 'uint64_t cbTarget'!
00,304 #ifdef QB64_32bit
00,305 char * Railgun_Trolldom_64 (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern)
00,306 #else
00,307 char * Railgun_Trolldom_64 (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern)
00,308 #endif
00,309 {
00,310     char * pbTargetMax = pbTarget + cbTarget;
00,311     register uint32_t ulHashPattern;
00,312     //signed long count;
00,313     signed long long count; // 2020-Jan-11
00,314
00,315     unsigned char bm_Horspool_Order2[256*256]; // Bitwise soon...
00,316     unsigned char bm_Horspool_Order2bitwise[(256*256)>>3]; // Bitwise soon...
00,317     //uint32_t i, Gulliver;
00,318     uint64_t i, Gulliver;
00,319
00,320     //uint32_t PRIMALposition, PRIMALpositionCANDIDATE;
00,321     //uint32_t PRIMALlength, PRIMALlengthCANDIDATE;
00,322     //uint32_t j, FoundAtPosition;
00,323
00,324     uint64_t PRIMALposition, PRIMALpositionCANDIDATE;
00,325     uint64_t PRIMALlength, PRIMALlengthCANDIDATE;
00,326     uint64_t j, FoundAtPosition;
00,327
00,328 // Quadruplet [
00,329     //char * pbTargetMax = pbTarget + cbTarget;
00,330     //register unsigned long ulHashPattern;
00,331     unsigned long ulHashTarget;
00,332     //unsigned long count;
00,333     unsigned long countSTATIC;
00,334     unsigned char SINGLET;
00,335     unsigned long Quadruplet2nd;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,336 unsigned long Quadruplet3rd;
00,337 unsigned long Quadruplet4th;
00,338 unsigned long AdvanceHopperGrass;
00,339 // Quadruplet ]
00,340
00,341 // 2020-Jan-11 [
00,342 // vint64_t A=3123123123, B=5123123123;
00,343 //if ((signed int)A > 0) printf("(signed int)3billion OK\n"); else printf("(signed int)3billion Bug\n");
00,344 //if ((signed int)B > 0) printf("(signed int)5billion OK\n"); else printf("(signed int)5billion Bug\n");
00,345 //if ((signed long long)A > 0) printf("(signed long long)3billion OK\n"); else printf("(signed long long)3billion Bug\n");
00,346 //if ((signed long long)B > 0) printf("(signed long long)5billion OK\n"); else printf("(signed long long)5billion Bug\n");
00,347
00,348 //(signed int)3billion Bug
00,349 //(signed int)5billion OK
00,350 //(signed long long)3billion OK
00,351 //(signed long long)5billion OK
00,352 // 2020-Jan-11 ]
00,353
00,354 //GLOBAL_Railgun_INVOCATIONS++; // 2020-Jan-29
00,355 //GLOBAL_Railgun_INVOCATIONS_ARRAY[cbPattern]++; // 2020-Jan-29
00,356
00,357 if (cbPattern > cbTarget) return(NULL);
00,358
00,359 #ifdef LITE
00,360 return(NULL); // 2020-Feb-14
00,361 #endif
00,362
00,363 if ( cbPattern<4 ) {
00,364     // SSE2 i.e. 128bit Assembly rules here, Mischa knows best:
00,365     // ...
00,366     pbTarget = pbTarget+cbPattern;
00,367     ulHashPattern = ( (*char *) (pbPattern)<<8 ) + *(pbPattern+(cbPattern-1));
00,368     if ( cbPattern==3 ) {
00,369         for ( ;; ) {
00,370             if ( ulHashPattern == ( (*char *) (pbTarget-3)<<8 ) + *(pbTarget-1) ) {
00,371                 if ( (*char *) (pbPattern+1) == (*char *) (pbTarget-2) ) return((pbTarget-3));
00,372             }
00,373             if ( (char)(ulHashPattern>>8) != *(pbTarget-2) ) {
00,374                 pbTarget++;
00,375                 if ( (char)(ulHashPattern>>8) != *(pbTarget-2) ) pbTarget++;
00,376             }
00,377             pbTarget++;
00,378             if (pbTarget > pbTargetMax) return(NULL);
00,379         }
00,380     } else {
00,381     }
00,382     for ( ;; ) {
00,383         if ( ulHashPattern == ( (*char *) (pbTarget-2)<<8 ) + *(pbTarget-1) ) return((pbTarget-2));
00,384         if ( (char)(ulHashPattern>>8) != *(pbTarget-1) ) pbTarget++;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,385         pbTarget++;
00,386         if (pbTarget > pbTargetMax) return(NULL);
00,387     }
00,388 } else { //if ( cbPattern<4 )
00,389     if ( cbPattern<=NeedleThreshold2vs4swampLITE ) {
00,390
00,391 // This is the awesome 'Railgun_Quadruplet', it did outperform EVERYWHERE the fastest strstr (back in old GLIBCes ~2003, by the Dutch hacker Stephen B. van den Berg), suitable for short haystacks ~100bytes.
00,392 // Caution: For better speed the case 'if (cbPattern==1)' was removed, so Pattern must be longer than 1 char.
00,393 // char * Railgun_Quadruplet (char * pbTarget, char * pbPattern, unsigned long cbTarget, unsigned long cbPattern)
00,394 // ...
00,395 //     if (cbPattern > cbTarget) return(NULL);
00,396 //} else { //if ( cbPattern<4)
00,397 if (cbTarget<777) // This value is arbitrary(don't know how exactly), it ensures(at least must) better performance than 'Boyer_Moore_Horspool'.
00,398 {
00,399     pbTarget = pbTarget+cbPattern;
00,400     ulHashPattern = *(unsigned long *) (pbPattern);
00,401 //     countSTATIC = cbPattern-1;
00,402
00,403 //SINGLET = *(char *) (pbPattern);
00,404 SINGLET = ulHashPattern & 0xFF;
00,405 Quadruplet2nd = SINGLET<<8;
00,406 Quadruplet3rd = SINGLET<<16;
00,407 Quadruplet4th = SINGLET<<24;
00,408
00,409 for ( ;; )
00,410 {
00,411     AdvanceHopperGrass = 0;
00,412     ulHashTarget = *(unsigned long *) (pbTarget-cbPattern);
00,413
00,414     if ( ulHashPattern == ulHashTarget ) { // Three unnecessary comparisons here, but 'AdvanceHopperGrass' must be calculated - it has a higher priority.
00,415 //         count = countSTATIC;
00,416 //         while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
00,417 //             if ( countSTATIC==AdvanceHopperGrass+count && SINGLET != *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) AdvanceHopperGrass++;
00,418 //             count--;
00,419 //         }
00,420         count = cbPattern-1;
00,421         while ( count && *(char *) (pbPattern+(cbPattern-count)) == *(char *) (pbTarget-count) ) {
00,422             if ( cbPattern-1==AdvanceHopperGrass+count && SINGLET != *(char *) (pbTarget-count) ) AdvanceHopperGrass++;
00,423             count--;
00,424         }
00,425         if ( count == 0) return((pbTarget-cbPattern));
00,426     } else { // The goal here: to avoid memory accesses by stressing the registers.
00,427         if ( Quadruplet2nd != (ulHashTarget & 0x0000FF00) ) {
00,428             AdvanceHopperGrass++;
00,429             if ( Quadruplet3rd != (ulHashTarget & 0x00FF0000) ) {
00,430                 AdvanceHopperGrass++;
00,431                 if ( Quadruplet4th != (ulHashTarget & 0xFF000000) ) AdvanceHopperGrass++;
00,432             }
00,433         }

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,434     }
00,435
00,436     AdvanceHopperGrass++;
00,437
00,438     pbTarget = pbTarget + AdvanceHopperGrass;
00,439     if (pbTarget > pbTargetMax)
00,440         return(NULL);
00,441     }
00,442 } else if (cbTarget<77777) { // The warmup/overhead is lowered from 64K down to 8K, however the bitwise additional instructions quickly start hurting the throughput/traversal.
00,443 // The below bitwise 0!1 BMH2 gives 1427 bytes/s for 'Don_Quixote' with Intel:
00,444 // The below bitwise 0!1 BMH2 gives 1242 bytes/s for 'Don_Quixote' with GCC:
00,445 // } else { //if ( cbPattern<4 )
00,446 //     if ( cbPattern<=NeedleThreshold2vs4Decumanus ) {
00,447 //         // BMH order 2, needle should be >=4:
00,448 //         ulHashPattern = *(uint32_t *) (pbPattern); // First four bytes
00,449 //         //for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]=0;}
00,450 //         for (i=0; i < (256*256)>>3; i++) {bm_Horspool_Order2bitwise[i]=0;}
00,451 //         //for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[*(unsigned short *) (pbPattern+i)]=1;
00,452 //         for (i=0; i < cbPattern-2+1; i++) bm_Horspool_Order2bitwise[*(unsigned short *) (pbPattern+i)>>3]=bm_Horspool_Order2bitwise[*(unsigned short *) (pbPattern+i)>>3] | (1<<<((*(unsigned short *) (pbPattern+i))&0x7));
00,453 //         i=0;
00,454 //         while (i <= cbTarget-cbPattern) {
00,455 //             Gulliver = 1; // 'Gulliver' is the skip
00,456 //             //if ( bm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1]] != 0 ) {
00,457 //                 if ( ( bm_Horspool_Order2bitwise[*(unsigned short *)&pbTarget[i+cbPattern-1-1]>>3] & (1<<<((*(unsigned short *)&pbTarget[i+cbPattern-1-1])&0x7)) ) != 0 ) {
00,458 //                     //if ( bm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1-2]] == 0 ) Gulliver = cbPattern-(2-1)-2; else {
00,459 //                         if ( ( bm_Horspool_Order2bitwise[*(unsigned short *)&pbTarget[i+cbPattern-1-1-2]>>3] & (1<<<((*(unsigned short *)&pbTarget[i+cbPattern-1-1-2])&0x7)) ) == 0 ) Gulliver =
cbPattern-(2-1)-2; else {
00,460 //                             if ( *(uint32_t *)&pbTarget[i] == ulHashPattern) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,461 //                                 count = cbPattern-4+1;
00,462 //                                 while ( count > 0 && *(uint32_t *) (pbPattern+count-1) == *(uint32_t *) (&pbTarget[i]+(count-1)) )
00,463 //                                     count = count-4;
00,464 //                                 if ( count <= 0 ) return(pbTarget+i);
00,465 //                             }
00,466 //                         }
00,467 //                     } else Gulliver = cbPattern-(2-1);
00,468 //                     i = i + Gulliver;
00,469 //                     //Global++; // Comment it, it is only for stats.
00,470 //                 }
00,471 //             return(NULL);
00,472 //         } else { //if ( cbPattern<=NeedleThreshold2vs4Decumanus )
00,473 //         } else { //if (cbTarget<777)
00,474 //             // BMH order 2, needle should be >=4:
00,475 //             ulHashPattern = *(uint32_t *) (pbPattern); // First four bytes
00,476 //             for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]=0;}
00,477 //             for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[*(unsigned short *) (pbPattern+i)]=1;
00,478 //             i=0;
00,479 //             while (i <= cbTarget-cbPattern) {
00,480 //                 Gulliver = 1; // 'Gulliver' is the skip

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,481         if ( hm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1]] != 0 ) {
00,482             if ( hm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1-2]] == 0 ) Gulliver = cbPattern-(2-1)-2; else {
00,483                 if ( *(uint32_t *)&pbTarget[i] == ulHashPattern ) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,484                     count = cbPattern-4+1;
00,485                     while ( count > 0 && *(uint32_t *)(&pbPattern+count-1) == *(uint32_t *)(&pbTarget[i]+(count-1)) )
00,486                         count = count-4;
00,487                     if ( count <= 0 ) return(pbTarget+i);
00,488                 }
00,489             }
00,490         } else Gulliver = cbPattern-(2-1);
00,491         i = i + Gulliver;
00,492         //GlobalI++; // Comment it, it is only for stats.
00,493     }
00,494     return(NULL);
00,495
00,496 // Slower than Swampshine's simple 0/1 segment:
00,497 /*
00,498 PRIMALlength=0;
00,499 for (i=0+(1); i < cbPattern-2+1+(1)-(1); i++) { // -(1) because the last BB order 2 has no counterpart(s)
00,500     FoundAtPosition = cbPattern;
00,501     PRIMALpositionCANDIDATE=i;
00,502     while ( PRIMALpositionCANDIDATE <= (FoundAtPosition-1) ) {
00,503         j = PRIMALpositionCANDIDATE + 1;
00,504         while ( j <= (FoundAtPosition-1) ) {
00,505             if ( *(unsigned short *)(&pbPattern+PRIMALpositionCANDIDATE-(1)) == *(unsigned short *)(&pbPattern+j-(1)) ) FoundAtPosition = j;
00,506             j++;
00,507         }
00,508         PRIMALpositionCANDIDATE++;
00,509     }
00,510     PRIMALlengthCANDIDATE = (FoundAtPosition-1)-i+(2);
00,511     if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=i; PRIMALlength = PRIMALlengthCANDIDATE;}
00,512 }
00,513 PRIMALlengthCANDIDATE = cbPattern;
00,514 cbPattern = PRIMALlength;
00,515 pbPattern = pbPattern + (PRIMALposition-1);
00,516 if (cbPattern<4) {
00,517     cbPattern = PRIMALlengthCANDIDATE;
00,518     pbPattern = pbPattern - (PRIMALposition-1);
00,519 }
00,520 if (cbPattern == PRIMALlengthCANDIDATE) {
00,521     // BMH order 2, needle should be >=4:
00,522     ulHashPattern = *(uint32_t *)(&pbPattern); // First four bytes
00,523     for (i=0; i < 256*256; i++) {hm_Horspool_Order2[i]=0;}
00,524     for (i=0; i < cbPattern-1; i++) hm_Horspool_Order2[*(unsigned short *)(&pbPattern+i)]=1;
00,525     i=0;
00,526     while (i <= cbTarget-cbPattern) {
00,527         Gulliver = 1; // 'Gulliver' is the skip
00,528         if ( hm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1]] != 0 ) {
00,529             if ( hm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1-2]] == 0 ) Gulliver = cbPattern-(2-1)-2; else {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

00,530         if ( *(uint32_t *)&pbTarget[i] == ulHashPattern) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,531             count = cbPattern-4+1;
00,532             while ( count > 0 && *(uint32_t *)(&pbPattern+count-1) == *(uint32_t *)(&pbTarget[i]+(count-1)) )
00,533                 count = count-4;
00,534             if ( count <= 0 ) return(pbTarget+i);
00,535         }
00,536     }
00,537     } else Gulliver = cbPattern-(2-1);
00,538     i = i + Gulliver;
00,539     //GlobalI++; // Comment it, it is only for stats.
00,540 }
00,541 return(NULL);
00,542 } else { //if (cbPattern == PRIMALlengthCANDIDATE) {
00,543 // BMH Order 2 [
00,544     ulHashPattern = *(uint32_t *)(&pbPattern); // First four bytes
00,545     for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]= cbPattern-1; } // cbPattern-(Order-1) for Horspool; 'memset' if not optimized
00,546     // The above 'for' gives 1424 bytes/s for 'Don_Quixote' with Intel:
00,547     // The above 'for' gives 1431 bytes/s for 'Don_Quixote' with GCC:
00,548     // The below 'memset' gives 1389 bytes/s for 'Don_Quixote' with Intel:
00,549     // The below 'memset' gives 1432 bytes/s for 'Don_Quixote' with GCC:
00,550     //memset(&bm_Horspool_Order2[0], cbPattern-1, 256*256); // Why why? It is 1700:1000 slower than above 'for'!?
00,551     for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[*(unsigned short *)(&pbPattern+i)]=i; // Rightmost appearance/position is needed
00,552     i=0;
00,553     while (i <= cbTarget-cbPattern) {
00,554         Gulliver = bm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1]];
00,555         if ( Gulliver != cbPattern-1 ) { // CASE #2: if equal means the pair (char order 2) is not found i.e. Gulliver remains intact, skip the whole pattern and fall back (Order-1) chars i.e.
one char for Order 2
00,556             if ( Gulliver == cbPattern-2 ) { // CASE #1: means the pair (char order 2) is found
00,557                 if ( *(uint32_t *)&pbTarget[i] == ulHashPattern) {
00,558                     count = cbPattern-4+1;
00,559                     while ( count > 0 && *(uint32_t *)(&pbPattern+count-1) == *(uint32_t *)(&pbTarget[i]+(count-1)) )
00,560                         count = count-4;
00,561 // If we miss to hit then no need to compare the original: Needle
00,562 if ( count <= 0 ) {
00,563 // I have to add out-of-range checks...
00,564 // i-(PRIMALposition-1) >= 0
00,565 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,566 // i-(PRIMALposition-1)+(count-1) >= 0
00,567 // &pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4
00,568
00,569 // "FIX" from 2014-Apr-27:
00,570 // Because (count-1) is negative, above fours are reduced to next twos:
00,571 // i-(PRIMALposition-1)+(count-1) >= 0
00,572 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,573 // The line below is BUGGY:
00,574 //if ( (i-(PRIMALposition-1) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) && (&pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4) ) {
00,575 // The line below is NOT OKAY, in fact so stupid, grrr, not a blunder, not carelessness, but overconfidence in writing "on the fly":
00,576 //if ( ((signed int)(i-(PRIMALposition-1)+(count-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) ) {
00,577 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,578 if ( ((signed long long)(i-(PRIMALposition-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)]+((PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4) ) { // 2020-jan-11
00,579     if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)&(pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,580         count = PRIMALlengthCANDIDATE-4+1;
00,581         while ( count > 0 && *(uint32_t *)&(pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)&(pbTarget[i-(PRIMALposition-1)]+(count-1)) )
00,582             count = count-4;
00,583         if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
00,584     }
00,585 }
00,586 }
00,587     }
00,588     Gulliver = 1;
00,589 } else
00,590     Gulliver = cbPattern - Gulliver - 2; // CASE #3: the pair is found and not as suffix i.e. rightmost position
00,591 }
00,592 i = i + Gulliver;
00,593 //GlobalI++; // Comment it, it is only for stats.
00,594 }
00,595 return(NULL);
00,596 // BMH Order 2 ]
00,597 } //if (cbPattern == PRIMALlengthCANDIDATE) {
00,598 */
00,599
00,600 /*

```

00,601 So the result on Core 2 Q9550s @2.83GHz:

00,602	-----							
00,603	testfile\Searcher	GNU/GLIBC memmem()	Railgun_Swampshine	Railgun_Trolldom				
00,604	-----							
00,605	Compiler	Intel 15.0   GCC 5.10	Intel 15.0   GCC 5.10	Intel 15.0   GCC 5.10				
00,606	-----							
00,607	The_Project_Gutenberg_EBook_of_Don	190	226	1654	1729	1147	1764	
00,608	Quixote_996_(ANSI).txt							
00,609	2,347,772 bytes							
00,610	-----							
00,611	The_Project_Gutenberg_EBook_of_Dokoe	582	760	3094	2898	2410	3036	
00,612	_by_Hakucho_Masamune_(Japanese_UTF-8).txt							
00,613	899,425 bytes							
00,614	-----							
00,615	Dragonfly_genome_shotgun_sequence	104	109	445	458	484	553	
00,616	(ACGT_alphabet).fasta							
00,617	4,487,433 bytes							
00,618	-----							
00,619	LAOTZU_Wu_Wei_(BINARY).pdf	99	144	629	580	185	570	
00,620	954,035 bytes							
00,621	-----							

00,622 Below segment (when compiled with Intel) is very slow, see Railgun\_Trolldom two sub-columns above, compared to GCC:

00,623 \*/

00,624 /\*

00,625 // BMH Order 2 [

00,626 ulHashPattern = \*(uint32\_t \*)&(pbPattern); // First four bytes

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,627         for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]= (cbPattern-1);} // cbPattern-(Order-1) for Horspool; 'memset' if not optimized
00,628         // The above 'for' is translated by Intel as:
00,629 // .B5.21::
00,630 // 0013f 83 c0 40      add eax, 64
00,631 // 00142 66 0f 7f 44 14
00,632 //          60      movdqa XMMWORD PTR [96+rsp+rdx], xmm0
00,633 // 00148 3d 00 00 01 00      cmp eax, 65536
00,634 // 0014d 66 0f 7f 44 14
00,635 //          70      movdqa XMMWORD PTR [112+rsp+rdx], xmm0
00,636 // 00153 66 0f 7f 84 14
00,637 //          80 00 00 00      movdqa XMMWORD PTR [128+rsp+rdx], xmm0
00,638 // 0015c 66 0f 7f 84 14
00,639 //          90 00 00 00      movdqa XMMWORD PTR [144+rsp+rdx], xmm0
00,640 // 00165 89 c2      mov edx, eax
00,641 // 00167 72 d6      jb .B5.21
00,642         //memset(&bm_Horspool_Order2[0], cbPattern-1, 256*256); // Why why? It is 1700:1000 slower than above 'for'!?
00,643         // The above 'memset' is translated by Intel as:
00,644 // 00127 41 b8 00 00 01
00,645 //          00      mov r8d, 65536
00,646 // 0012d 44 8b 26      mov r12d, DWORD PTR [rsi]
00,647 // 00130 e8 fc ff ff ff      call _intel_fast_memset
00,648         // ! The problem is that 256*256, 64KB, is already too much, going bitwise i.e. 8KB is not that better, when 'cbPattern-1' is bigger than 255 - an unsigned char - then
00,649         // we must switch to 0/1 table i.e. present or not. Since we are in 'if ( cbPattern<=NeedleThreshold2vs4swampLITE ) {' branch and NeedleThreshold2vs4swampLITE, by default, is 19 - it is okay to
use 'memset'. !
00,650         for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[(unsigned short *)(&pbPattern+i)]=i; // Rightmost appearance/position is needed
00,651         i=0;
00,652         while (i <= cbTarget-cbPattern) {
00,653             Gulliver = bm_Horspool_Order2[(unsigned short *)&pbTarget[i+cbPattern-1]];
00,654             if ( Gulliver != cbPattern-1 ) { // CASE #2: if equal means the pair (char order 2) is not found i.e. Gulliver remains intact, skip the whole pattern and fall back (Order-1) chars i.e.
one char for Order 2
00,655                 if ( Gulliver == cbPattern-2 ) { // CASE #1: means the pair (char order 2) is found
00,656                     if ( *(uint32_t *)&pbTarget[i] == ulHashPattern ) {
00,657                         count = cbPattern-4+1;
00,658                         while ( count > 0 && *(uint32_t *)&pbPattern+count-1 == *(uint32_t *)&pbTarget[i]+(count-1) )
00,659                             count = count-4;
00,660                         if ( count <= 0 ) return(pbTarget+i);
00,661                     }
00,662                     Gulliver = 1;
00,663                 } else
00,664                     Gulliver = cbPattern - Gulliver - 2; // CASE #3: the pair is found and not as suffix i.e. rightmost position
00,665                 }
00,666                 i = i + Gulliver;
00,667                 //Global++; // Comment it, it is only for stats.
00,668             }
00,669             return(NULL);
00,670 // BMH Order 2 ]
00,671 */
00,672
00,673 } //if (cbTarget<777)

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus ToshibaTxL2 8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,723 // Needle: Sandokan          PRIMALposition=01 PRIMALlength=07 'Sandoka'
00,724 // Needle: shazamish         PRIMALposition=02 PRIMALlength=08 'hazamish'
00,725 // Needle: Simplicius Simplicissimus PRIMALposition=08 PRIMALlength=15 'ius Simplicissi'
00,726 // Needle: domilliaquadringenquattuorquinuagintillion PRIMALposition=01 PRIMALlength=19 'domilliaquadringenq'
00,727 // Needle: DODO              PRIMALposition=02 PRIMALlength=03 'ODO'
00,728 // Needle: DODOD            PRIMALposition=03 PRIMALlength=03 'DOD'
00,729 // Needle: aaadODO          PRIMALposition=02 PRIMALlength=05 'aadOD'
00,730 // Needle: aaadODOD         PRIMALposition=02 PRIMALlength=05 'aadOD'
00,731 // Needle: DODOaaa          PRIMALposition=02 PRIMALlength=05 'ODOaa'
00,732 // Needle: DODODaaa        PRIMALposition=03 PRIMALlength=05 'DODaa'
00,733 /*
00,734 PRIMALlength=0;
00,735 for (i=0+(1); i < cbPattern-2+1+(1)-(1); i++) { // -(1) because the last BB order 2 has no counterpart(s)
00,736     FoundAtPosition = cbPattern;
00,737     PRIMALpositionCANDIDATE=i;
00,738     while ( PRIMALpositionCANDIDATE <= (FoundAtPosition-1) ) {
00,739         j = PRIMALpositionCANDIDATE + 1;
00,740         while ( j <= (FoundAtPosition-1) ) {
00,741             if ( *(unsigned short *) (pbPattern+PRIMALpositionCANDIDATE-(1)) == *(unsigned short *) (pbPattern+j-(1)) ) FoundAtPosition = j;
00,742             j++;
00,743         }
00,744         PRIMALpositionCANDIDATE++;
00,745     }
00,746     PRIMALlengthCANDIDATE = (FoundAtPosition-1)-i+(2);
00,747     if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=i; PRIMALlength = PRIMALlengthCANDIDATE;}
00,748 }
00,749 */
00,750 // Swampwalker_BAILOUT heuristic order 2 (Needle should be bigger than 2) ]
00,751
00,752 /*
00,753 Legend:
00,754 '[' points to BB forming left or right boundary;
00,755 '{' points to BB being searched for;
00,756 '(' position of duplicate and new right boundary;
00,757
00,758         0000000001111111111222222222333
00,759         12345678901234567890123456789012
00,760 Example #1 for Needle: 1234567890qwertyuiopasdfghjklzxcv NewNeedle = '1234567890qwertyuiopasdfghjklzxcv'
00,761 Example #2 for Needle: vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv NewNeedle = 'vv'
00,762 Example #3 for Needle: vvvvvvvvvvBOOMSHAKALAKAvvvvvvvvv NewNeedle = 'vvBOOMSHAKALA'
00,763
00,764
00,765     PRIMALlength=00; FoundAtPosition=33;
00,766 Step 01_00: {}[12]34567890qwertyuiopasdfghjklzxc[v?] ! For position #01 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=01, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,767 Step 01_01: [{12}]34567890qwertyuiopasdfghjklzxc[v?] ! Searching for '12', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,768 Step 01_02: [1{2}3]4567890qwertyuiopasdfghjklzxc[v?] ! Searching for '23', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,769 ...
00,770 Step 01_30: [12]34567890qwertyuiopasdfghjkl{zx}c[v?] ! Searching for 'zx', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,771 Step 01_31: [12]34567890qwertyuiopasdfghjklz{xc}[v?] ! Searching for 'xc', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,772   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,773 Step 02_00: {}1[23]4567890qwertyuiopasdfghjklzxc[v?] ! For position #02 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=02, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,774 Step 02_01: 1[{23}]4567890qwertyuiopasdfghjklzxc[v?] ! Searching for '23', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,775 Step 02_02: 1[2{3}4]567890qwertyuiopasdfghjklzxc[v?] ! Searching for '34', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,776 ...
00,777 Step 02_29: 1[23]4567890qwertyuiopasdfghjkl{zx}c[v?] ! Searching for 'zx', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,778 Step 02_30: 1[23]4567890qwertyuiopasdfghjklz{xc}[v?] ! Searching for 'xc', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,779   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,780 ...
00,781 Step 31_00: {}1234567890qwertyuiopasdfghjklz{xc}[v?] ! For position #31 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=31, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-31+(2)=03 !
00,782 Step 31_01: 1234567890qwertyuiopasdfghjklz[{xc}][v?] ! Searching for 'xc', FoundAtPosition = 33, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(33-1)-31+(2)=03 !
00,783   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,784   Result:
00,785   PRIMALposition=01 PRIMALlength=33, NewNeedle = '1234567890qwertyuiopasdfghjklzxcv'
00,786
00,787
00,788   PRIMALlength=00; FoundAtPosition=33;
00,789 Step 01_00: {}[v]vvvvvvvvvvvvvvvvvvvv[v?] ! For position #01 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=01, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,790 Step 01_01: [{v(v)}v]vvvvvvvvvvvvvvvvvvvv ! Searching for 'vv', FoundAtPosition = 02, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(02-1)-01+(2)=02 !
00,791   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,792 Step 02_00: {}v[vv]vvvvvvvvvvvvvvvvvvvv[v?] ! For position #02 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=02, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,793 Step 02_01: v[{v(v)}v]vvvvvvvvvvvvvvvvvvvv ! Searching for 'vv', FoundAtPosition = 03, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(03-1)-02+(2)=02 !
00,794   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,795 ...
00,796 Step 31_00: {}vvvvvvvvvvvvvvvvvvvv[v][v?] ! For position #31 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=31, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-31+(2)=03 !
00,797 Step 31_01: vvvvvvvvvvvvvvvvvvvvv[{v(v)}v] ! Searching for 'vv', FoundAtPosition = 32, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(32-1)-31+(2)=02 !
00,798   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,799   Result:
00,800   PRIMALposition=31 PRIMALlength=02, NewNeedle = 'vv'
00,801
00,802
00,803   PRIMALlength=00; FoundAtPosition=33;
00,804 Step 01_00: {}[v]vvvvvvvvBOOMSHAKALAKAvvvvvvv[v?] ! For position #01 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=01, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-01+(2)=33 !
00,805 Step 01_01: [{v(v)}v]vvvvvvvvBOOMSHAKALAKAvvvvvvv ! Searching for 'vv', FoundAtPosition = 02, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(02-1)-01+(2)=02 !
00,806   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,807 Step 02_00: {}v[vv]vvvvvvvvBOOMSHAKALAKAvvvvvvv[v?] ! For position #02 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=02, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-02+(2)=32 !
00,808 Step 02_01: v[{v(v)}v]vvvvvvvvBOOMSHAKALAKAvvvvvvv ! Searching for 'vv', FoundAtPosition = 03, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(03-1)-02+(2)=02 !
00,809   if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,810 ...
00,811 Step 09_00: {}vvvvvv[v]BOOMSHAKALAKAvvvvvvv[v?] ! For position #09 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary=09, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-09+(2)=25 !
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,812 Step 09_01: vvvvvvv[{}v]BOOMSHAKALAKA[v]vvvvvvvv ! Searching for 'v', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,813 Step 09_02: vvvvvvv[v{v}B]OOMSHAKALAKA[v]vvvvvvvv ! Searching for 'vB', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,814 Step 09_03: vvvvvvv[vv]{BO}OMSHAKALAKA[v]vvvvvvvv ! Searching for 'BO', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,815 Step 09_04: vvvvvvv[vv]B{OO}MSHAKALAKA[v]vvvvvvvv ! Searching for 'OO', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,816 Step 09_05: vvvvvvv[vv]BO{OM}SHAKALAKA[v]vvvvvvvv ! Searching for 'OM', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,817 Step 09_06: vvvvvvv[vv]BOO{MS}HAKALAKA[v]vvvvvvvv ! Searching for 'MS', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,818 Step 09_07: vvvvvvv[vv]BOOM{SH}AKALAKA[v]vvvvvvvv ! Searching for 'SH', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,819 Step 09_08: vvvvvvv[vv]BOOMS{HA}KALAKA[v]vvvvvvvv ! Searching for 'HA', FoundAtPosition = 24, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(24-1)-09+(2)=16 !
00,820 Step 09_09: vvvvvvv[vv]BOOMSH{AK}AL{AK}Avvvvvvvvv ! Searching for 'AK', FoundAtPosition = 21, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(21-1)-09+(2)=13 !
00,821 Step 09_10: vvvvvvv[vv]BOOMSHA{KA}L{AK}Avvvvvvvvv ! Searching for 'KA', FoundAtPosition = 21, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(21-1)-09+(2)=13 !
00,822 Step 09_11: vvvvvvv[vv]BOOMSHAK{AL}{AK}Avvvvvvvvv ! Searching for 'AL', FoundAtPosition = 21, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(21-1)-09+(2)=13 !
00,823 Step 09_12: vvvvvvv[vv]BOOMSHAKA{L{A}K}Avvvvvvvvv ! Searching for 'LA', FoundAtPosition = 21, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(21-1)-09+(2)=13 !
00,824     if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,825 ...
00,826 Step 31_00: {}vvvvvv[vv]BOOMSHAKALAKAvvvvvvv[v?] ! For position #31 the initial boundaries are PRIMALpositionCANDIDATE=LeftBoundary-31, RightBoundary=FoundAtPosition-1, the CANDIDATE PRIMAL string length is
RightBoundary-LeftBoundary+(2)=(33-1)-31+(2)=03 !
00,827 Step 31_01: vvvvvvvvBOOMSHAKALAKAvvvvvvv[{}v{v}]v ! Searching for 'vv', FoundAtPosition = 32, PRIMALlengthCANDIDATE=RightBoundary-LeftBoundary+(2)=(32-1)-31+(2)=02 !
00,828     if (PRIMALlengthCANDIDATE >= PRIMALlength) {PRIMALposition=PRIMALpositionCANDIDATE; PRIMALlength = PRIMALlengthCANDIDATE;}
00,829     Result:
00,830     PRIMALposition=09 PRIMALlength=13, NewNeedle = 'vvBOOMSHAKALA'
00,831 */
00,832
00,833 // Here we have 4 or bigger NewNeedle, apply order 2 for pbPattern[i+(PRIMALposition-1)] with length 'PRIMALlength' and compare the pbPattern[i] with length 'cbPattern':
00,834 PRIMALlengthCANDIDATE = cbPattern;
00,835 cbPattern = PRIMALlength;
00,836 pbPattern = pbPattern + (PRIMALposition-1);
00,837
00,838 // Revision 2 commented section [
00,839 /*
00,840 if (cbPattern-1 <= 255) {
00,841 // BMH Order 2 [
00,842         ulHashPattern = *(uint32_t *) (pbPattern); // First four bytes
00,843         for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]= cbPattern-1;} // cbPattern-(Order-1) for Horspool; 'memset' if not optimized
00,844         for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[(*(unsigned short *) (pbPattern+i))]=i; // Rightmost appearance/position is needed
00,845         i=0;
00,846         while (i <= cbTarget-cbPattern) {
00,847             Gulliver = bm_Horspool_Order2[(*(unsigned short *) (pbTarget[i+cbPattern-1-1])];
00,848             if ( Gulliver != cbPattern-1 ) { // CASE #2: if equal means the pair (char order 2) is not found i.e. Gulliver remains intact, skip the whole pattern and fall back (Order-1) chars i.e.
one char for Order 2
00,849                 if ( Gulliver == cbPattern-2 ) { // CASE #1: means the pair (char order 2) is found
00,850                     if ( *(uint32_t *) (pbTarget[i] == ulHashPattern) {
00,851                         count = cbPattern-4+1;
00,852                         while ( count > 0 && *(uint32_t *) (pbPattern+count-1) == *(uint32_t *) (pbTarget[i]+(count-1)) )
00,853                             count = count-4;
00,854 // If we miss to hit then no need to compare the original: Needle
00,855 if ( count <= 0 ) {
00,856 // I have to add out-of-range checks...
00,857 // i-(PRIMALposition-1) >= 0
00,858 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,859 // i-(PRIMALposition-1)+(count-1) >= 0
00,860 // &pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4
00,861
00,862 // "FIX" from 2014-Apr-27:
00,863 // Because (count-1) is negative, above fours are reduced to next twos:
00,864 // i-(PRIMALposition-1)+(count-1) >= 0
00,865 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,866 // The line below is BUGGY:
00,867 //if ( (i-(PRIMALposition-1)) >= 0 ) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) && (&pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4) ) {
00,868 // The line below is NOT OKAY, in fact so stupid, grrr, not a blunder, not carelessness, but overconfidence in writing "on the fly":
00,869 //if ( ((signed int)(i-(PRIMALposition-1)+(count-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) ) {
00,870 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):
00,871 if ( ((signed long long)(i-(PRIMALposition-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)]+((PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4) ) { // 2020-jan-11
00,872     if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)(&pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,873         count = PRIMALlengthCANDIDATE-4+1;
00,874         while ( count > 0 && *(uint32_t *)(&pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)(&pbTarget[i-(PRIMALposition-1)]+(count-1)) )
00,875             count = count-4;
00,876         if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
00,877     }
00,878 }
00,879 }
00,880
00,881     Gulliver = 1;
00,882 } else
00,883     Gulliver = cbPattern - Gulliver - 2; // CASE #3: the pair is found and not as suffix i.e. rightmost position
00,884 }
00,885 i = i + Gulliver;
00,886 //GlobalI++; // Comment it, it is only for stats.
00,887 }
00,888 return(NULL);
00,889 // BMH Order 2 ]
00,890 } else {
00,891     // BMH order 2, needle should be >=4:
00,892     ulHashPattern = *(uint32_t *)(&pbPattern); // First four bytes
00,893     for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]=0;}
00,894     for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[(unsigned short *)(&pbPattern+i)]=1;
00,895     i=0;
00,896     while (i <= cbTarget-cbPattern) {
00,897         Gulliver = 1; // 'Gulliver' is the skip
00,898         if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+cbPattern-1-1]] != 0 ) {
00,899             if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+cbPattern-1-1-2]] == 0 ) Gulliver = cbPattern-(2-1)-2; else {
00,900                 if ( *(uint32_t *)&pbTarget[i] == ulHashPattern) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,901                     count = cbPattern-4+1;
00,902                     while ( count > 0 && *(uint32_t *)(&pbPattern+count-1) == *(uint32_t *)(&pbTarget[i]+(count-1)) )
00,903                         count = count-4;
00,904 // If we miss to hit then no need to compare the original: Needle
00,905 if ( count <= 0 ) {
00,906 // I have to add out-of-range checks...
00,907 // i-(PRIMALposition-1) >= 0

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,908 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,909 // i-(PRIMALposition-1)+(count-1) >= 0
00,910 // &pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4
00,911
00,912 // "FIX" from 2014-Apr-27:
00,913 // Because (count-1) is negative, above fours are reduced to next twos:
00,914 // i-(PRIMALposition-1)+(count-1) >= 0
00,915 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,916 // The line below is BUGGY:
00,917 //if ( (i-(PRIMALposition-1)) >= 0 ) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) && (&pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4) ) {
00,918 // The line below is NOT OKAY, in fact so stupid, grrr, not a blunder, not carelessness, but overconfidence in writing "on the fly":
00,919 //if ( ((signed int)(i-(PRIMALposition-1)+(count-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) ) {
00,920 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):
00,921 if ( ((signed long long)(i-(PRIMALposition-1))) >= 0 ) && (&pbTarget[i-(PRIMALposition-1)]+(PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4 ) { // 2020-Jan-11
00,922     if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)(&pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,923         count = PRIMALlengthCANDIDATE-4+1;
00,924         while ( count > 0 && *(uint32_t *)(&pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)(&pbTarget[i-(PRIMALposition-1)]+(count-1)) )
00,925             count = count-4;
00,926         if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
00,927     }
00,928 }
00,929 }
00,930
00,931     }
00,932 } else Gulliver = cbPattern-(2-1);
00,933 i = i + Gulliver;
00,934 //GlobalI++; // Comment it, it is only for stats.
00,935 }
00,936 return(NULL);
00,937 }
00,938 */
00,939 // Revision 2 commented section ]
00,940
00,941 if ( cbPattern<=NeedleThreshold2vs4swampLITE ) {
00,942
00,943     // BMH order 2, needle should be >=4:
00,944     ulHashPattern = *(uint32_t *)(&pbPattern); // First four bytes
00,945     for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]=0;}
00,946     // Above line is translated by Intel as:
00,947 // 0044c 41 b8 00 00 01
00,948 // 00 mov r8d, 65536
00,949 // 00452 44 89 5c 24 20 mov DWORD PTR [32+rsp], r11d
00,950 // 00457 44 89 54 24 60 mov DWORD PTR [96+rsp], r10d
00,951 // 0045c e8 fc ff ff ff call _intel_fast_memset
00,952     for (i=0; i < cbPattern-1; i++) bm_Horspool_Order2[(unsigned short *)(&pbPattern+i)]=1;
00,953     i=0;
00,954     while (i <= cbTarget-cbPattern) {
00,955         Gulliver = 1; // 'Gulliver' is the skip
00,956         if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+cbPattern-1]] != 0 ) {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,957         if ( hm_Horspool_Order2[*(unsigned short *)&pbTarget[i+cbPattern-1-1-2]] == 0 ) Gulliver = cbPattern-(2-1)-2; else {
00,958             if ( *(uint32_t *)&pbTarget[i] == ulHashPattern ) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,959                 count = cbPattern-4+1;
00,960                 while ( count > 0 && *(uint32_t *)&(pbPattern+count-1) == *(uint32_t *)&(pbTarget[i]+(count-1)) )
00,961                     count = count-4;
00,962
00,963         if (cbPattern != PRIMALlengthCANDIDATE) { // No need of same comparison when Needle and NewNeedle are equal!
00,964 // If we miss to hit then no need to compare the original: Needle
00,965 if ( count <= 0 ) {
00,966 // I have to add out-of-range checks...
00,967 // i-(PRIMALposition-1) >= 0
00,968 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,969 // i-(PRIMALposition-1)+(count-1) >= 0
00,970 // &pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4
00,971
00,972 // "FIX" from 2014-Apr-27:
00,973 // Because (count-1) is negative, above fours are reduced to next twos:
00,974 // i-(PRIMALposition-1)+(count-1) >= 0
00,975 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
00,976 // The line below is BUGGY:
00,977 //if ( (i-(PRIMALposition-1) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) && (&pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4) ) {
00,978 // The line below is NOT OKAY, in fact so stupid, grrr, not a blunder, not carelessness, but overconfidence in writing "on the fly":
00,979 //if ( ((signed int)(i-(PRIMALposition-1)+(count-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) ) {
00,980 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):
00,981 if ( ((signed long long)(i-(PRIMALposition-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)]+((PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4) ) { // 2020-jan-11
00,982     if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)&(pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
00,983         count = PRIMALlengthCANDIDATE-4+1;
00,984         while ( count > 0 && *(uint32_t *)&(pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)&(pbTarget[i-(PRIMALposition-1)]+(count-1)) )
00,985             count = count-4;
00,986         if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
00,987     }
00,988 }
00,989 }
00,990 } else { //if (cbPattern != PRIMALlengthCANDIDATE)
00,991     if ( count <= 0 ) return(pbTarget+i);
00,992 }
00,993     }
00,994 }
00,995     } else Gulliver = cbPattern-(2-1);
00,996     i = i + Gulliver;
00,997     //GlobalI++; // Comment it, it is only for stats.
00,998 }
00,999 return(NULL);
01,000
01,001 } else { // if ( cbPattern<=NeedleThreshold2vs4swampLITE )
01,002
01,003     // BMH pseudo-order 4, needle should be >=8+2:
01,004     ulHashPattern = *(uint32_t *)&(pbPattern); // First four bytes
01,005     for (i=0; i < 256*256; i++) {bm_Horspool_Order2[i]=0;}

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,006 // In line below we "hash" 4bytes to 2bytes i.e. 16bit table, how to compute TOTAL number of BBs, 'cbPattern - Order + 1' is the number of BBs for text 'cbPattern' bytes long, for example, for
cbPattern=11 'fastest fox' and Order=4 we have BBs = 11-4+1=8:
01,007 // "fast"
01,008 // "aste"
01,009 // "stes"
01,010 // "test"
01,011 // "est "
01,012 // "st f"
01,013 // "t fo"
01,014 // " fox"
01,015 //for (i=0; i < cbPattern-4+1; i++) bm_Horspool_Order2[(*(unsigned short *) (pbPattern+i+0) + *(unsigned short *) (pbPattern+i+2)) & ((1<<16)-1)]=1;
01,016 //for (i=0; i < cbPattern-4+1; i++) bm_Horspool_Order2[(*(uint32_t *) (pbPattern+i+0)>>16)+*(uint32_t *) (pbPattern+i+0)&0xFFFF) & ((1<<16)-1)]=1;
01,017 // Above line is replaced by next one with better hashing:
01,018 for (i=0; i < cbPattern-4+1; i++) bm_Horspool_Order2[(*(uint32_t *) (pbPattern+i+0)>>(16-1))+*(uint32_t *) (pbPattern+i+0)&0xFFFF) & ((1<<16)-1)]=1;
01,019 i=0;
01,020 while (i <= cbTarget-cbPattern) {
01,021     Gulliver = 1;
01,022     //if ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2]>>16)+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2]&0xFFFF) & ((1<<16)-1) ] != 0 ) { // DWORD #1
01,023     // Above line is replaced by next one with better hashing:
01,024     if ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2]>>(16-1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2]&0xFFFF) & ((1<<16)-1) ] != 0 ) { // DWORD #1
01,025         //if ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]>>16)+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) & ((1<<16)-1) ] == 0 ) Gulliver =
cbPattern-(2-1)-2-4; else {
01,026         // Above line is replaced in order to strengthen the skip by checking the middle DWORD,if the two DWORDs are 'ab' and 'cd' i.e. [2x][2a][2b][2c][2d] then the middle DWORD is
'bc'.
01,027         // The respective offsets (backwards) are: -10/-8/-6/-4 for 'xa'/'ab'/'bc'/'cd'.
01,028         //if ( ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]>>16)+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]&0xFFFF) & ((1<<16)-1) ] + (
bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]>>16)+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) & ((1<<16)-1) ] ) + ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-
2]>>16)+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) & ((1<<16)-1) ] ) < 3 ) Gulliver = cbPattern-(2-1)-2-4-2; else {
01,029         // Above line is replaced by next one with better hashing:
01,030         // When using (16-1) right shifting instead of 16 we will have two different pairs (if they are equal), the highest bit being lost do the job especially for ASCII texts with
no symbols in range 128-255.
01,031         // Example for genomesque pair TT+TT being shifted by (16-1):
01,032         // T           = 01010100
01,033         // TT          = 01010100 01010100
01,034         // TTTT         = 01010100 01010100 01010100 01010100
01,035         // TTTT>>16     = 00000000 00000000 01010100 01010100
01,036         // TTTT>>(16-1) = 00000000 00000000 10101000 10101000 <--- Due to the left shift by 1, the 8th bits of 1st and 2nd bytes are populated - usually they are 0 for English texts
& 'ACGT' data.
01,037         //if ( ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]>>(16-1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]&0xFFFF) & ((1<<16)-1) ] ) + (
bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]>>(16-1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) & ((1<<16)-1) ] ) + ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]>>(16-
1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) & ((1<<16)-1) ] ) < 3 ) Gulliver = cbPattern-(2-1)-2-4-2; else {
01,038         // 'Maximus' uses branched 'if', again.
01,039         if ( \
01,040             ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6 +1]>>(16-1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6 +1]&0xFFFF) & ((1<<16)-1) ] ) == 0 \
01,041             || ( bm_Horspool_Order2[(*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4 +1]>>(16-1))+*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4 +1]&0xFFFF) & ((1<<16)-1) ] ) == 0 \
01,042             ) Gulliver = cbPattern-(2-1)-2-4-2 +1; else {
01,043         // Above line is not optimized (several a SHR are used), we have 5 non-overlapping WORDs, or 3 overlapping WORDs, within 4 overlapping DWORDs so:
01,044         // [2x][2a][2b][2c][2d]
01,045         // DWORD #4

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,046 // [2a] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]>>16) = !SHR to be avoided! <--
01,047 // [2x] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]&0xFFFF) = |
01,048 // DWORD #3 |
01,049 // [2b] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]>>16) = !SHR to be avoided! |<--
01,050 // [2a] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) = ----- |
01,051 // DWORD #2 |
01,052 // [2c] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]>>16) = !SHR to be avoided! |<--
01,053 // [2b] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) = ----- |
01,054 // DWORD #1 |
01,055 // [2d] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-0]>>16) = |
01,056 // [2c] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-0]&0xFFFF) = -----
01,057 //
01,058 // So in order to remove 3 SHR instructions the equal extractions are:
01,059 // DWORD #4
01,060 // [2a] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) = !SHR to be avoided! <--
01,061 // [2x] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]&0xFFFF) = |
01,062 // DWORD #3 |
01,063 // [2b] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) = !SHR to be avoided! |<--
01,064 // [2a] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) = ----- |
01,065 // DWORD #2 |
01,066 // [2c] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-0]&0xFFFF) = !SHR to be avoided! |<--
01,067 // [2b] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) = ----- |
01,068 // DWORD #1 |
01,069 // [2d] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-0]>>16) = |
01,070 // [2c] (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-0]&0xFFFF) = -----
01,071 //if ( ( bm_Horspool_Order2[( (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF)*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-6]&0xFFFF) & ( (1<<16)-1 ) ] ) + (
bm_Horspool_Order2[( (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF)*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) & ( (1<<16)-1 ) ] ) + ( bm_Horspool_Order2[( (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-
0]&0xFFFF)*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-2]&0xFFFF) & ( (1<<16)-1 ) ] ) < 3 ) Gulliver = cbPattern-(2-1)-2-6; else {
01,072 // Since the above Decumanus mumbo-jumbo (3 overlapping lookups vs 2 non-overlapping lookups) is not fast enough we go DuoDecumanus or 3x4:
01,073 // [2y][2x][2a][2b][2c][2d]
01,074 // DWORD #3
01,075 // DWORD #2
01,076 // DWORD #1
01,077 //if ( ( bm_Horspool_Order2[( (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]>>16)*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-4]&0xFFFF) & ( (1<<16)-1 ) ] ) + (
bm_Horspool_Order2[( (*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-8]>>16)*(uint32_t *)&pbTarget[i+cbPattern-1-1-2-8]&0xFFFF) & ( (1<<16)-1 ) ] ) < 2 ) Gulliver = cbPattern-(2-1)-2-8; else {
01,078 //if ( (*(uint32_t *)&pbTarget[i] == ulHashPattern) {
01,079 // Order 4 [
01,080 // Let's try something "outrageous" like comparing with[out] overlap BBs 4bytes long instead of 1 byte back-to-back:
01,081 // Inhere we are using order 4, 'cbPattern - Order + 1' is the number of BBs for text 'cbPattern' bytes long, for example, for cbPattern=11 'fastest fox' and
Order=4 we have BBs = 11-4+1=8:
01,082 //0:"fast" if the comparison failed here, 'count' is 1; 'Gulliver' is cbPattern-(4-1)-7
01,083 //1:"aste" if the comparison failed here, 'count' is 2; 'Gulliver' is cbPattern-(4-1)-6
01,084 //2:"stes" if the comparison failed here, 'count' is 3; 'Gulliver' is cbPattern-(4-1)-5
01,085 //3:"test" if the comparison failed here, 'count' is 4; 'Gulliver' is cbPattern-(4-1)-4
01,086 //4:"est " if the comparison failed here, 'count' is 5; 'Gulliver' is cbPattern-(4-1)-3
01,087 //5:"st f" if the comparison failed here, 'count' is 6; 'Gulliver' is cbPattern-(4-1)-2
01,088 //6:"t fo" if the comparison failed here, 'count' is 7; 'Gulliver' is cbPattern-(4-1)-1
01,089 //7:" fox" if the comparison failed here, 'count' is 8; 'Gulliver' is cbPattern-(4-1)
01,090 count = cbPattern-4+1;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,091                                     // Below comparison is UNIdirectional:
01,092                                     while ( count > 0 && *(uint32_t *)(&pbTarget[i]+(count-1)) == *(uint32_t *)(&pbPattern+count-1) )
01,093                                         count = count-4;
01,094
01,095     if (cbPattern != PRIMALlengthCANDIDATE) { // No need of same comparison when Needle and NewNeedle are equal!
01,096 // count = cbPattern-4+1 = 23-4+1 = 20
01,097 // boomshakalakaZZZZZ[ZZZZ] 20
01,098 // boomshakalakaZZ[ZZZZ]ZZZZ 20-4
01,099 // boomshakala[kazz]ZZZZZZZZ 20-8 = 12
01,100 // boomsha[kala]kaZZZZZZZZZZ 20-12 = 8
01,101 // boo[msha]kalakaZZZZZZZZZZ 20-16 = 4
01,102
01,103 // If we miss to hit then no need to compare the original: Needle
01,104 if ( count <= 0 ) {
01,105 // I have to add out-of-range checks...
01,106 // i-(PRIMALposition-1) >= 0
01,107 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
01,108 // i-(PRIMALposition-1)+(count-1) >= 0
01,109 // &pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4
01,110
01,111 // "FIX" from 2014-Apr-27:
01,112 // Because (count-1) is negative, above fours are reduced to next twos:
01,113 // i-(PRIMALposition-1)+(count-1) >= 0
01,114 // &pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4
01,115 // The line below is BUGGY:
01,116 //if ( (i-(PRIMALposition-1)) >= 0 && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) && (&pbTarget[i-(PRIMALposition-1)+(count-1)] <= pbTargetMax - 4) ) {
01,117 // The line below is NOT OKAY, in fact so stupid, grrr, not a blunder, not carelessness, but overconfidence in writing "on the fly":
01,118 //if ( ((signed int)(i-(PRIMALposition-1)+(count-1)) >= 0) && (&pbTarget[i-(PRIMALposition-1)] <= pbTargetMax - 4) ) {
01,119 // FIX from 2016-Aug-10 (two times failed to do simple boundary checks, pfu):
01,120 if ( ((signed long long)(i-(PRIMALposition-1))) >= 0 && (&pbTarget[i-(PRIMALposition-1)]+((PRIMALlengthCANDIDATE-4+1)-1) <= pbTargetMax - 4) ) { // 2020-jan-11
01,121     if ( *(uint32_t *)&pbTarget[i-(PRIMALposition-1)] == *(uint32_t *)(&pbPattern-(PRIMALposition-1))) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
01,122         count = PRIMALlengthCANDIDATE-4+1;
01,123         while ( count > 0 && *(uint32_t *)(&pbPattern-(PRIMALposition-1)+count-1) == *(uint32_t *)(&pbTarget[i-(PRIMALposition-1)]+(count-1)) )
01,124             count = count-4;
01,125         if ( count <= 0 ) return(pbTarget+i-(PRIMALposition-1));
01,126     }
01,127 }
01,128 }
01,129 } else { //if (cbPattern != PRIMALlengthCANDIDATE)
01,130                                     if ( count <= 0 ) return(pbTarget+i);
01,131 }
01,132
01,133                                     // In order to avoid only-left or only-right WCS the memcmp should be done as left-to-right and right-to-left AT THE SAME TIME.
01,134                                     // Below comparison is BIdirectional. It pays off when needle is 8+++ long:
01,135 // for (count = cbPattern-4+1; count > 0; count = count-4) {
01,136 //     if ( *(uint32_t *)(&pbPattern+count-1) != *(uint32_t *)(&pbTarget[i]+(count-1)) ) {break;};
01,137 //     if ( *(uint32_t *)(&pbPattern+(cbPattern-4+1)-count) != *(uint32_t *)(&pbTarget[i]+(cbPattern-4+1)-count) ) {count = (cbPattern-4+1)-count + (1);
break;} // +(1) because two lookups are implemented as one, also no danger of 'count' being 0 because of the fast check outwith the 'while': if ( *(uint32_t *)&pbTarget[i] == ulHashPattern)
01,138 // }

```

```

01,139 //          if ( count <= 0 ) return(pbTarget+i);
01,140          // Checking the order 2 pairs in mismatched DWORD, all the 3:
01,141          //if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1]] == 0 ) Gulliver = count; // 1 or bigger, as it should
01,142          //if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1+1]] == 0 ) Gulliver = count+1; // 1 or bigger, as it should
01,143          //if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1+1+1]] == 0 ) Gulliver = count+1+1; // 1 or bigger, as it should
01,144          //          if ( bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1]] + bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1+1]] +
bm_Horspool_Order2[(unsigned short *)&pbTarget[i+count-1+1+1]] < 3 ) Gulliver = count; // 1 or bigger, as it should, THE MIN(count,count+1,count+1+1)
01,145          // Above compound 'if' guarantees not that Gulliver > 1, an example:
01,146          // Needle:   fastest tax
01,147          // Window: ...fastast tax...
01,148          // After matching 'tax' vs 'tax' and 'fast' vs 'fast' the mismatched DWORD is 'test' vs 'tast':
01,149          // 'tast' when factorized down to order 2 yields: 'ta','as','st' - all the three when summed give 1+1+1=3 i.e. Gulliver remains 1.
01,150          // Roughly speaking, this attempt maybe has its place in worst-case scenarios but not in English text and even not in ACGT data, that's why I
commented it in original 'Shockeroo'.
01,151          //if ( bm_Horspool_Order2[( (uint32_t *)&pbTarget[i+count-1]>>16)+(uint32_t *)&pbTarget[i+count-1]&0xFFFF) & ( (1<<16)-1 )] == 0 )
Gulliver = count; // 1 or bigger, as it should
01,152          // Above line is replaced by next one with better hashing:
01,153 //          if ( bm_Horspool_Order2[( (uint32_t *)&pbTarget[i+count-1]>>(16-1))+(uint32_t *)&pbTarget[i+count-1]&0xFFFF) & ( (1<<16)-1 )] == 0 )
Gulliver = count; // 1 or bigger, as it should
01,154          // Order 4 ]
01,155          }
01,156      }
01,157      } else Gulliver = cbPattern-(2-1)-2; // -2 because we check the 4 rightmost bytes not 2.
01,158      i = i + Gulliver;
01,159      //Global++; // Comment it, it is only for stats.
01,160  }
01,161  return(NULL);
01,162
01,163      } // if ( cbPattern<=NeedleThreshold2vs4swampLITE )
01,164      } // if ( cbPattern<=NeedleThreshold2vs4swampLITE )
01,165  } //if ( cbPattern<4 )
01,166 }
01,167
01,168 // For short needles, and mainly haystacks, 'Doublet' is quite effective. Consider it or 'Quadruplet'.
01,169 // Fixed version from 2012-Feb-27.
01,170 // Caution: For better speed the case 'if (cbPattern==1)' was removed, so Pattern must be longer than 1 char.
01,171 char * Railgun_Doublet (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern)
01,172 {
01,173     char * pbTargetMax = pbTarget + cbTarget;
01,174     register uint32_t ulHashPattern;
01,175     uint32_t ulHashTarget, count, countSTATIC;
01,176
01,177     if (cbPattern > cbTarget) return(NULL);
01,178
01,179     countSTATIC = cbPattern-2;
01,180
01,181     pbTarget = pbTarget+cbPattern;
01,182     ulHashPattern = (*(uint16_t *)&pbPattern));
01,183

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,184     for ( ;; ) {
01,185         if ( ulHashPattern == (*(uint16_t *) (pbTarget-cbPattern)) ) {
01,186             count = countSTATIC;
01,187             while ( count && *(char *) (pbPattern+2*(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+2*(countSTATIC-count)) ) {
01,188                 count--;
01,189             }
01,190             if ( count == 0 ) return((pbTarget-cbPattern));
01,191         }
01,192         pbTarget++;
01,193         if (pbTarget > pbTargetMax) return(NULL);
01,194     }
01,195 }
01,196
01,197 #define XMMtengu
01,198 // Caution: It doesn't work for needles 1 byte long!
01,199 #ifdef QB64_32bit
01,200 char * Railgun_Nyotengu_XMM_YMM_ZMM (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern) // Last change: 2020-Nov-30
01,201 #else
01,202 char * Railgun_Nyotengu_XMM_YMM_ZMM (char * pbTarget, char * pbPattern, uint64_t cbTarget, uint32_t cbPattern) // Last change: 2020-Nov-30
01,203 //char * Railgun_Nyotengu_XMM_YMM_ZMM (char * pbTarget, char * pbPattern, uint32_t cbTarget, uint32_t cbPattern)
01,204 #endif
01,205
01,206 {
01,207     char * pbTargetMax = pbTarget + cbTarget;
01,208     register uint32_t ulHashPattern;
01,209     register uint32_t ulHashTarget;
01,210     signed long count;
01,211
01,212     unsigned char SINGLET;
01,213     uint32_t Quadruplet2nd;
01,214     uint32_t Quadruplet3rd;
01,215     uint32_t Quadruplet4th;
01,216
01,217     uint32_t AdvanceHopperGrass;
01,218
01,219     size_t i;
01,220     size_t j;
01,221     size_t VECTORchunks;
01,222     uint32_t mask;
01,223     //uint8_t mask;
01,224
01,225 #ifdef XMMtengu
01,226     int SkipWholeVector=16;
01,227     __m128i last4;
01,228     __m128i last1;
01,229     __m128i first2;
01,230     __m128i first4;
01,231     __m128i HaystackVector1;
01,232     __m128i HaystackVector2;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,233     _m128i HaystackVector3;
01,234     _m128i HaystackVector4;
01,235
01,236     _m128i EQD1;
01,237     _m128i EQD2;
01,238     _m128i EQD3;
01,239     _m128i EQD4;
01,240
01,241     _m128i FinalVector12;
01,242     _m128i FinalVector34;
01,243 #endif
01,244
01,245 #ifdef YMMtengu
01,246     int SkipWholeVector=32;
01,247     _m256i last4;
01,248     _m256i last1;
01,249     _m256i first2;
01,250     _m256i first4;
01,251     _m256i HaystackVector1;
01,252     _m256i HaystackVector2;
01,253     _m256i HaystackVector3;
01,254     _m256i HaystackVector4;
01,255
01,256     _m256i EQD1;
01,257     _m256i EQD2;
01,258     _m256i EQD3;
01,259     _m256i EQD4;
01,260
01,261     _m256i FinalVector12;
01,262     _m256i FinalVector34;
01,263 #endif
01,264
01,265 #ifdef ZMMtengu
01,266     int SkipWholeVector=64;
01,267     _m512i last4;
01,268     _m512i last1;
01,269     _m512i first2;
01,270     _m512i first4;
01,271     _m512i HaystackVector1;
01,272     _m512i HaystackVector2;
01,273     _m512i HaystackVector3;
01,274     _m512i HaystackVector4;
01,275
01,276     uint16_t EQD1mask16;
01,277     uint16_t EQD2mask16;
01,278     uint16_t EQD3mask16;
01,279     uint16_t EQD4mask16;
01,280
01,281     uint16_t FinalVector12mask16;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,282   uint16_t FinalVector34mask16;
01,283 #endif
01,284
01,285   if (cbPattern > cbTarget) return(NULL);
01,286
01,287   if ( cbPattern<4 ) { // needle 2..3; SCALAR
01,288
01,289       pbTarget = pbTarget+cbPattern;
01,290       ulHashPattern = ( (*char *) (pbPattern)<<8 ) + *(pbPattern+(cbPattern-1));
01,291       if ( cbPattern==3 ) {
01,292           for ( ;; ) {
01,293               if ( ulHashPattern == ( (*char *) (pbTarget-3)<<8 ) + *(pbTarget-1) ) {
01,294                   if ( (*char *) (pbPattern+1) == (*char *) (pbTarget-2) ) return((pbTarget-3));
01,295               }
01,296               if ( (char)(ulHashPattern>>8) != *(pbTarget-2) ) {
01,297                   pbTarget++;
01,298                   if ( (char)(ulHashPattern>>8) != *(pbTarget-2) ) pbTarget++;
01,299               }
01,300               pbTarget++;
01,301               if (pbTarget > pbTargetMax) return(NULL);
01,302           }
01,303       } else {
01,304       }
01,305       for ( ;; ) {
01,306           if ( ulHashPattern == ( (*char *) (pbTarget-2)<<8 ) + *(pbTarget-1) ) return((pbTarget-2));
01,307           if ( (char)(ulHashPattern>>8) != *(pbTarget-1) ) pbTarget++;
01,308           pbTarget++;
01,309           if (pbTarget > pbTargetMax) return(NULL);
01,310       }
01,311
01,312   } else { // Below: haystack <128; needle >=4; SCALAR
01,313       if (cbTarget<128) { // This value is arbitrary (don't know how exactly), it ensures (at least must) better performance than 'Boyer_Moore_Horspool'.
01,314
01,315           pbTarget = pbTarget+cbPattern;
01,316           ulHashPattern = *(uint32_t *) (pbPattern);
01,317           SINGLET = ulHashPattern & 0xFF;
01,318           Quadruplet2nd = SINGLET<<8;
01,319           Quadruplet3rd = SINGLET<<16;
01,320           Quadruplet4th = SINGLET<<24;
01,321           for ( ;; ) {
01,322               AdvanceHopperGrass = 0;
01,323               ulHashTarget = *(uint32_t *) (pbTarget-cbPattern);
01,324               if ( ulHashPattern == ulHashTarget ) { // Three unnecessary comparisons here, but 'AdvanceHopperGrass' must be calculated - it has a higher priority.
01,325                   count = cbPattern-1;
01,326                   while ( count && (*char *) (pbPattern+(cbPattern-count)) == (*char *) (pbTarget-count) ) {
01,327                       if ( cbPattern-1==AdvanceHopperGrass+count && SINGLET != (*char *) (pbTarget-count) ) AdvanceHopperGrass++;
01,328                       count--;
01,329                   }
01,330                   if ( count == 0 ) return((pbTarget-cbPattern));

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,331 } else { // The goal here: to avoid memory accesses by stressing the registers.
01,332     if ( Quadruplet2nd != (ulHashTarget & 0x0000FF00) ) {
01,333         AdvanceHopperGrass++;
01,334         if ( Quadruplet3rd != (ulHashTarget & 0x00FF0000) ) {
01,335             AdvanceHopperGrass++;
01,336             if ( Quadruplet4th != (ulHashTarget & 0xFF000000) ) AdvanceHopperGrass++;
01,337         }
01,338     }
01,339 }
01,340 AdvanceHopperGrass++;
01,341 pbTarget = pbTarget + AdvanceHopperGrass;
01,342 if (pbTarget > pbTargetMax) return(NULL);
01,343 }
01,344 } else { // Below: haystack >=128; needle >=4; VECTOR
01,345
01,346     // Stage 1: SSE2 or AVX2 i.e. 16 or 32 strides.
01,347     // Stage 2: Dealing with the eventual remainder.
01,348     // Careful! Remainder starts (overlapping with the last 32byte chunk, if Needle<32) at NEXT position to 32*(YMM_Chunks_Traversed)+Order4-Needle_Length = 2*32+4-14 = 54:
01,349     //      Chunk #0      Chunk #1      Remainder
01,350     //      [000000000011111111122222222233][333333334444444445555555556666][6666...
01,351     //      [01234567890123456789012345678901][23456789012345678901234567890123][4567...
01,352     //
01,353     //      Linus Torva lds ! The needle's postfix of order 4 was sought up to 63 (ensuring the 32 bytes skips).
01,354     //      Linus Torv alds ! Then next possible hit is at 54 position, or suffix starting at next to 63 or 63+1=64.
01,355     // The main idea: Stressing the registers as it was done in Quadruplet (the above fastest etude) - outperforms Stephen R. van den Berg's strstr at
01,356     http://www.scs.stanford.edu/histat/src/pkg/uclibc/libc/string/generic/strstr.c
01,357     // __m256i __mm256_cmpeq_epi32 (__m256i a, __m256i b) needs AVX2; the more attractive __mmask8 __mm256_cmpeq_epi32_mask (__m256i a, __m256i b) needs AVX512??
01,358 // Pattern: "Linus Torvalds"
01,359 // Order4:      [      ] skip 32 if not a single occurrence of 'alds' within YMM + (Order - 1) = 32 + 3 = 35 bytes window:
01,360 // Haystack:      "otto.....Torvalds"
01,361 // YMM HaystackVector1:      "otto.....Torva"
01,362 // YMM HaystackVector2:      "tto.....Torval"
01,363 // YMM HaystackVector3:      "to.....Torvald"
01,364 // YMM HaystackVector4:      "o.....Torvalds"
01,365 // YMM Vector1:      "aldsaldsaldsaldsaldsaldsalds"
01,366 // Mask1=(HaystackVector1 eqd Vector1):      0 0 0 0 0 0 0 0 ! 8bit !
01,367 // Mask2=(HaystackVector2 eqd Vector1):      0 0 0 0 0 0 0 0 ! 8bit !
01,368 // Mask3=(HaystackVector3 eqd Vector1):      0 0 0 0 0 0 0 0 ! 8bit !
01,369 // Mask4=(HaystackVector4 eqd Vector1):      0 0 0 0 0 0 0 1 ! 8bit !
01,370 // Result=(Mask1 OR Mask2 OR Mask3 OR Mask4): 0 0 0 0 0 0 0 1 ! 8bit !
01,371
01,372 // printf("&pbPattern[cbPattern - 1 -3] = %s\n",&pbPattern[cbPattern - 1 -3]); //debug
01,373
01,374 #ifdef XMMtengu
01,375     VECTORchunks = cbTarget/SkipWholeVector -1; // in here, ensured at least 7 chunks; in order to avoid past haystack XMM reads - decrease 1 chunk and finish with Scalar_Quadruplet
01,376     // The preemptive search for the first char is slower SIGNIFICANTLY on i7 3rd gen?!
01,377     // last1 = __mm_set1_epi8(*(uint8_t*)&pbPattern[0]); //cbPattern - 1
01,378     // first2 = __mm_set1_epi16(*(uint16_t*)&pbPattern[0]);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,379 first4 = _mm_set1_epi32(*(uint32_t*)&pbPattern[0]);
01,380 for (i = 0; i < VECTORchunks*SkipWholeVector; i += SkipWholeVector) {
01,381 HaystackVector1 = _mm_loadu_si128 ((const __m128i*)(pbTarget + i + 0));
01,382 // EQD1 = _mm_cmpeq_epi8(HaystackVector1, last1);
01,383 // mask = _mm_movemask_epi8( EQD1 );
01,384
01,385 // if ( mask != 0 )
01,386 {
01,387 HaystackVector2 = _mm_loadu_si128 ((const __m128i*)(pbTarget + i + 1));
01,388 EQD1 = _mm_cmpeq_epi32(HaystackVector1, first4);
01,389 EQD2 = _mm_cmpeq_epi32(HaystackVector2, first4);
01,390 HaystackVector3 = _mm_loadu_si128 ((const __m128i*)(pbTarget + i + 2));
01,391 HaystackVector4 = _mm_loadu_si128 ((const __m128i*)(pbTarget + i + 3));
01,392 EQD3 = _mm_cmpeq_epi32(HaystackVector3, first4);
01,393 EQD4 = _mm_cmpeq_epi32(HaystackVector4, first4);
01,394
01,395 FinalVector12 = _mm_or_si128(EQD1, EQD2);
01,396 FinalVector34 = _mm_or_si128(EQD3, EQD4);
01,397
01,398 mask = _mm_movemask_ps( _mm_castsi128_ps(_mm_or_si128(FinalVector12, FinalVector34)) );
01,399 }
01,400
01,401 _mm_prefetch((char*)(pbTarget + 64*64), _MM_HINT_T0);
01,402 #endif
01,403 #ifdef YMMtengu
01,404 VECTORchunks = cbTarget/SkipWholeVector -1; // in here, ensured at least 3 chunks; in order to avoid past haystack YMM reads - decrease 1 chunk and finish with Scalar_Quadruplet
01,405 // last1 = _mm256_set1_epi8(*(uint8_t*)&pbPattern[0]); //cbPattern - 1
01,406 first4 = _mm256_set1_epi32(*(uint32_t*)&pbPattern[0]);
01,407 for (i = 0; i < VECTORchunks*SkipWholeVector; i += SkipWholeVector) {
01,408 HaystackVector1 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 0));
01,409 // EQD1 = _mm256_cmpeq_epi8(HaystackVector1, last1);
01,410 // mask = _mm256_movemask_epi8( EQD1 );
01,411
01,412 // if ( mask != 0 )
01,413 {
01,414 HaystackVector2 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 1));
01,415 EQD1 = _mm256_cmpeq_epi32(HaystackVector1, first4);
01,416 EQD2 = _mm256_cmpeq_epi32(HaystackVector2, first4);
01,417 HaystackVector3 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 2));
01,418 HaystackVector4 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 3));
01,419 EQD3 = _mm256_cmpeq_epi32(HaystackVector3, first4);
01,420 EQD4 = _mm256_cmpeq_epi32(HaystackVector4, first4);
01,421
01,422 FinalVector12 = _mm256_or_si256(EQD1, EQD2);
01,423 FinalVector34 = _mm256_or_si256(EQD3, EQD4);
01,424
01,425 mask = _mm256_movemask_ps( _mm256_castsi256_ps(_mm256_or_si256(FinalVector12, FinalVector34)) );
01,426 }
01,427 _mm_prefetch((char*)(pbTarget + 64*64), _MM_HINT_T0);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,428 #endif
01,429 // The vector/main loop is (0012c-000fd+2)+(00175-0016d+2)= 59 bytes
01,430 /*
01,431 ; mark_description "Intel(R) C++ Compiler XE for applications running on IA-32, Version 15.0.0.108 Build 20140726";
01,432 ; mark_description "-O3 -DMMtengu -D_WIN32_ENVIRONMENT_ -D_N_HIGH_PRIORITY -FeNyotengu_YMM_IntelV150_32bit -FAcs";
01,433
01,434 .B10.21:
01,435
01,436 ;;;      HaystackVector1 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 0));
01,437 ;;;      HaystackVector2 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 1));
01,438 ;;;      EQD1 = _mm256_cmpeq_epi32(HaystackVector1, first4);
01,439 ;;;      EQD2 = _mm256_cmpeq_epi32(HaystackVector2, first4);
01,440 ;;;      HaystackVector3 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 2));
01,441 ;;;      HaystackVector4 = _mm256_loadu_si256((const __m256i*)(pbTarget + i + 3));
01,442 ;;;      EQD3 = _mm256_cmpeq_epi32(HaystackVector3, first4);
01,443 ;;;      EQD4 = _mm256_cmpeq_epi32(HaystackVector4, first4);
01,444 ;;;      FinalVector12 = _mm256_or_si256(EQD1, EQD2);
01,445 ;;;      FinalVector34 = _mm256_or_si256(EQD3, EQD4);
01,446 ;;;      mask = _mm256_movemask_ps(_mm256_castsi256_ps(_mm256_or_si256(FinalVector12, FinalVector34)));
01,447 ;;;      _mm_prefetch((char*)(pbTarget + 64*64), _MM_HINT_T0);
01,448
01,449 000fd 0f 18 8e 00 10
01,450      00 00      prefetcht0 BYTE PTR [4096+esi]
01,451 00104 c5 fd 76 0b      vpcmpeqd ymm1, ymm0, YMMWORD PTR [ebx]
01,452 00108 c5 fd 76 54 32
01,453      01      vpcmpeqd ymm2, ymm0, YMMWORD PTR [1+edx+esi]
01,454 0010e c5 fd 76 5c 32
01,455      02      vpcmpeqd ymm3, ymm0, YMMWORD PTR [2+edx+esi]
01,456 00114 c5 fd 76 64 32
01,457      03      vpcmpeqd ymm4, ymm0, YMMWORD PTR [3+edx+esi]
01,458 0011a c5 f5 eb ea      vpor ymm5, ymm1, ymm2
01,459 0011e c5 e5 eb f4      vpor ymm6, ymm3, ymm4
01,460 00122 c5 d5 eb fe      vpor ymm7, ymm5, ymm6
01,461 00126 c5 fc 50 cf      vmovmskps ecx, ymm7
01,462
01,463 ;;;      if ( mask != 0 ) {
01,464
01,465 0012a 85 c9      test ecx, ecx
01,466 0012c 74 3f      je .B10.26
01,467
01,468 ...
01,469
01,470 .B10.26:
01,471 0016d 83 c2 20      add edx, 32
01,472 00170 83 c3 20      add ebx, 32
01,473 00173 3b d0      cmp edx, eax
01,474 00175 72 86      jb .B10.21
01,475
01,476 .B10.28:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,477 */
01,478
01,479 // __mmask16 __mm256_cmpeq_epi16_mask ( __m256i a, __m256i b)
01,480 // __mmask16 __mm512_cmpeq_epi32_mask ( __m512i a, __m512i b)
01,481 // __mmask8 __mm256_cmpeq_epi32_mask ( __m256i a, __m256i b)
01,482 #ifdef ZMMtengu
01,483     VECTORchunks = cbTarget/SkipWholeVector -1; // in here, ensured at least 128/64-1=1 chunk; in order to avoid past haystack ZMM reads - decrease 1 chunk and finish with Scalar_Quadruplet
01,484     first4 = __mm512_set1_epi32(*(uint32_t*)&pbPattern[0]);
01,485     for (i = 0; i < VECTORchunks*SkipWholeVector; i += SkipWholeVector) {
01,486         HaystackVector1 = __mm512_loadu_si512((const __m256i*)(pbTarget + i + 0));
01,487         HaystackVector2 = __mm512_loadu_si512((const __m256i*)(pbTarget + i + 1));
01,488         EQD1mask16 = __mm512_cmpeq_epi32_mask(HaystackVector1, first4);
01,489         EQD2mask16 = __mm512_cmpeq_epi32_mask(HaystackVector2, first4);
01,490         HaystackVector3 = __mm512_loadu_si512((const __m256i*)(pbTarget + i + 2));
01,491         HaystackVector4 = __mm512_loadu_si512((const __m256i*)(pbTarget + i + 3));
01,492         EQD3mask16 = __mm512_cmpeq_epi32_mask(HaystackVector3, first4);
01,493         EQD4mask16 = __mm512_cmpeq_epi32_mask(HaystackVector4, first4);
01,494
01,495         FinalVector12mask16 = (EQD1mask16 | EQD2mask16);
01,496         FinalVector34mask16 = (EQD3mask16 | EQD4mask16);
01,497
01,498         mask = (FinalVector12mask16 | FinalVector34mask16);
01,499
01,500         __mm_prefetch((char*)(pbTarget + 64*64), __MM_HINT_T0);
01,501 #endif
01,502 // The vector/main loop is (00143-000fc+2)+(00187-0017d+6)= 89 bytes
01,503 /*
01,504 ; mark_description "Intel(R) C++ Compiler XE for applications running on IA-32, Version 15.0.0.108 Build 20140726";
01,505 ; mark_description "-O3 -DZMMtengu -D_WIN32_ENVIRONMENT_ -D_N_HIGH_PRIORITY -FeNyotengu_ZMM_IntelV150_32bit -Facs";
01,506
01,507 ;; #ifdef ZMMtengu
01,508 ;;     VECTORchunks = cbTarget/SkipWholeVector -1; // in here, ensured at least 128/64-1=1 chunk; in order to avoid past haystack ZMM reads - decrease 1 chunk and finish with Scalar_Quadruplet
01,509 ;;     first4 = __mm512_set1_epi32(*(uint32_t*)&pbPattern[0]);
01,510
01,511     000df 8b 45 0c      mov eax, DWORD PTR [12+ebp]
01,512     000e2 8d 53 c0      lea edx, DWORD PTR [-64+ebx]
01,513
01,514 ;;     for (i = 0; i < VECTORchunks*SkipWholeVector; i += SkipWholeVector) {
01,515
01,516     000e5 83 e2 c0      and edx, -64
01,517     000e8 62 f2 7d 48 58     vpbroadcastd zmm0, DWORD PTR [eax]
01,518     00      00      vpbroadcastd zmm0, DWORD PTR [eax]
01,519     000ee 0f 84 99 00 00     je .B10.28
01,520     00
01,521
01,522 .B10.20:
01,523     000f4 89 54 24 44     mov DWORD PTR [68+esp], edx
01,524     000f8 33 c0      xor eax, eax
01,525     000fa 8b de      mov ebx, esi

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,526
01,527 .B10.21:
01,528
01,529 ;;      HaystackVector1 = _mm512_loadu_si512((const __m256i*)(pbTarget + i + 0));
01,530 ;;      HaystackVector2 = _mm512_loadu_si512((const __m256i*)(pbTarget + i + 1));
01,531 ;;      EQD1mask16 = _mm512_cmpeq_epi32_mask(HaystackVector1, first4);
01,532 ;;      EQD2mask16 = _mm512_cmpeq_epi32_mask(HaystackVector2, first4);
01,533 ;;      HaystackVector3 = _mm512_loadu_si512((const __m256i*)(pbTarget + i + 2));
01,534 ;;      HaystackVector4 = _mm512_loadu_si512((const __m256i*)(pbTarget + i + 3));
01,535 ;;      EQD3mask16 = _mm512_cmpeq_epi32_mask(HaystackVector3, first4);
01,536 ;;      EQD4mask16 = _mm512_cmpeq_epi32_mask(HaystackVector4, first4);
01,537 ;;      FinalVector12mask16 = (EQD1mask16 | EQD2mask16);
01,538 ;;      FinalVector34mask16 = (EQD3mask16 | EQD4mask16);
01,539 ;;      mask = (FinalVector12mask16 | FinalVector34mask16);
01,540 ;;      _mm_prefetch((char*)(pbTarget + 64*64), _MM_HINT_T0);
01,541
01,542 000fc 8b 7d 08      mov edi, DWORD PTR [8+ebp]
01,543 000ff 0f 18 8f 00 10
01,544      00 00          prefetcht0 BYTE PTR [4096+edi]
01,545 00106 62 f1 7d 48 76
01,546      03          vpcmpeqd k0, zmm0, ZMMWORD PTR [ebx]
01,547 0010c 62 f1 7d 48 76
01,548      8c 38 01 00 00
01,549      00          vpcmpeqd k1, zmm0, ZMMWORD PTR [1+eax+edi]
01,550 00117 62 f1 7d 48 76
01,551      94 38 02 00 00
01,552      00          vpcmpeqd k2, zmm0, ZMMWORD PTR [2+eax+edi]
01,553 00122 62 f1 7d 48 76
01,554      9c 38 03 00 00
01,555      00          vpcmpeqd k3, zmm0, ZMMWORD PTR [3+eax+edi]
01,556 0012d c5 f8 93 f0      kmovw esi, k0
01,557 00131 c5 f8 93 c9      kmovw ecx, k1
01,558 00135 c5 f8 93 d2      kmovw edx, k2
01,559 00139 c5 f8 93 fb      kmovw edi, k3
01,560 0013d 0b f1          or esi, ecx
01,561 0013f 0b d7          or edx, edi
01,562 00141 0b f2          or esi, edx
01,563 ;;      if ( mask != 0 ) {
01,564 00143 74 38          je .B10.26
01,565
01,566 ...
01,567
01,568 .B10.26:
01,569 0017d 83 c0 40      add eax, 64
01,570 00180 83 c3 40      add ebx, 64
01,571 00183 3b 44 24 44      cmp eax, DWORD PTR [68+esp]
01,572 00187 0f 82 6f ff ff
01,573      ff          jb .B10.21
01,574

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,575 .B10.28:
01,576 */
01,577
01,578
01,579 //      printf("mask = %02x\n", mask); //debug
01,580 if ( mask != 0 ) {
01,581 //      printf("_mm_popcnt_u32(mask) = %d\n", _mm_popcnt_u32(mask)); //debug
01,582
01,583 // For these two:
01,584 // char *Haystack = "CPU Benchmark: Linus Torvalds ..... Linus Torvalds"; // 128 bytes long
01,585 // char *Needle = "Linus Torvalds"; // 14 bytes long
01,586 // the 'debug' outcome is:
01,587 // &pbPattern[cbPattern - 1 -3] = alds
01,588 // mask = 40
01,589 // _mm_popcnt_u32(mask) = 1
01,590 // Okay, 0x40 is 0000 0010 (LSB first) i.e. 6th bit is set, it means 4 possible positions within Chunk #6 (4*6=24 offset) (in fact it is only 25th):
01,591 // DWORD #0DWORD #1DWORD #2DWORD #3DWORD #4DWORD #5DWORD #6DWORD #7
01,592 // [00..03][04..07][08..11][12..15][16..19][20..23][24..27][28..31]
01,593 //
01,594 //
01,595 //
01,596 //
01,597 //      000000000001111111112222[2222]2233
01,598 //      012345678901234567890123[4567]8901
01,599 //      CPU Benchmark: Linus Tor[vald]s ..
01,600
01,601 // Manually find the first suffix position:
01,602 //j = i; // somewhere in chunk #i lie possible POPCNT(mask) matches...
01,603 // Okay, doing it dirty as a start - checking all the 16/32/64 positions one-by-one:
01,604 for (j = 0; j < SkipWholeVector; j++)
01,605     if (memcmp(pbTarget + i + j, &pbPattern[cbPattern - cbPattern], cbPattern) == 0) return( pbTarget + i + j ); //first4
01,606 // pbTarget + i + j points to offset of DWORD/suffix so we have to repoint it to the start offset, namely, pbTarget + i + j - (cbPattern-4)
01,607 //while (memcmp(pbTarget + j, &pbPattern[cbPattern - 1 -3], 4) != 0) j++;
01,608 // Don't forget! Comparing the rest of the Needle (to the left) has to be boundary checked (to not go outside) - for speed, this check has to be done outside this loop.
01,609 /*
01,610
01,611     if ( *(uint32_t *)&pbTarget[i] == ulHashPattern) { // This fast check ensures not missing a match (for remainder) when going under 0 in loop below:
01,612         // Order 4 [
01,613         // Let's try something "outrageous" like comparing with[out] overlap BBs 4bytes long instead of 1 byte back-to-back:
01,614         // Inhere we are using order 4, 'cbPattern - Order + 1' is the number of BBs for text 'cbPattern' bytes long, for example, for cbPattern=11 'fastest fox' and Order=4 we have
01,615         //BBs = 11-4+1=8:
01,616         //0:"fast" if the comparison failed here, 'count' is 1; 'Gulliver' is cbPattern-(4-1)-7
01,617         //1:"aste" if the comparison failed here, 'count' is 2; 'Gulliver' is cbPattern-(4-1)-6
01,618         //2:"stes" if the comparison failed here, 'count' is 3; 'Gulliver' is cbPattern-(4-1)-5
01,619         //3:"test" if the comparison failed here, 'count' is 4; 'Gulliver' is cbPattern-(4-1)-4
01,620         //4:"est " if the comparison failed here, 'count' is 5; 'Gulliver' is cbPattern-(4-1)-3
01,621         //5:"st f" if the comparison failed here, 'count' is 6; 'Gulliver' is cbPattern-(4-1)-2
01,622         //6:"t fo" if the comparison failed here, 'count' is 7; 'Gulliver' is cbPattern-(4-1)-1
01,623         //7:" fox" if the comparison failed here, 'count' is 8; 'Gulliver' is cbPattern-(4-1)
01,624         count = cbPattern-4+1;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,623 //count = count-4; // Double-beauty here of already being checked 'ulHashTarget' and not polluting/repeating the final lookup below.
01,624 while ( count > 0 && *(uint32_t *) (pbPattern+count-1) == *(uint32_t *) (&pbTarget[i]+(count-1)) )
01,625     count = count-4; // - order, of course order 4 is much more SWEET&CHEAP - less loops
01,626 if ( count <= 0 )
01,627     return(pbTarget+i);
01,628 // Order 4 ]
01,629 }
01,630 */
01,631
01,632 } //if ( mask != 0 ) {
01,633 } //for ( i = 0; i <
01,634
01,635 // Deal with the remainder (starts right after the last chunk) with Scalar code [
01,636 pbTarget = pbTarget+ i; // 'i' has to be the traversed pool by the vector
01,637 //if (cbPattern > cbTarget) return(NULL);
01,638 // Above check precedes all Railguns, inhere 'cbTarget' is the HaystackLen-i i.e. the remainder
01,639 if (cbPattern > cbTarget-(i)) return(NULL);
01,640
01,641 pbTarget = pbTarget+cbPattern;
01,642 ulHashPattern = *(uint32_t *) (pbPattern);
01,643 SINGLET = ulHashPattern & 0xFF;
01,644 Quadruplet2nd = SINGLET<<8;
01,645 Quadruplet3rd = SINGLET<<16;
01,646 Quadruplet4th = SINGLET<<24;
01,647 for ( ;; ) {
01,648     AdvanceHopperGrass = 0;
01,649     ulHashTarget = *(uint32_t *) (pbTarget-cbPattern);
01,650     if ( ulHashPattern == ulHashTarget ) { // Three unnecessary comparisons here, but 'AdvanceHopperGrass' must be calculated - it has a higher priority.
01,651         count = cbPattern-1;
01,652         while ( count && *(char *) (pbPattern+(cbPattern-count)) == *(char *) (pbTarget-count) ) {
01,653             if ( cbPattern-1==AdvanceHopperGrass+count && SINGLET != *(char *) (pbTarget-count) ) AdvanceHopperGrass++;
01,654             count--;
01,655         }
01,656         if ( count == 0 ) return((pbTarget-cbPattern));
01,657     } else { // The goal here: to avoid memory accesses by stressing the registers.
01,658         if ( Quadruplet2nd != (ulHashTarget & 0x0000FF00) ) {
01,659             AdvanceHopperGrass++;
01,660             if ( Quadruplet3rd != (ulHashTarget & 0x00FF0000) ) {
01,661                 AdvanceHopperGrass++;
01,662                 if ( Quadruplet4th != (ulHashTarget & 0xFF000000) ) AdvanceHopperGrass++;
01,663             }
01,664         }
01,665     }
01,666     AdvanceHopperGrass++;
01,667     pbTarget = pbTarget + AdvanceHopperGrass;
01,668     if (pbTarget > pbTargetMax) return(NULL);
01,669 }
01,670 // Deal with the remainder (starts right after the last chunk) with Scalar code ]
01,671

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,672      } //if (cbTarget<128) {
01,673      } //if ( cbPattern<4 ) { needle 2..3; SCALAR
01,674
01,675 return(NULL);
01,676 }
01,677
01,678 // https://software.intel.com/sites/landingpage/IntrinsicsGuide/#expand=233,272&text=_mm_aesenc_si128
01,679 /*
01,680 __m128i _mm_aesenc_si128 (__m128i a, __m128i RoundKey)
01,681 Synopsis
01,682 __m128i _mm_aesenc_si128 (__m128i a, __m128i RoundKey)
01,683 #include <wmmintrin.h>
01,684 Instruction: aesenc xmm, xmm
01,685 CPUID Flags: AES
01,686 Description
01,687 Perform one round of an AES encryption flow on data (state) in a using the round key in RoundKey, and store the result in dst."
01,688 Operation
01,689 a[127:0] := ShiftRows(a[127:0])
01,690 a[127:0] := SubBytes(a[127:0])
01,691 a[127:0] := MixColumns(a[127:0])
01,692 dst[127:0] := a[127:0] XOR RoundKey[127:0]
01,693
01,694 Performance
01,695 Architecture Latency Throughput (CPI)
01,696 Skylake      4      1
01,697 Broadwell    7      1
01,698 Haswell      7      1
01,699 Ivy Bridge   8      1
01,700 */
01,701
01,702 /*
01,703 #include <stdlib.h>
01,704 #include <stdint.h> // uint64_t needed
01,705 #include <string.h> // memset
01,706 #include <smmintrin.h> // SSE4.1 intrinsics
01,707 #include <wmmintrin.h>
01,708 void SlowCopy128bit (const char *SOURCE, char *TARGET) { _mm_storeu_si128((__m128i *) (TARGET), _mm_loadu_si128((const __m128i *) (SOURCE))); }
01,709 unsigned char DDAES[16];
01,710 void DoubleDeuceAES(const uint8_t *buffer, const size_t length) {
01,711     uint32_t i;
01,712     char MaxTo64a[64], MaxTo64b[64], MaxTo64c[64], MaxTo64d[64];
01,713     __m128i hashA = _mm_set_epi64x(0x6c62272e07bb0142, 0x62b821756295c58d); // 0x6c62272e07bb014262b821756295c58d // _mm_setzero_si128();
01,714     __m128i hashB = _mm_set_epi64x(0xdd268dbcaac55036, 0x2d98c384c4e576cc); // 0xdd268dbcaac550362d98c384c4e576ccc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
01,715     __m128i hashC = _mm_set_epi64x(0xc8b1536847b6bbb3, 0x1023b4c8caee0535); // 0xdd268dbcaac550362d98c384c4e576ccc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
01,716     __m128i hashD = _mm_setzero_si128();
01,717     const __m128i *ptr128a, *ptr128b, *ptr128c, *ptr128d;
01,718     memset(MaxTo64a,0x33,4*(128/8)); // padding the keys to be multiples of 128, up to 64 bytes
01,719     memset(MaxTo64b,0x77,4*(128/8)); // padding the keys to be multiples of 128, up to 64 bytes
01,720     for (i = 0; i < length; i++) {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,721         MaxTo64a[i]=buffer[i];
01,722         MaxTo64b[63-i]=buffer[i]; // MaxTo64b[63-i]=MaxTo64a[i];
01,723     }
01,724     for (i = 0; i < (64>>1)/1; i++) { // 64/2/BYTE=31 i.e 0..31
01,725         MaxTo64c[(i<<1)+0]=MaxTo64a[i+0]; // a: 00,32 / 01,33 / ...31,63
01,726         MaxTo64c[(i<<1)+1]=MaxTo64a[i+32]; // c: 0*2+0,0*2+1 / 1*2+0,1*2+1 / 2*2+0,2*2+1 which is 0,1 / 2,3 / 4,5
01,727         MaxTo64d[(i<<1)+0]=MaxTo64b[i+0];
01,728         MaxTo64d[(i<<1)+1]=MaxTo64b[i+32];
01,729     }
01,730     ptr128a=(_m128i *)MaxTo64a;
01,731     ptr128b=(_m128i *)MaxTo64b;
01,732     ptr128c=(_m128i *)MaxTo64c;
01,733     ptr128d=(_m128i *)MaxTo64d;
01,734     for (i = 0; i < 64 / 16; i++) {
01,735         _m128i a = _mm_loadu_si128(ptr128a++);
01,736         _m128i b = _mm_loadu_si128(ptr128b++);
01,737         _m128i c = _mm_loadu_si128(ptr128c++);
01,738         _m128i d = _mm_loadu_si128(ptr128d++);
01,739         hashA = _mm_aesenc_si128(hashA, a);
01,740         hashB = _mm_aesenc_si128(hashB, b);
01,741         hashC = _mm_aesenc_si128(hashC, c);
01,742         hashD = _mm_aesenc_si128(hashD, d);
01,743     }
01,744     hashA = _mm_aesenc_si128(hashA, hashB);
01,745     hashA = _mm_aesenc_si128(hashA, hashC);
01,746     hashA = _mm_aesenc_si128(hashA, hashD);
01,747     SlowCopy128bit( (const char *)&hashA, (char *)&DDAES[0]);
01,748 }
01,749 */
01,750
01,751 //static const uint8_t VectorsNeedNonVariable1[256] __attribute__((aligned(16))) =
01,752 static const uint8_t VectorsNeedNonVariable1[256] =
01,753 {
01,754     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,755     0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,756     0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,757     0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,758     0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,759     0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,760     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,761     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,762     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,763     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,764     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,
01,765     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,0x00,
01,766     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,0x00,
01,767     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,0x00,
01,768     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00,0x00,
01,769     0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0x00

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,770 };
01,771 static const _m128i *Jumbotron = (_m128i *) VectorsNeedNonVariable1;
01,772 //static const uint8_t VectorsNeedNonVariable2[256] __attribute__((aligned(16))) =
01,773 static const uint8_t VectorsNeedNonVariable2[256] =
01,774 {
01,775     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
01,776     0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,777     0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,778     0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,779     0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,780     0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,781     0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,782     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,783     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,784     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,785     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,786     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,787     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,788     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,789     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,790     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,
01,791 };
01,792 static const _m128i *Jumbotron = (_m128i *) VectorsNeedNonVariable2;
01,793
01,794 // https://github.com/Cyan4973/xxHash/issues/568
01,795 // https://github.com/google/highwayhash/issues/102
01,796 // Revision 2021-Aug-28 [
01,797 // Written by Sanmayce, inspired by J. Andrew Rogers's https://github.com/jandrewrogers/AquaHash/blob/master/aquahash.h
01,798 // This hash function serves two ... functions - useful for table lookups and to shrink keys (usually 64...256 bytes in length) down to 16 bytes:
01,799 // Its linear speed is quite good - 8+GB/s on Zen 2 Renoir 4.3GHz, DDR4 3200MHz.
01,800 // Some non-synthetic speed measurements:
01,801 /*
01,802 Testfile: KAZE(Dictionary_Specification_Language(ABBY Software_House))Hanyu_Cihai_new_Sea-of-Words(Zho-Zho).dsl (42,920,232 bytes)
01,803 Testmachine: Testmachine: laptop 'Brutalitto' AMD 4800H max turbo 4.3GHz, 64GB DDR4 3200MHz, Windows 10
01,804 Hashtable: 26bit, i.e. 67,108,864 slots, greater than (42,920,232 bytes), since in case of perfect hasher - slots should be more than the keys (could be all unique) at each position
01,805
01,806 +-----+-----+-----+-----+
01,807 | Hasher,          | Number Of Hash Collisions = | RAW Hashing Speed (in one pass, | Linear Hashing Speed,      |
01,808 | GCC-10.1 compiler | Distinct Keys -            | at each position) for keys      | the whole file as one key  |
01,809 | -O3 -mavx        | Number Of Trees            | 4,6,8,10,12,14,16,18,36,64 bytes |                             |
01,810 +-----+-----+-----+-----+
01,811 | XXH3_64bits v0.8.0 | 41,108,202 | 295,187,276 KEYS-PER-SECOND | 21,786,919,796 BYTES-PER-SECOND |
01,812 | HighwayHash128 (generic) | 41,109,295 | 5,986,502 KEYS-PER-SECOND | 1,644,642,372 BYTES-PER-SECOND |
01,813 | CRC32C (mm_crc32_u32) | 41,109,478 | 274,426,023 KEYS-PER-SECOND | 5,241,205,519 BYTES-PER-SECOND |
01,814 | XXH3_128bits v0.8.0 | 41,111,196 | 214,493,903 KEYS-PER-SECOND | 20,331,706,300 BYTES-PER-SECOND |
01,815 | SHA3-224 | 41,111,291 | 153,854 KEYS-PER-SECOND | 22,319,413 BYTES-PER-SECOND |
01,816 | wyhash final | 41,112,870 | 449,897,589 KEYS-PER-SECOND | 15,086,197,539 BYTES-PER-SECOND |
01,817 | DoubleDeuceAES_Gumbotron | 41,117,352 | 204,869,832 KEYS-PER-SECOND | 8,690,065,195 BYTES-PER-SECOND |
01,818 | FNV1A_Pippip | 41,488,327 | 449,897,589 KEYS-PER-SECOND | 8,101,214,043 BYTES-PER-SECOND |

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,819 +-----+
01,820 Note1: The second column houses the cumulative value for all collisions, the collisions for all orders 4..64 were summed, that is.
01,821 Note2: Folding of those 128bits should lessen the collisions.
01,822
01,823 Testfile: TERAPIG_Encyclopaedia_Judaica_(in_22_volumes)_TXT.tar (107,784,192 bytes)
01,824 Testmachine: Testmachine: laptop 'Brutalitto' AMD 4800H max turbo 4.3GHz, 64GB DDR4 3200MHz, Windows 10
01,825 Hashtable: 27bit, i.e. 134,217,728 slots, greater than (107,784,192 bytes), since in case of perfect hasher - slots should be more than the keys (could be all unique) at each position
01,826
01,827 +-----+
01,828 | Hasher,          | Number Of Hash Collisions = | RAW Hashing Speed (in one pass, | Linear Hashing Speed, |
01,829 | GCC-10.1 compiler | Distinct Keys -           | at each position) for keys | the whole file as one key |
01,830 | -O3 -mavx        | Number Of Trees           | 4,6,8,10,12,14,16,18,36,64 bytes |
01,831 +-----+
01,832 | DoubleDeuceAES_Gumbotron | 135,752,271 | 204,640,573 KEYS-PER-SECOND | 8,742,330,440 BYTES-PER-SECOND |
01,833 | HighwayHash128 (generic) | 135,754,873 | 6,336,146 KEYS-PER-SECOND | 1,435,801,622 BYTES-PER-SECOND |
01,834 | XXH3_128bits v0.8.0     | 135,756,978 | 212,843,977 KEYS-PER-SECOND | 22,539,563,362 BYTES-PER-SECOND |
01,835 | wyhash final            | 135,762,454 | 442,100,861 KEYS-PER-SECOND | 14,959,638,029 BYTES-PER-SECOND |
01,836 | XXH3_64bits v0.8.0      | 135,763,366 | 290,994,033 KEYS-PER-SECOND | 22,464,400,166 BYTES-PER-SECOND |
01,837 | CBC32C (_mm_crc32_u32)   | 135,764,628 | 252,599,460 KEYS-PER-SECOND | 5,241,402,061 BYTES-PER-SECOND |
01,838 | FNV1A_Pippip            | 135,768,302 | 450,602,801 KEYS-PER-SECOND | 8,048,401,433 BYTES-PER-SECOND |
01,839 | SHA3-224                | 135,771,905 | 153,841 KEYS-PER-SECOND | 22,246,479 BYTES-PER-SECOND |
01,840 +-----+
01,841 The part I use is from https://github.com/google/highwayhash/tree/master/c
01,842 */
01,843 // // https://godbolt.org/ [[[
01,844 // #include <stdlib.h>
01,845 // #include <stdint.h> // uint64_t needed
01,846 // #include <string.h> // memset
01,847 // #include <smmintrin.h> // SSE4.1 intrinsics
01,848 // #include <wmmintrin.h>
01,849 // void SlowCopy128bit (const char *SOURCE, char *TARGET) { _mm_storeu_si128((__m128i *) (TARGET), _mm_loadu_si128((const __m128i *) (SOURCE))); }
01,850 // unsigned char DDAES[16];
01,851 // // https://godbolt.org/ ]]]
01,852 // Collision Benchmark - DoubleDeuceAES_128bits versus XXH3_64bits v0.8.0
01,853 //
01,854 // Testset: "A billion Knight-Tours variants (each KT with 256 variants, the KT itself omitted) - each 128 bytes long"
01,855 // Testfile: 1000000000.KnightTours.txt (130,000,000,000 bytes)
01,856 //
01,857 // The name of the game - hashing all lines and taking either 5 bytes or 6,7,8 bytes from the hash.
01,858 //
01,859 // ```
01,860 // +-----+
01,861 // | Hasher          | Collisions within first 5 bytes |
01,862 // +-----+
01,863 // | XXH3_64bits v0.8.0 | 1,000,000,000 - 999,545,727 distinct lines = 454,273 |
01,864 // +-----+
01,865 // | DoubleDeuceAES_128bits | 1,000,000,000 - 999,545,796 distinct lines = 454,204 |
01,866 // +-----+
01,867 //

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,868 // +-----+
01,869 // | Hasher          | Collisions within first 6 bytes |
01,870 // +-----+
01,871 // | XXH3_64bits v0.8.0 | 1,000,000,000 - 999,998,214 distinct lines =    1,786 |
01,872 // +-----+
01,873 // | DoubleDeuceAES_128bits | 1,000,000,000 - 999,998,213 distinct lines =    1,787 |
01,874 // +-----+
01,875 //
01,876 // +-----+
01,877 // | Hasher          | Collisions within first 7 bytes |
01,878 // +-----+
01,879 // | XXH3_64bits v0.8.0 | 1,000,000,000 - 999,999,989 distinct lines =     11 |
01,880 // +-----+
01,881 // | DoubleDeuceAES_128bits | 1,000,000,000 - 999,999,994 distinct lines =      6 |
01,882 // +-----+
01,883 //
01,884 // +-----+
01,885 // | Hasher          | Collisions within first 8 bytes |
01,886 // +-----+
01,887 // | XXH3_64bits v0.8.0 | 1,000,000,000 - 1,000,000,000 distinct lines =      0 |
01,888 // +-----+
01,889 // | DoubleDeuceAES_128bits | 1,000,000,000 - 1,000,000,000 distinct lines =      0 |
01,890 // +-----+
01,891 // ```
01,892 //
01,893 // The benchmark package (allowing to reproduce all the stuff):
01,894 // www.sanmayce.com/COLLISION\_Hashliner.zip
01,895 //
01,896 // This is how the console looks like:
01,897 //
01,898 // ```
01,899 // C:\test\COLLISION_Hashliner>GENERATE_XmillionKnight-Tours.bat 1000000000
01,900 // Generating 1000000000 Knight-Tours and dumping them into file ...
01,901 //
01,902 // C:\test\COLLISION_Hashliner>Knight-Tour_FNV1A_YoshimitsuTRIADii_vs_CRC32_TRISMUS.exe a8 1000000000 1>1000000000.KnightTours.txt
01,903 //
01,904 // C:\test\COLLISION_Hashliner>bench7.bat 1000000000.KnightTours.txt
01,905 //
01,906 // C:\test\COLLISION_Hashliner>Hashliner_XXH3_dump7byteshash.exe 1000000000.KnightTours.txt 1>1000000000.KnightTours.txt.xzh3.txt
01,907 //
01,908 // C:\test\COLLISION_Hashliner>Hashliner_DD AES_dump7byteshash.exe 1000000000.KnightTours.txt 1>1000000000.KnightTours.txt.DDAES.txt
01,909 //
01,910 // C:\test\COLLISION_Hashliner>Sandokan_QuickSortExternal_Deduplicated_4+GB_64bit_Intel.exe 1000000000.KnightTours.txt.xzh3.txt /fast /descend 3000
01,911 // Sandokan_QuickSortExternal_4+GB r.3+, written by Kaze, using Bill Durango's Quicksort source.
01,912 // Size of input file: 16,000,000,000
01,913 // Counting lines ...
01,914 // Lines encountered: 1,000,000,000
01,915 // Longest line (including CR if present): 15
01,916 // Allocated memory for pointers-to-lines in MB: 7629

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
01,917 // Assigning pointers ...
01,918 // sizeof(int), sizeof(void*): 4, 8
01,919 // Trying to allocate memory for the file itself in MB: 15258 ... OK! Get on with fast internal accesses.
01,920 // Uploading ...
01,921 // Sorting 1,000,000,000 Pointers ...
01,922 // Quicksort (Insertionsort for small blocks) commenced ...
01,923 // / RightEnd: 000,328,304,267; NumberOfSplittings: 0,114,284,204; Done: 100% ...
01,924 // NumberOfComparisons: 34,310,536,510
01,925 // The time to sort 1,000,000,000 items via Quicksort+Insertionsort was 2,848,402 clocks.
01,926 // Performance: 12,045,534 Comparisons_128B_long-Per-Second i.e 24,091,068 RandomReads_128B_long-Per-Second.
01,927 // Dumping the sorted data (Regime=2)...
01,928 // \ Done 100% ...
01,929 // Dumped 1,000,000,000 lines.
01,930 // OK! Incoming and resultant file's sizes match.
01,931 // Dumping the sorted data [deduplicated] ...
01,932 // Dumped 999,999,989 distinct lines.
01,933 // Dump time: 460,940 clocks.
01,934 // Total time: 3,347,265 clocks.
01,935 // Performance: 4,780 bytes/clock.
01,936 // Done successfully.
01,937 //
01,938 // C:\test\COLLISION_Hashliner>sort /R QuickSortExternal_4+GB.distinct.txt 1>1000000000.KnightTours.txt.xzh3.7bytes.2orABOVE.txt
01,939 //
01,940 // C:\test\COLLISION_Hashliner>Sandokan_QuickSortExternal_Deduplicated_4+GB_64bit_Intel.exe 1000000000.KnightTours.txt.DDAES.txt /fast /descend 3000
01,941 // Sandokan_QuickSortExternal_4+GB r.3+, written by Kaze, using Bill Durango's Quicksort source.
01,942 // Size of input file: 16,000,000,000
01,943 // Counting lines ...
01,944 // Lines encountered: 1,000,000,000
01,945 // Longest line (including CR if present): 15
01,946 // Allocated memory for pointers-to-lines in MB: 7629
01,947 // Assigning pointers ...
01,948 // sizeof(int), sizeof(void*): 4, 8
01,949 // Trying to allocate memory for the file itself in MB: 15258 ... OK! Get on with fast internal accesses.
01,950 // Uploading ...
01,951 // Sorting 1,000,000,000 Pointers ...
01,952 // Quicksort (Insertionsort for small blocks) commenced ...
01,953 // - RightEnd: 000,759,555,061; NumberOfSplittings: 0,114,282,509; Done: 100% ...
01,954 // NumberOfComparisons: 34,551,039,764
01,955 // The time to sort 1,000,000,000 items via Quicksort+Insertionsort was 2,896,271 clocks.
01,956 // Performance: 11,929,487 Comparisons_128B_long-Per-Second i.e 23,858,974 RandomReads_128B_long-Per-Second.
01,957 // Dumping the sorted data (Regime=2)...
01,958 // \ Done 100% ...
01,959 // Dumped 1,000,000,000 lines.
01,960 // OK! Incoming and resultant file's sizes match.
01,961 // Dumping the sorted data [deduplicated] ...
01,962 // Dumped 999,999,994 distinct lines.
01,963 // Dump time: 458,406 clocks.
01,964 // Total time: 3,393,196 clocks.
01,965 // Performance: 4,715 bytes/clock.
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
01,966 // Done successfully.
01,967 //
01,968 // C:\test\COLLISION_Hashliner>sort /R QuickSortExternal_4+GB.distinct.txt 1>1000000000.KnightTours.txt.DDAES.7bytes.2orABOVE.txt
01,969 //
01,970 // C:\test\COLLISION_Hashliner>dir *7b*
01,971 //
01,972 // 15-Aug-21 12:28          156 1000000000.KnightTours.txt.DDAES.7bytes.2orABOVE.txt
01,973 // 15-Aug-21 11:32          286 1000000000.KnightTours.txt.xxh3.7bytes.2orABOVE.txt
01,974 //
01,975 // C:\test\COLLISION_Hashliner>type 1000000000.KnightTours.txt.xxh3.7bytes.2orABOVE.txt
01,976 // 0,000,002      f84627e722e85e
01,977 // 0,000,002      f0039d0c4e4fce
01,978 // 0,000,002      c87c64d97df0e7
01,979 // 0,000,002      bb4344a5546572
01,980 // 0,000,002      af2f628f4b3ffb
01,981 // 0,000,002      a8cb8675c94610
01,982 // 0,000,002      a742cf83948622
01,983 // 0,000,002      657cb9dffa2d962
01,984 // 0,000,002      436aef7ab54ce7
01,985 // 0,000,002      270fcde0563670
01,986 // 0,000,002      0b533d70915c51
01,987 //
01,988 // C:\test\COLLISION_Hashliner>
01,989 // ``
01,990
01,991 #include <wmmintrin.h>
01,992 void SlowCopy128bit (const char *SOURCE, char *TARGET) { _mm_storeu_si128((__m128i *) (TARGET), _mm_loadu_si128((const __m128i *) (SOURCE))); }
01,993 unsigned char DDAES[16];
01,994
01,995 void DoubleDeuceAES_Gumbotron(const uint8_t *buffer, size_t length, uint64_t *LoPart, uint64_t *HiPart) {
01,996     size_t i, Cycles;
01,997     __m128i hashA = _mm_set_epi64x(0x6c62272e07bb0142, 0x62b821756295c58d); // 0x6c62272e07bb014262b821756295c58d // _mm_setzero_si128();
01,998     __m128i hashB = _mm_set_epi64x(0xdd268dbcaac55036, 0x2d98c384c4e576cc); // 0xdd268dbcaac550362d98c384c4e576cc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
01,999     __m128i hashC = _mm_set_epi64x(0xc8b1536847b6bbb3, 0x1023b4c8caee0535); // 0xdd268dbcaac550362d98c384c4e576cc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
02,000     __m128i hashD = _mm_setzero_si128();
02,001     __m128i a0,a1,a2,a3; // Instead of this chunkenization, ZMM houses the 4 XMMs, if there is shuffle across all the 512bits, use it. There is, but __m256i __m256_shuffle_epi8(__m256i a, __m256i b) is more handy.
02,002     __m128i b0,b1,b2,b3;
02,003     __m128i c0,c1,c2,c3;
02,004     __m128i d0,d1,d2,d3;
02,005     __m128i tmp0,tmp1,tmp2,tmp3;
02,006     __m128i ReverseMask = _mm_set_epi8(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15);
02,007     __m128i PartialInterleavingMask1 = _mm_set_epi8(0x80,7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0);
02,008     __m128i PartialInterleavingMask2 = _mm_set_epi8(0x80,0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8);
02,009     __m128i PartialInterleavingMask3 = _mm_set_epi8(7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0,0x80);
02,010     __m128i PartialInterleavingMask4 = _mm_set_epi8(0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8,0x80);
02,011     const __m128i *ptr128a, *ptr128b, *ptr128c, *ptr128d;
02,012
02,013     __m128i AgainstRules, GumbotronREVER, GumbotronINTER, Gumbotron, GumbotronREVERINTER;
02,014     const __m128i *ptr128;
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
02,015   _m128i InterleaveMask = _mm_set_epi8(15,7,14,6,13,5,12,4,11,3,10,2,9,1,8,0);
02,016
02,017   if (length >= 64) {
02,018       Cycles = length/64;
02,019       for(; Cycles--; buffer += 64) {
02,020           a0 = _mm_loadu_si128((__m128i *) (buffer+0*16));
02,021           a1 = _mm_loadu_si128((__m128i *) (buffer+1*16));
02,022           a2 = _mm_loadu_si128((__m128i *) (buffer+2*16));
02,023           a3 = _mm_loadu_si128((__m128i *) (buffer+3*16));
02,024           b0 = _mm_shuffle_epi8 (a3, ReverseMask);
02,025           b1 = _mm_shuffle_epi8 (a2, ReverseMask);
02,026           b2 = _mm_shuffle_epi8 (a1, ReverseMask);
02,027           b3 = _mm_shuffle_epi8 (a0, ReverseMask);
02,028           tmp0 = _mm_shuffle_epi8 (a0, PartialInterleavingMask1);
02,029           tmp1 = _mm_shuffle_epi8 (a0, PartialInterleavingMask2);
02,030           tmp2 = _mm_shuffle_epi8 (a2, PartialInterleavingMask3);
02,031           tmp3 = _mm_shuffle_epi8 (a2, PartialInterleavingMask4);
02,032           c0 = _mm_or_si128 (tmp0, tmp2);
02,033           c1 = _mm_or_si128 (tmp1, tmp3);
02,034           tmp0 = _mm_shuffle_epi8 (a1, PartialInterleavingMask1);
02,035           tmp1 = _mm_shuffle_epi8 (a1, PartialInterleavingMask2);
02,036           tmp2 = _mm_shuffle_epi8 (a3, PartialInterleavingMask3);
02,037           tmp3 = _mm_shuffle_epi8 (a3, PartialInterleavingMask4);
02,038           c2 = _mm_or_si128 (tmp0, tmp2);
02,039           c3 = _mm_or_si128 (tmp1, tmp3);
02,040           tmp0 = _mm_shuffle_epi8 (b0, PartialInterleavingMask1);
02,041           tmp1 = _mm_shuffle_epi8 (b0, PartialInterleavingMask2);
02,042           tmp2 = _mm_shuffle_epi8 (b2, PartialInterleavingMask3);
02,043           tmp3 = _mm_shuffle_epi8 (b2, PartialInterleavingMask4);
02,044           d0 = _mm_or_si128 (tmp0, tmp2);
02,045           d1 = _mm_or_si128 (tmp1, tmp3);
02,046           tmp0 = _mm_shuffle_epi8 (b1, PartialInterleavingMask1);
02,047           tmp1 = _mm_shuffle_epi8 (b1, PartialInterleavingMask2);
02,048           tmp2 = _mm_shuffle_epi8 (b3, PartialInterleavingMask3);
02,049           tmp3 = _mm_shuffle_epi8 (b3, PartialInterleavingMask4);
02,050           d2 = _mm_or_si128 (tmp0, tmp2);
02,051           d3 = _mm_or_si128 (tmp1, tmp3);
02,052
02,053           hashA = _mm_aesenc_si128(hashA, a0);
02,054           hashB = _mm_aesenc_si128(hashB, b0);
02,055           hashC = _mm_aesenc_si128(hashC, c0);
02,056           hashD = _mm_aesenc_si128(hashD, d0);
02,057
02,058           hashA = _mm_aesenc_si128(hashA, a1);
02,059           hashB = _mm_aesenc_si128(hashB, b1);
02,060           hashC = _mm_aesenc_si128(hashC, c1);
02,061           hashD = _mm_aesenc_si128(hashD, d1);
02,062
02,063           hashA = _mm_aesenc_si128(hashA, a2);
```



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,064         hashB = _mm_aesenc_si128(hashB, b2);
02,065         hashC = _mm_aesenc_si128(hashC, c2);
02,066         hashD = _mm_aesenc_si128(hashD, d2);
02,067
02,068         hashA = _mm_aesenc_si128(hashA, a3);
02,069         hashB = _mm_aesenc_si128(hashB, b3);
02,070         hashC = _mm_aesenc_si128(hashC, c3);
02,071         hashD = _mm_aesenc_si128(hashD, d3);
02,072
02,073         hashA = _mm_aesenc_si128(hashA, hashB);
02,074         hashA = _mm_aesenc_si128(hashA, hashC);
02,075         hashA = _mm_aesenc_si128(hashA, hashD);
02,076         length = length - 64;
02,077     }
02,078 }
02,079
02,080 ptr128 = (_m128i *)buffer;
02,081 if (length >= 16) {
02,082     Cycles = length/16;
02,083     for(; Cycles--; buffer += 16) {
02,084         AgainstRules = _mm_loadu_si128(ptr128++);
02,085         GumbotronREVER = _mm_shuffle_epi8 (AgainstRules, ReverseMask);
02,086         GumbotronINTER = _mm_shuffle_epi8 (AgainstRules, InterleaveMask);
02,087         GumbotronREVERINTER = _mm_shuffle_epi8 (GumbotronREVER, InterleaveMask);
02,088         hashA = _mm_aesenc_si128(hashA, AgainstRules);
02,089         hashB = _mm_aesenc_si128(hashB, GumbotronREVER);
02,090         hashC = _mm_aesenc_si128(hashC, GumbotronINTER);
02,091         hashD = _mm_aesenc_si128(hashD, GumbotronREVERINTER);
02,092         hashA = _mm_aesenc_si128(hashA, hashB);
02,093         hashA = _mm_aesenc_si128(hashA, hashC);
02,094         hashA = _mm_aesenc_si128(hashA, hashD);
02,095         length = length - 16;
02,096     }
02,097 }
02,098 // Inhere using Pippip's approach to read past the end ("the dirty" sentinel like style, or more like padding):
02,099 if (length & (16-1)) {
02,100     AgainstRules = _mm_loadu_si128(ptr128);
02,101     //AgainstRules = _mm_srli_si128 (AgainstRules, 16-length); // catastrophic error: Intrinsic parameter must be an immediate value
02,102     AgainstRules = _mm_and_si128 (AgainstRules, Gumbotron[length]);
02,103     //Gumbotron = _mm_slli_si128 (Gumbotron, 16-length); // catastrophic error: Intrinsic parameter must be an immediate value
02,104     Gumbotron = _mm_and_si128 (hashB, Gumbotron[length]);
02,105     AgainstRules = _mm_or_si128 (AgainstRules, Gumbotron);
02,106
02,107     GumbotronREVER = _mm_shuffle_epi8 (AgainstRules, ReverseMask);
02,108     GumbotronINTER = _mm_shuffle_epi8 (AgainstRules, InterleaveMask);
02,109     GumbotronREVERINTER = _mm_shuffle_epi8 (GumbotronREVER, InterleaveMask);
02,110     hashA = _mm_aesenc_si128(hashA, AgainstRules);
02,111     hashB = _mm_aesenc_si128(hashB, GumbotronREVER);
02,112     hashC = _mm_aesenc_si128(hashC, GumbotronINTER);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,113         hashD = _mm_aesenc_si128(hashD, GumbottronREVERINTER);
02,114         hashA = _mm_aesenc_si128(hashA, hashB);
02,115         hashA = _mm_aesenc_si128(hashA, hashC);
02,116         hashA = _mm_aesenc_si128(hashA, hashD);
02,117     }
02,118     //SlowCopy128bit( (const char *)(&hashA), (char *)&DDAES[0]);
02,119     memcpy ( LoPart, (const char *)(&hashA), 8);
02,120     memcpy ( HiPart, (const char *)(&hashA)+8, 8);
02,121 }
02,122 /*
02,123 ; mark_description "Intel(R) C++ Compiler XE for applications running on Intel(R) 64, Version 15.0.0.108 Build 20140726";
02,124 ; mark_description "-O3 -arch:avx -FA -D_WIN32_ENVIRONMENT_ -D_N_DDAES";
02,125
02,126 DoubleDeuceAES_GumbottronPROC
02,127     sub     rsp, 216
02,128     mov     r8, rdx
02,129     vmovups xmm3, XMMWORD PTR [_2il0floatpacket.0]
02,130     lea     rdx, QWORD PTR [_ImageBase]
02,131     vmovups xmm4, XMMWORD PTR [_2il0floatpacket.1]
02,132     vpxor   xmm2, xmm2, xmm2
02,133     vmovups xmm5, XMMWORD PTR [_2il0floatpacket.2]
02,134     vmovdqu xmm1, XMMWORD PTR [_2il0floatpacket.4]
02,135     vmovdqu xmm0, XMMWORD PTR [_2il0floatpacket.5]
02,136     cmp     r8, 64
02,137     jbe     .B6.6
02,138     mov     rax, r8
02,139     shr     rax, 6
02,140     dec     rax
02,141     cmp     rax, -1
02,142     je     .B6.6
02,143     vmovups XMMWORD PTR [64+rsp], xmm6
02,144     vmovups XMMWORD PTR [48+rsp], xmm7
02,145     vmovups XMMWORD PTR [32+rsp], xmm8
02,146     vmovups XMMWORD PTR [80+rsp], xmm9
02,147     vmovups XMMWORD PTR [96+rsp], xmm10
02,148     vmovups XMMWORD PTR [112+rsp], xmm11
02,149     vmovups XMMWORD PTR [128+rsp], xmm12
02,150     vmovups XMMWORD PTR [144+rsp], xmm13
02,151     vmovups XMMWORD PTR [160+rsp], xmm14
02,152     vmovups XMMWORD PTR [176+rsp], xmm15
02,153 .B6.4:
02,154     vmovdqu xmm10, XMMWORD PTR [48+rcx]
02,155     vmovdqu xmm6, XMMWORD PTR [32+rcx]
02,156     vmovdqu xmm11, XMMWORD PTR [16+rcx]
02,157     vmovdqu xmm12, XMMWORD PTR [rcx]
02,158     vmovdqu xmm14, XMMWORD PTR [_2il0floatpacket.3]
02,159     dec     rax
02,160     vpshufb xmm8, xmm10, xmm14
02,161     vpshufb xmm13, xmm6, xmm14

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,162      vpshufb   xmm9, xmm11, xmm14
02,163      vpshufb   xmm14, xmm12, xmm14
02,164      vpshufb   xmm15, xmm12, xmm1
02,165      vaesenc   xmm4, xmm4, xmm8
02,166      add      r8, -64
02,167      vaesenc   xmm7, xmm4, xmm13
02,168      vaesenc   xmm4, xmm7, xmm9
02,169      vaesenc   xmm7, xmm4, xmm14
02,170      vmovdqu   xmm4, XMMWORD PTR [_2il0floatpacket.6]
02,171      vmovups   XMMWORD PTR [192+rsp], xmm7
02,172      vpshufb   xmm7, xmm6, xmm4
02,173      vpor      xmm15, xmm15, xmm7
02,174      vmovdqu   xmm7, XMMWORD PTR [_2il0floatpacket.7]
02,175      vaesenc   xmm15, xmm5, xmm15
02,176      vpshufb   xmm5, xmm12, xmm0
02,177      vpshufb   xmm6, xmm6, xmm7
02,178      vpor      xmm5, xmm5, xmm6
02,179      vaesenc   xmm15, xmm15, xmm5
02,180      vpshufb   xmm6, xmm11, xmm1
02,181      vpshufb   xmm5, xmm10, xmm4
02,182      vpor      xmm6, xmm6, xmm5
02,183      vaesenc   xmm5, xmm15, xmm6
02,184      vpshufb   xmm15, xmm11, xmm0
02,185      vpshufb   xmm6, xmm10, xmm7
02,186      vpor      xmm15, xmm15, xmm6
02,187      vaesenc   xmm5, xmm5, xmm15
02,188      vpshufb   xmm6, xmm8, xmm1
02,189      vpshufb   xmm15, xmm9, xmm4
02,190      vpshufb   xmm8, xmm8, xmm0
02,191      vpshufb   xmm9, xmm9, xmm7
02,192      vpshufb   xmm4, xmm14, xmm4
02,193      vpor      xmm6, xmm6, xmm15
02,194      vaesenc   xmm2, xmm2, xmm6
02,195      vpor      xmm6, xmm8, xmm9
02,196      vaesenc   xmm6, xmm2, xmm6
02,197      vpshufb   xmm2, xmm13, xmm1
02,198      vpshufb   xmm13, xmm13, xmm0
02,199      vpor      xmm2, xmm2, xmm4
02,200      vaesenc   xmm8, xmm6, xmm2
02,201      vpshufb   xmm2, xmm14, xmm7
02,202      vaesenc   xmm3, xmm3, xmm12
02,203      vpor      xmm4, xmm13, xmm2
02,204      vaesenc   xmm2, xmm8, xmm4
02,205      vaesenc   xmm4, xmm3, xmm11
02,206      vaesenc   xmm3, xmm4, XMMWORD PTR [32+rcx]
02,207      add      rcx, 64
02,208      vaesenc   xmm6, xmm3, xmm10
02,209      vmovups   xmm4, XMMWORD PTR [192+rsp]
02,210      vaesenc   xmm7, xmm6, xmm4

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,211      vaesenc  xmm8, xmm7, xmm5
02,212      vaesenc  xmm3, xmm8, xmm2
02,213      cmp      rax, -1
02,214      jne      .B6.4
02,215      vmovups  xmm6, XMMWORD PTR [64+rsp]
02,216      vmovups  xmm7, XMMWORD PTR [48+rsp]
02,217      vmovups  xmm8, XMMWORD PTR [32+rsp]
02,218      vmovups  xmm9, XMMWORD PTR [80+rsp]
02,219      vmovups  xmm10, XMMWORD PTR [96+rsp]
02,220      vmovups  xmm11, XMMWORD PTR [112+rsp]
02,221      vmovups  xmm12, XMMWORD PTR [128+rsp]
02,222      vmovups  xmm13, XMMWORD PTR [144+rsp]
02,223      vmovups  xmm14, XMMWORD PTR [160+rsp]
02,224      vmovups  xmm15, XMMWORD PTR [176+rsp]
02,225      .B6.6::
02,226      cmp      r8, 16
02,227      jb      .B6.11
02,228      mov      rax, r8
02,229      shr      rax, 4
02,230      dec      rax
02,231      cmp      rax, -1
02,232      je      .B6.11
02,233      vmovups  XMMWORD PTR [64+rsp], xmm6
02,234      vmovups  XMMWORD PTR [48+rsp], xmm7
02,235      vmovups  XMMWORD PTR [32+rsp], xmm8
02,236      vmovdqu  xmm0, XMMWORD PTR [_2i10floatpacket.8]
02,237      vmovdqu  xmm6, XMMWORD PTR [_2i10floatpacket.3]
02,238      ALIGN    16
02,239      .B6.9::
02,240      vmovdqu  xmm1, XMMWORD PTR [rcx]
02,241      vpshufb  xmm8, xmm1, xmm6
02,242      vpshufb  xmm7, xmm1, xmm0
02,243      vaesenc  xmm5, xmm5, xmm7
02,244      dec      rax
02,245      vpshufb  xmm7, xmm8, xmm0
02,246      vaesenc  xmm4, xmm4, xmm8
02,247      add      rcx, 16
02,248      vaesenc  xmm3, xmm3, xmm1
02,249      add      r8, -16
02,250      vaesenc  xmm1, xmm3, xmm4
02,251      vaesenc  xmm2, xmm2, xmm7
02,252      vaesenc  xmm3, xmm1, xmm5
02,253      vaesenc  xmm3, xmm3, xmm2
02,254      cmp      rax, -1
02,255      jne      .B6.9
02,256      vmovups  xmm6, XMMWORD PTR [64+rsp]
02,257      vmovups  xmm7, XMMWORD PTR [48+rsp]
02,258      vmovups  xmm8, XMMWORD PTR [32+rsp]
02,259      .B6.11::

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,260      test      r8, 15
02,261      je       .B6.13
02,262      shl      r8, 4
02,263      vmovdqu  xmm1, XMMWORD PTR [rcx]
02,264      vpand     xmm0, xmm1, XMMWORD PTR [imagerel(VectorsNeedNonVariable1)+rdx+r8]
02,265      vpand     xmm1, xmm4, XMMWORD PTR [imagerel(VectorsNeedNonVariable2)+rdx+r8]
02,266      vpor      xmm0, xmm0, xmm1
02,267      vpshufb   xmm1, xmm0, XMMWORD PTR [_2il0floatpacket.3]
02,268      vaesenc   xmm3, xmm3, xmm0
02,269      vaesenc   xmm4, xmm4, xmm1
02,270      vaesenc   xmm4, xmm3, xmm4
02,271      vmovdqu  xmm3, XMMWORD PTR [_2il0floatpacket.8]
02,272      vpshufb   xmm0, xmm0, xmm3
02,273      vpshufb   xmm1, xmm1, xmm3
02,274      vaesenc   xmm5, xmm5, xmm0
02,275      vaesenc   xmm0, xmm4, xmm5
02,276      vaesenc   xmm2, xmm2, xmm1
02,277      vaesenc   xmm3, xmm0, xmm2
02,278 .B6.13::
02,279      vmovups   XMMWORD PTR [DDAES], xmm3
02,280      add       rsp, 216
02,281      ret
02,282      ALIGN    16
02,283 DoubleDeuceAES_Gumbotron ENDP
02,284 */
02,285
02,286 // Revision 2021-Aug-22 [
02,287 //VPSHUFb: _m256i _mm256_shuffle_epi8(_m256i a, _m256i b)
02,288 //VPSHUFb _m512i _mm512_shuffle_epi8(_m512i a, _m512i b);
02,289 void DoubleDeuceAES_Gumbotron_YMM(const uint8_t *buffer, size_t length, uint64_t *LoPart, uint64_t *HiPart) {
02,290     size_t i, Cycles;
02,291     _m128i hashA = _mm_set_epi64x(0x6c62272e07bb0142, 0x62b821756295c58d); // 0x6c62272e07bb014262b821756295c58d // _mm_setzero_si128();
02,292     _m128i hashB = _mm_set_epi64x(0xdd268dbcaac55036, 0x2d98c384c4e576cc); // 0xdd268dbcaac550362d98c384c4e576ccc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
02,293     _m128i hashC = _mm_set_epi64x(0xc8b1536847b6bbb3, 0x1023b4c8caee0535); // 0xdd268dbcaac550362d98c384c4e576ccc8b1536847b6bbb31023b4c8caee0535 // FNV offset basis
02,294     _m128i hashD = _mm_setzero_si128();
02,295     _m128i a0,a1,a2,a3; // Instead of this chunkenization, YMM houses the 4 XMMs, if there is shuffle across all the 512bits, use it. There is, but _m256i _mm256_shuffle_epi8(_m256i a, _m256i b) is more handy.
02,296     _m256i a0YMM,a2YMM;
02,297     _m128i b0,b1,b2,b3;
02,298     _m256i b0YMM,b2YMM;
02,299     _m128i c0,c1,c2,c3;
02,300     _m256i c0YMM,c2YMM;
02,301     _m128i d0,d1,d2,d3;
02,302     _m256i d0YMM,d2YMM;
02,303     _m128i tmp0,tmp1,tmp2,tmp3;
02,304     _m256i tmp0YMM,tmp2YMM;
02,305     _m128i ReverseMask = _mm_set_epi8(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15);
02,306     _m256i ReverseMaskYMM = _mm256_set_epi8(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31);
02,307     _m128i PartialInterleavingMask1 = _mm_set_epi8(0x80,7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0);
02,308     _m128i PartialInterleavingMask2 = _mm_set_epi8(0x80,0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,309 __m128i PartialInterleavingMask3 = _mm_set_epi8(7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0,0x80);
02,310 __m128i PartialInterleavingMask4 = _mm_set_epi8(0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8,0x80);
02,311 __m256i PartialInterleavingMask1YMM = _mm256_set_epi8(0x80,0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8,0x80,7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0);
02,312 // __m256i PartialInterleavingMask2YMM =
__mm256_set_epi8(0x80,0xf+16,0x80,0xe+16,0x80,0xd+16,0x80,0xc+16,0x80,0xb+16,0x80,0xa+16,0x80,9+16,0x80,8+16,0x80,7+16,0x80,6+16,0x80,5+16,0x80,4+16,0x80,3+16,0x80,2+16,0x80,1+16,0x80,0+16);
02,313 __m256i PartialInterleavingMask3YMM = _mm256_set_epi8(0xf,0x80,0xe,0x80,0xd,0x80,0xc,0x80,0xb,0x80,0xa,0x80,9,0x80,8,0x80,7,0x80,6,0x80,5,0x80,4,0x80,3,0x80,2,0x80,1,0x80,0,0x80);
02,314 // __m256i PartialInterleavingMask4YMM =
__mm256_set_epi8(0xf+16,0x80,0xe+16,0x80,0xd+16,0x80,0xc+16,0x80,0xb+16,0x80,0xa+16,0x80,9+16,0x80,8+16,0x80,7+16,0x80,6+16,0x80,5+16,0x80,4+16,0x80,3+16,0x80,2+16,0x80,1+16,0x80,0+16,0x80);
02,315 const __m128i *ptr128a, *ptr128b, *ptr128c, *ptr128d;
02,316
02,317 __m128i AgainstRules, GumbotronREVER, GumbotronINTER, Gumbotron, GumbotronREVERINTER;
02,318 const __m128i *ptr128;
02,319 __m128i InterleaveMask = _mm_set_epi8(15,7,14,6,13,5,12,4,11,3,10,2,9,1,8,0);
02,320 uint8_t vector[32];
02,321
02,322 if (length >= 64) {
02,323     Cycles = length/64;
02,324     for(; Cycles--; buffer += 64) {
02,325         //a0 = _mm_loadu_si128((__m128i *) (buffer+0*16));
02,326         //a1 = _mm_loadu_si128((__m128i *) (buffer+1*16));
02,327         //a2 = _mm_loadu_si128((__m128i *) (buffer+2*16));
02,328         //a3 = _mm_loadu_si128((__m128i *) (buffer+3*16));
02,329         a0YMM = _mm256_loadu_si256((__m256i *) (buffer+0*16));
02,330         a2YMM = _mm256_loadu_si256((__m256i *) (buffer+2*16));
02,331         //b0 = _mm_shuffle_epi8 (a3, ReverseMask);
02,332         //b1 = _mm_shuffle_epi8 (a2, ReverseMask);
02,333         //b2 = _mm_shuffle_epi8 (a1, ReverseMask);
02,334         //b3 = _mm_shuffle_epi8 (a0, ReverseMask);
02,335         b0YMM = _mm256_shuffle_epi8 (a2YMM, ReverseMaskYMM); // Caution: the stupid intrinsic works on 128bit not on 256bit! b0YMM = b1+b0 not b0+b1
02,336         b2YMM = _mm256_shuffle_epi8 (a0YMM, ReverseMaskYMM);
02,337 // Should swap:
02,338 // https://godbolt.org/z/dY74zv1Ph
02,339 b0YMM = _mm256_permute4x64_epi64(b0YMM, 0b01001110); // # ymm0 = ymm0[2,3,0,1]
02,340 b2YMM = _mm256_permute4x64_epi64(b2YMM, 0b01001110); // # ymm0 = ymm0[2,3,0,1]
02,341
02,342 /*
02,343 __m256i __mm256_permute4x64_epi64 (__m256i a, const int imm8)
02,344 #include <immintrin.h>
02,345 Instruction: vpermq ymm, ymm, imm8
02,346 CPUID Flags: AVX2
02,347 Description
02,348 Shuffle 64-bit integers in a across lanes using the control in imm8, and store the results in dst.
02,349 Operation
02,350 #define SELECT4(src, control) {
02,351     CASE(control[1:0]) OF
02,352     0:     tmp[63:0] := src[63:0]
02,353     1:     tmp[63:0] := src[127:64]
02,354     2:     tmp[63:0] := src[191:128]
02,355     3:     tmp[63:0] := src[255:192]

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,356     ESAC
02,357     RETURN tmp[63:0]
02,358 }
02,359 dst[63:0] := SELECT4(a[255:0], imm8[1:0])
02,360 dst[127:64] := SELECT4(a[255:0], imm8[3:2])
02,361 dst[191:128] := SELECT4(a[255:0], imm8[5:4])
02,362 dst[255:192] := SELECT4(a[255:0], imm8[7:6])
02,363 dst[MAX:256] := 0
02,364 */
02,365
02,366 //a8C7E8G7H5G3H1F2H3G1E2C1A2B4A6B8D7F8H7G5F7H8G6H4G2E1C2A1B3A5B7D8C6A7C8E7G8H6G4H2F1D2B1A3B5D6F5D4F3E5C4B2D3F4E6C5A4B6D5F6E4C3D1E3
02,367 //                                a0]                                a2]                                a0]                                a2]
02,368 //a8C7E8G7H5G3H1F2H3G1E2C1A2B4A6B8 D7F8H7G5F7H8G6H4G2E1C2A1B3A5B7D8 C6A7C8E7G8H6G4H2F1D2B1A3B5D6F5D4 F3E5C4B2D3F4E6C5A4B6D5F6E4C3D1E3
02,369
02,370 //a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32 | 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,371 //a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34 | 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,372 //b0: 34 48 36 47 | 38 48 37 46 | 35 47 37 48 | 38 46 37 44 | 38 44 37 42 | 35 41 33 42 | 31 41 32 43 | 31 45 32 47
02,373 //b2: 32 46 31 48 | 33 47 35 48 | 37 47 38 45 | 37 43 38 61 | 38 42 36 41 | 34 42 32 41 | 31 43 32 45 | 31 47 33 48
02,374
02,375         //tmp0 = _mm_shuffle_epi8 (a0, PartialInterleavingMask1);
02,376         //tmp1 = _mm_shuffle_epi8 (a0, PartialInterleavingMask2);
02,377         //tmp2 = _mm_shuffle_epi8 (a2, PartialInterleavingMask3);
02,378         //tmp3 = _mm_shuffle_epi8 (a2, PartialInterleavingMask4);
02,379         //c0 = _mm_or_si128 (tmp0, tmp2);
02,380         //c1 = _mm_or_si128 (tmp1, tmp3);
02,381         //tmp0 = _mm_shuffle_epi8 (a1, PartialInterleavingMask1);
02,382         //tmp1 = _mm_shuffle_epi8 (a1, PartialInterleavingMask2);
02,383         //tmp2 = _mm_shuffle_epi8 (a3, PartialInterleavingMask3);
02,384         //tmp3 = _mm_shuffle_epi8 (a3, PartialInterleavingMask4);
02,385         //c2 = _mm_or_si128 (tmp0, tmp2);
02,386         //c3 = _mm_or_si128 (tmp1, tmp3);
02,387 // c0: 00 20 01 21 | 02 22 03 23 | 04 24 05 25 | 06 26 07 27
02,388 // c1: 08 28 09 29 | 0a 2a 0b 2b | 0c 2c 0d 2d | 0e 2e 0f 2f
02,389 // c2: 10 30 11 31 | 12 32 13 33 | 14 34 15 35 | 16 36 17 37
02,390 // c3: 18 38 19 39 | 1a 3a 1b 3b | 1c 3c 1d 3d | 1e 3e 1f 3f
02,391 /*
02,392 __m256i _mm256_unpacklo_epi8 (__m256i a, __m256i b)
02,393 Synopsis
02,394 __m256i _mm256_unpacklo_epi8 (__m256i a, __m256i b)
02,395 #include <immintrin.h>
02,396 Instruction: vpunpcklbw ymm, ymm, ymm
02,397 CPUID Flags: AVX2
02,398 Description
02,399 Unpack and interleave 8-bit integers from the low half of each 128-bit lane in a and b, and store the results in dst.
02,400 Operation
02,401 DEFINE INTERLEAVE_BYTES(src1[127:0], src2[127:0]) {
02,402     dst[7:0] := src1[7:0]
02,403     dst[15:8] := src2[7:0]
02,404     dst[23:16] := src1[15:8]

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,405 dst[31:24] := src2[15:8]
02,406 dst[39:32] := src1[23:16]
02,407 dst[47:40] := src2[23:16]
02,408 dst[55:48] := src1[31:24]
02,409 dst[63:56] := src2[31:24]
02,410 dst[71:64] := src1[39:32]
02,411 dst[79:72] := src2[39:32]
02,412 dst[87:80] := src1[47:40]
02,413 dst[95:88] := src2[47:40]
02,414 dst[103:96] := src1[55:48]
02,415 dst[111:104] := src2[55:48]
02,416 dst[119:112] := src1[63:56]
02,417 dst[127:120] := src2[63:56]
02,418 RETURN dst[127:0]
02,419 }
02,420 dst[127:0] := INTERLEAVE_BYTES(a[127:0], b[127:0])
02,421 dst[255:128] := INTERLEAVE_BYTES(a[255:128], b[255:128])
02,422 dst[MAX:256] := 0
02,423 */
02,424 //_m128i _mm_unpacklo_epi8 (_m128i a, __m128i b)
02,425
02,426 c0YMM = _mm256_unpacklo_epi8 (a0YMM, a2YMM);
02,427 c2YMM = _mm256_unpackhi_epi8 (a0YMM, a2YMM);
02,428 // Above two lines gave:
02,429 /*
02,430 [ 0] [ 1] [ 2] [ 3]
02,431 a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32 | 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,432 [ 4] [ 5] [ 6] [ 7]
02,433 a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34 | 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,434
02,435 [ 0+4] [ 2+6]
02,436 c0: 61 44 38 37 | 43 46 37 38 | 45 48 38 37 | 47 47 37 35 | 48 47 33 32 | 47 45 31 31 | 45 43 32 32 | 43 41 31 31
02,437 [ 1+5] [ 3+7]
02,438 c2: 48 46 35 37 | 47 48 33 38 | 48 47 31 36 | 46 48 32 34 | 41 42 32 33 | 42 41 34 35 | 41 42 36 37 | 42 44 38 38
02,439 */
02,440 // But I need:
02,441 // c0,c1,c2,c3 not c0,c2,c1,c3
02,442 // 0+4,1+5,2+6,3+7 not 0+4,2+6,1+5,3+7
02,443 // as in XMM:
02,444 // c0: 00 20 01 21 | 02 22 03 23 | 04 24 05 25 | 06 26 07 27
02,445 // c1: 08 28 09 29 | 0a 2a 0b 2b | 0c 2c 0d 2d | 0e 2e 0f 2f
02,446 // c2: 10 30 11 31 | 12 32 13 33 | 14 34 15 35 | 16 36 17 37
02,447 // c3: 18 38 19 39 | 1a 3a 1b 3b | 1c 3c 1d 3d | 1e 3e 1f 3f
02,448
02,449 //a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32 | 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,450 //a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34 | 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,451
02,452 //tmp0YMM: 61 00 38 00 | 43 00 37 00 | 45 00 38 00 | 47 00 37 00 | 41 00 32 00 | 42 00 34 00 | 41 00 36 00 | 42 00 38 00
02,453 //tmp2YMM: 00 44 00 37 | 00 46 00 38 | 00 48 00 37 | 00 47 00 35 | 00 42 00 33 | 00 41 00 35 | 00 42 00 37 | 00 44 00 38

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

02,454
02,455 //      tmp0YMM = _mm256_shuffle_epi8 (a0YMM, PartialInterleavingMask1YMM);
02,456 //      tmp2YMM = _mm256_shuffle_epi8 (a2YMM, PartialInterleavingMask3YMM);
02,457 //      c0YMM = _mm256_or_si256 (tmp0YMM, tmp2YMM);
02,458 //      tmp0YMM = _mm256_shuffle_epi8 (a0YMM, PartialInterleavingMask1YMM);
02,459 //      tmp2YMM = _mm256_shuffle_epi8 (a2YMM, PartialInterleavingMask3YMM);
02,460 //      c2YMM = _mm256_or_si256 (tmp0YMM, tmp2YMM);
02,461
02,462 //tmp0 = _mm_shuffle_epi8 (b0, PartialInterleavingMask1);
02,463 //tmp1 = _mm_shuffle_epi8 (b0, PartialInterleavingMask2);
02,464 //tmp2 = _mm_shuffle_epi8 (b2, PartialInterleavingMask3);
02,465 //tmp3 = _mm_shuffle_epi8 (b2, PartialInterleavingMask4);
02,466 //d0 = _mm_or_si128 (tmp0, tmp2);
02,467 //d1 = _mm_or_si128 (tmp1, tmp3);
02,468 //tmp0 = _mm_shuffle_epi8 (b1, PartialInterleavingMask1);
02,469 //tmp1 = _mm_shuffle_epi8 (b1, PartialInterleavingMask2);
02,470 //tmp2 = _mm_shuffle_epi8 (b3, PartialInterleavingMask3);
02,471 //tmp3 = _mm_shuffle_epi8 (b3, PartialInterleavingMask4);
02,472 //d2 = _mm_or_si128 (tmp0, tmp2);
02,473 //d3 = _mm_or_si128 (tmp1, tmp3);
02,474 // d0: 3f 1f 3e 1e | 3d 1d 3c 1c | 3b 1b 3a 1a | 39 19 38 18
02,475 // d1: 37 17 36 16 | 35 15 34 14 | 33 13 32 12 | 31 11 30 10
02,476 // d2: 2f 0f 2e 0e | 2d 0d 2c 0c | 2b 0b 2a 0a | 29 09 28 08
02,477 // d3: 27 07 26 06 | 25 05 24 04 | 23 03 22 02 | 21 01 20 00
02,478 // [[[ Next 6 lines are identical to simply REVERSE C vector - which is in 2 lines
02,479 // *
02,480 tmp0YMM = _mm256_shuffle_epi8 (b0YMM, PartialInterleavingMask1YMM);
02,481 tmp2YMM = _mm256_shuffle_epi8 (b2YMM, PartialInterleavingMask3YMM);
02,482 d0YMM = _mm256_or_si256 (tmp0YMM, tmp2YMM);
02,483 tmp0YMM = _mm256_shuffle_epi8 (b0YMM, PartialInterleavingMask2YMM);
02,484 tmp2YMM = _mm256_shuffle_epi8 (b2YMM, PartialInterleavingMask4YMM);
02,485 d2YMM = _mm256_or_si256 (tmp0YMM, tmp2YMM);
02,486 // *
02,487 // ]]] Next 6 lines are identical to simply REVERSE C vector - which is in 2 lines, but I don't have it, so {{{
02,488 d0YMM = _mm256_unpacklo_epi8 (b0YMM, b2YMM);
02,489 d2YMM = _mm256_unpackhi_epi8 (b0YMM, b2YMM);
02,490 // }}}
02,491
02,492 // For above code (the last thing to fix: b should be REVERSED a, not b1,b0,b3,b2):
02,493 //a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32 ! 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,494 //a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34 ! 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,495 //
02,496 //b0: 34 48 36 47 | 38 48 37 46 | 35 47 37 48 | 38 46 37 44 ! 38 44 37 42 | 35 41 33 42 | 31 41 32 43 | 31 45 32 47
02,497 //b2: 32 46 31 48 | 33 47 35 48 | 37 47 38 45 | 37 43 38 61 ! 38 42 36 41 | 34 42 32 41 | 31 43 32 45 | 31 47 33 48
02,498 //
02,499 //c0: 61 44 38 37 | 43 46 37 38 | 45 48 38 37 | 47 47 37 35 ! 48 47 33 32 | 47 45 31 31 | 45 43 32 32 | 43 41 31 31
02,500 //c2: 48 46 35 37 | 47 48 33 38 | 48 47 31 36 | 46 48 32 34 ! 41 42 32 33 | 42 41 34 35 | 41 42 36 37 | 42 44 38 38
02,501 //
02,502 //d0: 34 32 48 46 | 36 31 47 48 | 38 33 48 47 | 37 35 46 48 ! 38 38 44 42 | 37 36 42 41 | 35 34 41 42 | 33 32 42 41

```

02,503 //d2: 35 37 47 47 | 37 38 48 45 | 38 37 46 43 | 37 38 44 61 ! 31 31 41 43 | 32 32 43 45 | 31 31 45 47 | 32 33 47 48

02,504

02,505 /\*

02,506 \_mm256\_storeu\_si256((\_\_m256i\*)vector, a0YMM);

02,507 printf("a0: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,508 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,509 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,510 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,511 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,512

02,513 \_mm256\_storeu\_si256((\_\_m256i\*)vector, a2YMM);

02,514 printf("a2: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,515 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,516 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,517 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,518 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,519

02,520 \_mm256\_storeu\_si256((\_\_m256i\*)vector, b0YMM);

02,521 printf("b0: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,522 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,523 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,524 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,525 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,526

02,527 \_mm256\_storeu\_si256((\_\_m256i\*)vector, b2YMM);

02,528 printf("b2: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,529 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,530 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,531 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,532 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,533

02,534 \_mm256\_storeu\_si256((\_\_m256i\*)vector, c0YMM);

02,535 printf("c0: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,536 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,537 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,538 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,539 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,540 \_mm256\_storeu\_si256((\_\_m256i\*)vector, c2YMM);

02,541 printf("c2: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,542 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,543 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,544 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

02,545 vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);

02,546

02,547 \_mm256\_storeu\_si256((\_\_m256i\*)vector, d0YMM);

02,548 printf("d0: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x ! %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",

02,549 vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],

02,550 vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],

02,551 vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,552     vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);
02,553     _mm256_storeu_si256((__m256i*)vector, dZYMM);
02,554     printf("d2: %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x\n",
02,555     vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],
02,556     vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15],
02,557     vector[0+16], vector[1+16], vector[2+16], vector[3+16], vector[4+16], vector[5+16], vector[6+16], vector[7+16],
02,558     vector[8+16], vector[9+16], vector[10+16], vector[11+16], vector[12+16], vector[13+16], vector[14+16], vector[15+16]);
02,559
02,560     printf("\n");
02,561 */
02,562
02,563 /*
02,564 G:\Lookupperorama_r13\COLLISION_Hashliner>Hashliner_DD AES_dump5byteshash.exe 1.KnightTours.txt
02,565 394e907d43
02,566
02,567 G:\Lookupperorama_r13\COLLISION_Hashliner>Hashliner_DD AES-XMM_dump5byteshash.exe 1.KnightTours.txt
02,568 f4b027e3ab
02,569
02,570 G:\Lookupperorama_r13\COLLISION_Hashliner>d:
02,571
02,572 D:\>g:
02,573
02,574 G:\Lookupperorama_r13\COLLISION_Hashliner>Hashliner_DD AES_dump5byteshash.exe 1.KnightTours.txt
02,575 a0: 61 38 43 37 ! 45 38 47 37 ! 48 35 47 33 ! 48 31 46 32 ! 48 33 47 31 ! 45 32 43 31 ! 41 32 42 34 ! 41 36 42 38
02,576 a2: 44 37 46 38 ! 48 37 47 35 ! 46 37 48 38 ! 47 36 48 34 ! 47 32 45 31 ! 43 32 41 31 ! 42 33 41 35 ! 42 37 44 38
02,577 c0: 61 44 38 37 ! 43 46 37 38 ! 45 48 38 37 ! 47 47 37 35 ! 41 42 32 33 ! 42 41 34 35 ! 41 42 36 37 ! 42 44 38 38
02,578 c2: 61 44 38 37 ! 43 46 37 38 ! 45 48 38 37 ! 47 47 37 35 ! 41 42 32 33 ! 42 41 34 35 ! 41 42 36 37 ! 42 44 38 38
02,579
02,580 a0: 43 36 41 37 ! 43 38 45 37 ! 47 38 48 36 ! 47 34 48 32 ! 46 31 44 32 ! 42 31 41 33 ! 42 35 44 36 ! 46 35 44 34
02,581 a2: 46 33 45 35 ! 43 34 42 32 ! 44 33 46 34 ! 45 36 43 35 ! 41 34 42 36 ! 44 35 46 36 ! 45 34 43 33 ! 44 31 45 33
02,582 c0: 43 46 36 33 ! 41 45 37 35 ! 43 43 38 34 ! 45 42 37 32 ! 42 45 35 34 ! 44 43 36 33 ! 46 44 35 31 ! 44 45 34 33
02,583 c2: 43 46 36 33 ! 41 45 37 35 ! 43 43 38 34 ! 45 42 37 32 ! 42 45 35 34 ! 44 43 36 33 ! 46 44 35 31 ! 44 45 34 33
02,584
02,585 394e907d43
02,586
02,587 G:\Lookupperorama_r13\COLLISION_Hashliner>type 1.KnightTours.txt
02,588 a8C7E8G7H5G3H1F2H3G1E2C1A2B4A6B8D7F8H7G5F7H8G6H4G2E1C2A1B3A5B7D8C6A7C8E7G8H6G4H2F1D2B1A3B5D6F5D4F3E5C4B2D3F4E6C5A4B6D5F6E4C3D1E3
02,589
02,590 G:\Lookupperorama_r13\COLLISION_Hashliner>
02,591     a0]                                a2]                                a0]                                a2]
02,592 a8C7E8G7H5G3H1F2H3G1E2C1A2B4A6B8 D7F8H7G5F7H8G6H4G2E1C2A1B3A5B7D8 C6A7C8E7G8H6G4H2F1D2B1A3B5D6F5D4 F3E5C4B2D3F4E6C5A4B6D5F6E4C3D1E3
02,593
02,594 */
02,595
02,596 /*
02,597     _mm_storeu_si128((__m128i*)vector, *(__m128i *)(&a0YMM));
02,598     printf("a0lo: %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x ! %02x %02x %02x %02x\n",
02,599     vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],
02,600     vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15]);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
02,601  __mm_storeu_si128((__m128i*)vector, *((__m128i *)(&a0YMM)+1));
02,602  printf("a0hi: %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x | %02x %02x %02x %02x\n",
02,603      vector[0], vector[1], vector[2], vector[3], vector[4], vector[5], vector[6], vector[7],
02,604      vector[8], vector[9], vector[10], vector[11], vector[12], vector[13], vector[14], vector[15]);
02,605
02,606  printf("Above two should equal a0YMM\n");
02,607 */
02,608 //void SlowCopy128bit (const char *SOURCE, char *TARGET) { __mm_storeu_si128((__m128i *) (TARGET), __mm_loadu_si128((const __m128i *) (SOURCE))); }
02,609
02,610 // Okay, the final dump (shows XMM and YMM variants produce the same hash):
02,611 /*
02,612
02,613 D:\Lookupperorama_r13\COLLISION_Hashliner>Hashliner_DD AES_dump5byteshash.exe 1.KnightTours.txt
02,614 a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32
02,615 a1: 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,616 a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34
02,617 a3: 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,618
02,619 b0: 38 44 37 42 | 35 41 33 42 | 31 41 32 43 | 31 45 32 47
02,620 b1: 34 48 36 47 | 38 48 37 46 | 35 47 37 48 | 38 46 37 44
02,621 b2: 38 42 36 41 | 34 42 32 41 | 31 43 32 45 | 31 47 33 48
02,622 b3: 32 46 31 48 | 33 47 35 48 | 37 47 38 45 | 37 43 38 61
02,623
02,624 c0: 61 44 38 37 | 43 46 37 38 | 45 48 38 37 | 47 47 37 35
02,625 c1: 48 46 35 37 | 47 48 33 38 | 48 47 31 36 | 46 48 32 34
02,626 c2: 48 47 33 32 | 47 45 31 31 | 45 43 32 32 | 43 41 31 31
02,627 c3: 41 42 32 33 | 42 41 34 35 | 41 42 36 37 | 42 44 38 38
02,628
02,629 d0: 38 38 44 42 | 37 36 42 41 | 35 34 41 42 | 33 32 42 41
02,630 d1: 31 31 41 43 | 32 32 43 45 | 31 31 45 47 | 32 33 47 48
02,631 d2: 34 32 48 46 | 36 31 47 48 | 38 33 48 47 | 37 35 46 48
02,632 d3: 35 37 47 47 | 37 38 48 45 | 38 37 46 43 | 37 38 44 61
02,633
02,634 a0: 43 36 41 37 | 43 38 45 37 | 47 38 48 36 | 47 34 48 32
02,635 a1: 46 31 44 32 | 42 31 41 33 | 42 35 44 36 | 46 35 44 34
02,636 a2: 46 33 45 35 | 43 34 42 32 | 44 33 46 34 | 45 36 43 35
02,637 a3: 41 34 42 36 | 44 35 46 36 | 45 34 43 33 | 44 31 45 33
02,638
02,639 b0: 33 45 31 44 | 33 43 34 45 | 36 46 35 44 | 36 42 34 41
02,640 b1: 35 43 36 45 | 34 46 33 44 | 32 42 34 43 | 35 45 33 46
02,641 b2: 34 44 35 46 | 36 44 35 42 | 33 41 31 42 | 32 44 31 46
02,642 b3: 32 48 34 47 | 36 48 38 47 | 37 45 38 43 | 37 41 36 43
02,643
02,644 c0: 43 46 36 33 | 41 45 37 35 | 43 43 38 34 | 45 42 37 32
02,645 c1: 47 44 38 33 | 48 46 36 34 | 47 45 34 36 | 48 43 32 35
02,646 c2: 46 41 31 34 | 44 42 32 36 | 42 44 31 35 | 41 46 33 36
02,647 c3: 42 45 35 34 | 44 43 36 33 | 46 44 35 31 | 44 45 34 33
02,648
02,649 d0: 33 34 45 44 | 31 35 44 46 | 33 36 43 44 | 34 35 45 42
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,650 d1: 36 33 46 41 | 35 31 44 42 | 36 32 42 44 | 34 31 41 46
02,651 d2: 35 32 43 48 | 36 34 45 47 | 34 36 46 48 | 33 38 44 47
02,652 d3: 32 37 42 45 | 34 38 43 43 | 35 37 45 41 | 33 36 46 43
02,653
02,654 f4b027e3ab
02,655 a0: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32 | 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,656 a2: 44 37 46 38 | 48 37 47 35 | 46 37 48 38 | 47 36 48 34 | 47 32 45 31 | 43 32 41 31 | 42 33 41 35 | 42 37 44 38
02,657 b0: 38 44 37 42 | 35 41 33 42 | 31 41 32 43 | 31 45 32 47 | 34 48 36 47 | 38 48 37 46 | 35 47 37 48 | 38 46 37 44
02,658 b2: 38 42 36 41 | 34 42 32 41 | 31 43 32 45 | 31 47 33 48 | 32 46 31 48 | 33 47 35 48 | 37 47 38 45 | 37 43 38 61
02,659 c0: 61 44 38 37 | 43 46 37 38 | 45 48 38 37 | 47 47 37 35 | 48 47 33 32 | 47 45 31 31 | 45 43 32 32 | 43 41 31 31
02,660 c2: 48 46 35 37 | 47 48 33 38 | 48 47 31 36 | 46 48 32 34 | 41 42 32 33 | 42 41 34 35 | 41 42 36 37 | 42 44 38 38
02,661 d0: 38 38 44 42 | 37 36 42 41 | 35 34 41 42 | 33 32 42 41 | 34 32 48 46 | 36 31 47 48 | 38 33 48 47 | 37 35 46 48
02,662 d2: 31 31 41 43 | 32 32 43 45 | 31 31 45 47 | 32 33 47 48 | 35 37 47 47 | 37 38 48 45 | 38 37 46 43 | 37 38 44 61
02,663
02,664 a0lo: 61 38 43 37 | 45 38 47 37 | 48 35 47 33 | 48 31 46 32
02,665 a0hi: 48 33 47 31 | 45 32 43 31 | 41 32 42 34 | 41 36 42 38
02,666 Above two should equal a0YMM
02,667 a0: 43 36 41 37 | 43 38 45 37 | 47 38 48 36 | 47 34 48 32 | 46 31 44 32 | 42 31 41 33 | 42 35 44 36 | 46 35 44 34
02,668 a2: 46 33 45 35 | 43 34 42 32 | 44 33 46 34 | 45 36 43 35 | 41 34 42 36 | 44 35 46 36 | 45 34 43 33 | 44 31 45 33
02,669 b0: 33 45 31 44 | 33 43 34 45 | 36 46 35 44 | 36 42 34 41 | 35 43 36 45 | 34 46 33 44 | 32 42 34 43 | 35 45 33 46
02,670 b2: 34 44 35 46 | 36 44 35 42 | 33 41 31 42 | 32 44 31 46 | 32 48 34 47 | 36 48 38 47 | 37 45 38 43 | 37 41 36 43
02,671 c0: 43 46 36 33 | 41 45 37 35 | 43 43 38 34 | 45 42 37 32 | 46 41 31 34 | 44 42 32 36 | 42 44 31 35 | 41 46 33 36
02,672 c2: 47 44 38 33 | 48 46 36 34 | 47 45 34 36 | 48 43 32 35 | 42 45 35 34 | 44 43 36 33 | 46 44 35 31 | 44 45 34 33
02,673 d0: 33 34 45 44 | 31 35 44 46 | 33 36 43 44 | 34 35 45 42 | 35 32 43 48 | 36 34 45 47 | 34 36 46 48 | 33 38 44 47
02,674 d2: 36 33 46 41 | 35 31 44 42 | 36 32 42 44 | 34 31 41 46 | 32 37 42 45 | 34 38 43 43 | 35 37 45 41 | 33 36 46 43
02,675
02,676 a0lo: 43 36 41 37 | 43 38 45 37 | 47 38 48 36 | 47 34 48 32
02,677 a0hi: 46 31 44 32 | 42 31 41 33 | 42 35 44 36 | 46 35 44 34
02,678 Above two should equal a0YMM
02,679 f4b027e3ab
02,680
02,681 D:\Lookupperorama_r13\COLLISION_Hashliner>
02,682 */
02,683
02,684 //In YMM should swap c1 and c2, also d1 and d2
02,685
02,686 //hashA = _mm_aesenc_si128(hashA, a0);
02,687 hashA = _mm_aesenc_si128(hashA, *(_m128i *)(&a0YMM));
02,688 //hashB = _mm_aesenc_si128(hashB, b0);
02,689 hashB = _mm_aesenc_si128(hashB, *(_m128i *)(&b0YMM));
02,690 //hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b0YMM)+1));
02,691 //hashC = _mm_aesenc_si128(hashC, c0);
02,692 hashC = _mm_aesenc_si128(hashC, *(_m128i *)(&c0YMM));
02,693 //hashD = _mm_aesenc_si128(hashD, d0);
02,694 hashD = _mm_aesenc_si128(hashD, *(_m128i *)(&d0YMM));
02,695
02,696 //hashA = _mm_aesenc_si128(hashA, a1);
02,697 hashA = _mm_aesenc_si128(hashA, *((_m128i *)(&a0YMM)+1));
02,698 //hashB = _mm_aesenc_si128(hashB, b1);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,699         hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b0YMM)+1));
02,700         //hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b0YMM)));
02,701         //hashC = _mm_aesenc_si128(hashC, c1);
02,702         hashC = _mm_aesenc_si128(hashC, *((_m128i *)(&c2YMM)));
02,703         //hashD = _mm_aesenc_si128(hashD, d1);
02,704         hashD = _mm_aesenc_si128(hashD, *((_m128i *)(&d2YMM)));
02,705
02,706         //hashA = _mm_aesenc_si128(hashA, a2);
02,707         hashA = _mm_aesenc_si128(hashA, *((_m128i *)(&a2YMM)));
02,708         //hashB = _mm_aesenc_si128(hashB, b2);
02,709         hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b2YMM)));
02,710         //hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b2YMM)+1));
02,711         //hashC = _mm_aesenc_si128(hashC, c2);
02,712         hashC = _mm_aesenc_si128(hashC, *((_m128i *)(&c0YMM)+1));
02,713         //hashD = _mm_aesenc_si128(hashD, d2);
02,714         hashD = _mm_aesenc_si128(hashD, *((_m128i *)(&d0YMM)+1));
02,715
02,716         //hashA = _mm_aesenc_si128(hashA, a3);
02,717         hashA = _mm_aesenc_si128(hashA, *((_m128i *)(&a2YMM)+1));
02,718         //hashB = _mm_aesenc_si128(hashB, b3);
02,719         hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b2YMM)+1));
02,720         //hashB = _mm_aesenc_si128(hashB, *((_m128i *)(&b2YMM)));
02,721         //hashC = _mm_aesenc_si128(hashC, c3);
02,722         hashC = _mm_aesenc_si128(hashC, *((_m128i *)(&c2YMM)+1));
02,723         //hashD = _mm_aesenc_si128(hashD, d3);
02,724         hashD = _mm_aesenc_si128(hashD, *((_m128i *)(&d2YMM)+1));
02,725
02,726         hashA = _mm_aesenc_si128(hashA, hashB);
02,727         hashA = _mm_aesenc_si128(hashA, hashC);
02,728         hashA = _mm_aesenc_si128(hashA, hashD);
02,729         length = length - 64;
02,730     }
02,731 }
02,732
02,733 ptr128 = (_m128i *)buffer;
02,734 if (length >= 16) {
02,735     Cycles = length/16;
02,736     for(; Cycles--; buffer += 16) {
02,737         AgainstRules = _mm_loadu_si128(ptr128++);
02,738         GumbotronREVER = _mm_shuffle_epi8 (AgainstRules, ReverseMask);
02,739         GumbotronINTER = _mm_shuffle_epi8 (AgainstRules, InterleaveMask);
02,740         GumbotronREVERINTER = _mm_shuffle_epi8 (GumbotronREVER, InterleaveMask);
02,741         hashA = _mm_aesenc_si128(hashA, AgainstRules);
02,742         hashB = _mm_aesenc_si128(hashB, GumbotronREVER);
02,743         hashC = _mm_aesenc_si128(hashC, GumbotronINTER);
02,744         hashD = _mm_aesenc_si128(hashD, GumbotronREVERINTER);
02,745         hashA = _mm_aesenc_si128(hashA, hashB);
02,746         hashA = _mm_aesenc_si128(hashA, hashC);
02,747         hashA = _mm_aesenc_si128(hashA, hashD);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,748         length = length - 16;
02,749     }
02,750 }
02,751 // Inhere using Pippip's approach to read past the end ("the dirty" sentinel like style, or more like padding):
02,752 if (length&(16-1)) {
02,753     AgainstRules = _mm_loadu_si128(ptr128);
02,754     //AgainstRules = _mm_srli_si128 (AgainstRules, 16-length); // catastrophic error: Intrinsic parameter must be an immediate value
02,755     AgainstRules = _mm_and_si128 (AgainstRules, Mumbotron[length]);
02,756     //Gumbotron = _mm_slli_si128 (Gumbotron, 16-length); // catastrophic error: Intrinsic parameter must be an immediate value
02,757     Gumbotron = _mm_and_si128 (hashB, Gumbotron[length]);
02,758     AgainstRules = _mm_or_si128 (AgainstRules, Gumbotron);
02,759
02,760     GumbotronREVER = _mm_shuffle_epi8 (AgainstRules, ReverseMask);
02,761     GumbotronINTER = _mm_shuffle_epi8 (AgainstRules, InterleaveMask);
02,762     GumbotronREVERINTER = _mm_shuffle_epi8 (GumbotronREVER, InterleaveMask);
02,763     hashA = _mm_aesenc_si128(hashA, AgainstRules);
02,764     hashB = _mm_aesenc_si128(hashB, GumbotronREVER);
02,765     hashC = _mm_aesenc_si128(hashC, GumbotronINTER);
02,766     hashD = _mm_aesenc_si128(hashD, GumbotronREVERINTER);
02,767     hashA = _mm_aesenc_si128(hashA, hashB);
02,768     hashA = _mm_aesenc_si128(hashA, hashC);
02,769     hashA = _mm_aesenc_si128(hashA, hashD);
02,770 }
02,771 //SlowCopy128bit( (const char *)&hashA, (char *)&DDAES[0]);
02,772 memcpy ( LoPart, (const char *)&hashA, 8);
02,773 memcpy ( HiPart, (const char *)&hashA+8, 8);
02,774 }
02,775 // Revision 2021-Aug-22 ]

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
F:\qb64>type m.bat
@if exist MASAKARI.current.tree.fullpaths.txt del MASAKARI.current.tree.fullpaths.txt
@rem For fullpaths use:
@rem yoshi -f -oMASAKARI.current.tree.fullpaths.txt
@rem yoshi -oMASAKARI.current.tree.fullpaths.txt "%1"
@dir %1/b/a-d>MASAKARI.current.tree.fullpaths.txt
"MASAKARI_Vanilla.exe" MASAKARI.current.tree.fullpaths.txt
@if exist MASAKARI.current.tree.fullpaths.txt del MASAKARI.current.tree.fullpaths.txt
```

```
F:\qb64>type mGesch.bat
1251toGesch "%1"
MASAKARI_Wrapper "%1.Gesch"
```

```
F:\qb64>type ShowInGesch.bat
@if '%1' == '' goto Skip
@if not exist "%1.Gesch" goto Convert
@goto ViewIt
:Convert
@UTF8toGesch.exe %1
:ViewIt
@MASAKARI_Vanilla.exe "%1.Gesch"
:Skip
```

```
F:\qb64>type z.bat
qb64 MASAKARI_Vanilla.BAS
```

```
copy MASAKARI_Vanilla.BAS MASAKARI_Wrapper.BAS /y
copy MASAKARI_Vanilla.BAS MASAKARI_Vanilla_NEC.BAS /y
```

```
F:\qb64>type _Make_EXEs.bat
qb64 -c MASAKARI_Vanilla.BAS
qb64 -c MASAKARI_Wrapper.BAS
qb64 -c MASAKARI_Vanilla_NEC.BAS
```

```
fc MASAKARI_Vanilla.BAS MASAKARI_Wrapper.BAS
echo.
fc MASAKARI_Vanilla.BAS MASAKARI_Vanilla_NEC.BAS
```

```
F:\qb64>
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

# LineJustify\_PAGODao5.c:

```
00,001 //LineJustify_PAGODao5.c
00,002 // Since new Leprechaun r.17 changed \t with '_', inhere same has to happen, 2021-Apr-21
00,003 //LineWordreporter.c, r1, 2013
00,004 //Linereporter.c
00,005 // strstr_SHORT-SHOWDOWN.c, revision 6, 2010 Dec 22.
00,006 // Home of Railgun_Quadruplet: www.sanmayce.com/Railgun
00,007
00,008 // In previous revisions invocation of GNU strstr were with an overhead: 2 parameters not needed, sorry, it was unintentionally.
00,009
00,010 // Wow, another catastrophe of mine,
00,011 // Karp-Rabin-Kaze appears to be inferior compared to GNU strstr.
00,012 // Back to school... no, no, back to kindergarten!
00,013 // Karp-Rabin-Kaze bends a knee before strstr_GNU_C Library, bravo Stephen R. van den Berg, I will have fun reading-understanding-learning it, for sure, thanks.
00,014 // Here a lite testbed(OSHO.TXT) is set, the other(26GB with 400+ million lines) will be tested off-line later, he-he.
00,015 // Thanks a lot also to,
00,016 // http://www-igm.univ-mlv.fr/~lecroq/string/index.html
00,017 // amazing site and GREAT/RICH resource on search technique.
00,018
00,019 // Revision 5: What a dumbo-tester I am, to test only with one(from 2002) C compiler, the damage is undone, here is:
00,020 //          Intel(R) C++ Compiler Professional for applications running on IA-32, Version 11.1(from 2010) along with
00,021 //          Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 13.10.3077 for 80x86.
00,022 // Compile line: icl /O3 /Qparallel strstr_SHORT-SHOWDOWN.c
00,023 // Compile line: cl /Ox strstr_SHORT-SHOWDOWN.c
00,024 // I guess '/Qparallel' gives(don't know) a very little boost if any.
00,025
00,026 // Revision 6: Microsoft (R) Optimizing Compiler Version 16.00.30319.01 step in, from VS2010.
00,027 // Short bottomline: Railgun_Quadruplet is very good but needs to be boosted/refined!
00,028
00,029 // Note: I have some high hopes for Railgun_Quadruplet, especially for new architectures after Core 2 i.e. i3[+].
00,030
00,031 /*
00,032 [On Intel T3400 CPU: Microsoft 16.00.30319.01 vs Intel IA-32 11.1:]
00,033 Input file: OSHO.TXT
00,034 Number/Size of lines: 2,459,508/206,908,949
00,035 Shortest/Longest line: 0/161
00,036
00,037 [strstr_SHORT-SHOWDOWN_Microsoft_Ox.exe:]
00,038
00,039 Searching for Pattern('an',2bytes) into String(206908949bytes) line-by-line ...
00,040
00,041 LinesEncountered: 2459508
00,042 strstr_Microsoft_hits/strstr_Microsoft_clocks: 75781/786
00,043 strstr_Microsoft performance: 172KB/clock
00,044 StrnglenTRAVERSED: 138478024 bytes
00,045 LinesEncountered: 2459508
00,046 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 75781/483
00,047 strstr_GNU_C_Library performance: 279KB/clock
00,048 StrnglenTRAVERSED: 138478024 bytes
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,049 LinesEncountered: 2459508
00,050 Railgun_hits/Railgun_clocks: 75781/515
00,051 Railgun performance: 262KB/clock
00,052 StrnglenTRAVERSED: 138478024 bytes
00,053 LinesEncountered: 2459508
00,054 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 75781/509
00,055 Railgun_Quadruplet performance: 265KB/clock
00,056 StrnglenTRAVERSED: 138478024 bytes
00,057
00,058 Searching for Pattern('to',2bytes) into String(206908949bytes) line-by-line ...
00,059
00,060 LinesEncountered: 2459508
00,061 strstr_Microsoft_hits/strstr_Microsoft_clocks: 48760/896
00,062 strstr_Microsoft performance: 179KB/clock
00,063 StrnglenTRAVERSED: 164505415 bytes
00,064 LinesEncountered: 2459508
00,065 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 48760/540
00,066 strstr_GNU_C_Library performance: 297KB/clock
00,067 StrnglenTRAVERSED: 164505415 bytes
00,068 LinesEncountered: 2459508
00,069 Railgun_hits/Railgun_clocks: 48760/546
00,070 Railgun performance: 294KB/clock
00,071 StrnglenTRAVERSED: 164505415 bytes
00,072 LinesEncountered: 2459508
00,073 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 48760/540
00,074 Railgun_Quadruplet performance: 297KB/clock
00,075 StrnglenTRAVERSED: 164505415 bytes
00,076
00,077 Searching for Pattern('TDK',3bytes) into String(206908949bytes) line-by-line ...
00,078
00,079 LinesEncountered: 2459508
00,080 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/929
00,081 strstr_Microsoft performance: 214KB/clock
00,082 StrnglenTRAVERSED: 204449441 bytes
00,083 LinesEncountered: 2459508
00,084 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/485
00,085 strstr_GNU_C_Library performance: 411KB/clock
00,086 StrnglenTRAVERSED: 204449441 bytes
00,087 LinesEncountered: 2459508
00,088 Railgun_hits/Railgun_clocks: 0/588
00,089 Railgun performance: 339KB/clock
00,090 StrnglenTRAVERSED: 204449441 bytes
00,091 LinesEncountered: 2459508
00,092 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/584
00,093 Railgun_Quadruplet performance: 341KB/clock
00,094 StrnglenTRAVERSED: 204449441 bytes
00,095
00,096 Searching for Pattern('the',3bytes) into String(206908949bytes) line-by-line ...
00,097

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,098 LinesEncountered: 2459508
00,099 strstr_Microsoft_hits/strstr_Microsoft_clocks: 74500/807
00,100 strstr_Microsoft performance: 164KB/clock
00,101 StrnglenTRAVERSED: 135882884 bytes
00,102 LinesEncountered: 2459508
00,103 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 74500/518
00,104 strstr_GNU_C_Library performance: 256KB/clock
00,105 StrnglenTRAVERSED: 135882884 bytes
00,106 LinesEncountered: 2459508
00,107 Railgun_hits/Railgun_clocks: 74500/505
00,108 Railgun performance: 262KB/clock
00,109 StrnglenTRAVERSED: 135882884 bytes
00,110 LinesEncountered: 2459508
00,111 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 74500/503
00,112 Railgun_Quadruplet performance: 263KB/clock
00,113 StrnglenTRAVERSED: 135882884 bytes
00,114
00,115 Searching for Pattern('fast',4bytes) into String(206908949bytes) line-by-line ...
00,116
00,117 LinesEncountered: 2459508
00,118 strstr_Microsoft_hits/strstr_Microsoft_clocks: 336/946
00,119 strstr_Microsoft performance: 210KB/clock
00,120 StrnglenTRAVERSED: 204186782 bytes
00,121 LinesEncountered: 2459508
00,122 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 336/503
00,123 strstr_GNU_C_Library performance: 396KB/clock
00,124 StrnglenTRAVERSED: 204186782 bytes
00,125 LinesEncountered: 2459508
00,126 Railgun_hits/Railgun_clocks: 336/588
00,127 Railgun performance: 339KB/clock
00,128 StrnglenTRAVERSED: 204186782 bytes
00,129 LinesEncountered: 2459508
00,130 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 336/504
00,131 Railgun_Quadruplet performance: 395KB/clock
00,132 StrnglenTRAVERSED: 204186782 bytes
00,133
00,134 Searching for Pattern('easy',4bytes) into String(206908949bytes) line-by-line ...
00,135
00,136 LinesEncountered: 2459508
00,137 strstr_Microsoft_hits/strstr_Microsoft_clocks: 301/1123
00,138 strstr_Microsoft performance: 177KB/clock
00,139 StrnglenTRAVERSED: 204202166 bytes
00,140 LinesEncountered: 2459508
00,141 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 301/672
00,142 strstr_GNU_C_Library performance: 296KB/clock
00,143 StrnglenTRAVERSED: 204202166 bytes
00,144 LinesEncountered: 2459508
00,145 Railgun_hits/Railgun_clocks: 301/590
00,146 Railgun performance: 337KB/clock

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,147 StrnglenTRAVERSED: 204202166 bytes
00,148 LinesEncountered: 2459508
00,149 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 301/660
00,150 Railgun_Quadruplet performance: 302KB/clock
00,151 StrnglenTRAVERSED: 204202166 bytes
00,152
00,153 Searching for Pattern('grmb1',5bytes) into String(206908949bytes) line-by-line ...
00,154
00,155 LinesEncountered: 2459508
00,156 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/953
00,157 strstr_Microsoft performance: 209KB/clock
00,158 StrnglenTRAVERSED: 204449441 bytes
00,159 LinesEncountered: 2459508
00,160 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/507
00,161 strstr_GNU_C_Library performance: 393KB/clock
00,162 StrnglenTRAVERSED: 204449441 bytes
00,163 LinesEncountered: 2459508
00,164 Railgun_hits/Railgun_clocks: 0/584
00,165 Railgun performance: 341KB/clock
00,166 StrnglenTRAVERSED: 204449441 bytes
00,167 LinesEncountered: 2459508
00,168 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/505
00,169 Railgun_Quadruplet performance: 395KB/clock
00,170 StrnglenTRAVERSED: 204449441 bytes
00,171
00,172 Searching for Pattern('email',5bytes) into String(206908949bytes) line-by-line ...
00,173
00,174 LinesEncountered: 2459508
00,175 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1120
00,176 strstr_Microsoft performance: 178KB/clock
00,177 StrnglenTRAVERSED: 204449414 bytes
00,178 LinesEncountered: 2459508
00,179 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/663
00,180 strstr_GNU_C_Library performance: 301KB/clock
00,181 StrnglenTRAVERSED: 204449414 bytes
00,182 LinesEncountered: 2459508
00,183 Railgun_hits/Railgun_clocks: 0/590
00,184 Railgun performance: 338KB/clock
00,185 StrnglenTRAVERSED: 204449414 bytes
00,186 LinesEncountered: 2459508
00,187 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/658
00,188 Railgun_Quadruplet performance: 303KB/clock
00,189 StrnglenTRAVERSED: 204449414 bytes
00,190
00,191 Searching for Pattern('pasting',7bytes) into String(206908949bytes) line-by-line ...
00,192
00,193 LinesEncountered: 2459508
00,194 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/947
00,195 strstr_Microsoft performance: 210KB/clock
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,196 StrnglenTRAVERSED: 204449363 bytes
00,197 LinesEncountered: 2459508
00,198 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/503
00,199 strstr_GNU_C_Library performance: 396KB/clock
00,200 StrnglenTRAVERSED: 204449363 bytes
00,201 LinesEncountered: 2459508
00,202 Railgun_hits/Railgun_clocks: 0/576
00,203 Railgun performance: 346KB/clock
00,204 StrnglenTRAVERSED: 204449363 bytes
00,205 LinesEncountered: 2459508
00,206 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/497
00,207 Railgun_Quadruplet performance: 401KB/clock
00,208 StrnglenTRAVERSED: 204449363 bytes
00,209
00,210 Searching for Pattern('amazing',7bytes) into String(206908949bytes) line-by-line ...
00,211
00,212 LinesEncountered: 2459508
00,213 strstr_Microsoft_hits/strstr_Microsoft_clocks: 19/1044
00,214 strstr_Microsoft performance: 191KB/clock
00,215 StrnglenTRAVERSED: 204432134 bytes
00,216 LinesEncountered: 2459508
00,217 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 19/589
00,218 strstr_GNU_C_Library performance: 338KB/clock
00,219 StrnglenTRAVERSED: 204432134 bytes
00,220 LinesEncountered: 2459508
00,221 Railgun_hits/Railgun_clocks: 19/577
00,222 Railgun performance: 345KB/clock
00,223 StrnglenTRAVERSED: 204432134 bytes
00,224 LinesEncountered: 2459508
00,225 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 19/575
00,226 Railgun_Quadruplet performance: 347KB/clock
00,227 StrnglenTRAVERSED: 204432134 bytes
00,228
00,229 Searching for Pattern('underdog',8bytes) into String(206908949bytes) line-by-line ...
00,230
00,231 LinesEncountered: 2459508
00,232 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/979
00,233 strstr_Microsoft performance: 203KB/clock
00,234 StrnglenTRAVERSED: 204449185 bytes
00,235 LinesEncountered: 2459508
00,236 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/530
00,237 strstr_GNU_C_Library performance: 376KB/clock
00,238 StrnglenTRAVERSED: 204449185 bytes
00,239 LinesEncountered: 2459508
00,240 Railgun_hits/Railgun_clocks: 0/573
00,241 Railgun performance: 348KB/clock
00,242 StrnglenTRAVERSED: 204449185 bytes
00,243 LinesEncountered: 2459508
00,244 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/518

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,245 Railgun_Quadruplet performance: 385KB/clock
00,246 StrnglenTRAVERSED: 204449185 bytes
00,247
00,248 Searching for Pattern('superdog',8bytes) into String(206908949bytes) line-by-line ...
00,249
00,250 LinesEncountered: 2459508
00,251 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1023
00,252 strstr_Microsoft performance: 195KB/clock
00,253 StrnglenTRAVERSED: 204449441 bytes
00,254 LinesEncountered: 2459508
00,255 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/575
00,256 strstr_GNU_C_Library performance: 347KB/clock
00,257 StrnglenTRAVERSED: 204449441 bytes
00,258 LinesEncountered: 2459508
00,259 Railgun_hits/Railgun_clocks: 0/575
00,260 Railgun performance: 347KB/clock
00,261 StrnglenTRAVERSED: 204449441 bytes
00,262 LinesEncountered: 2459508
00,263 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/561
00,264 Railgun_Quadruplet performance: 355KB/clock
00,265 StrnglenTRAVERSED: 204449441 bytes
00,266
00,267 Searching for Pattern('participants',12bytes) into String(206908949bytes) line-by-line ...
00,268
00,269 LinesEncountered: 2459508
00,270 strstr_Microsoft_hits/strstr_Microsoft_clocks: 8/948
00,271 strstr_Microsoft performance: 210KB/clock
00,272 StrnglenTRAVERSED: 204441500 bytes
00,273 LinesEncountered: 2459508
00,274 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 8/503
00,275 strstr_GNU_C_Library performance: 396KB/clock
00,276 StrnglenTRAVERSED: 204441500 bytes
00,277 LinesEncountered: 2459508
00,278 Railgun_hits/Railgun_clocks: 8/561
00,279 Railgun performance: 355KB/clock
00,280 StrnglenTRAVERSED: 204441500 bytes
00,281 LinesEncountered: 2459508
00,282 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 8/485
00,283 Railgun_Quadruplet performance: 411KB/clock
00,284 StrnglenTRAVERSED: 204441500 bytes
00,285
00,286 Searching for Pattern('skilllessness',12bytes) into String(206908949bytes) line-by-line ...
00,287
00,288 LinesEncountered: 2459508
00,289 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1021
00,290 strstr_Microsoft performance: 195KB/clock
00,291 StrnglenTRAVERSED: 204449441 bytes
00,292 LinesEncountered: 2459508
00,293 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/573

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,294 strstr_GNU_C_Library performance: 348KB/clock
00,295 StrnglenTRAVERSED: 204449441 bytes
00,296 LinesEncountered: 2459508
00,297 Railgun_hits/Railgun_clocks: 0/567
00,298 Railgun performance: 352KB/clock
00,299 StrnglenTRAVERSED: 204449441 bytes
00,300 LinesEncountered: 2459508
00,301 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/549
00,302 Railgun_Quadruplet performance: 363KB/clock
00,303 StrnglenTRAVERSED: 204449441 bytes
00,304
00,305 Searching for Pattern('I should have known',19bytes) into String(206908949bytes) line-by-line ...
00,306
00,307 LinesEncountered: 2459508
00,308 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/934
00,309 strstr_Microsoft performance: 213KB/clock
00,310 StrnglenTRAVERSED: 204449346 bytes
00,311 LinesEncountered: 2459508
00,312 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/490
00,313 strstr_GNU_C_Library performance: 407KB/clock
00,314 StrnglenTRAVERSED: 204449346 bytes
00,315 LinesEncountered: 2459508
00,316 Railgun_hits/Railgun_clocks: 0/540
00,317 Railgun performance: 369KB/clock
00,318 StrnglenTRAVERSED: 204449346 bytes
00,319 LinesEncountered: 2459508
00,320 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/456
00,321 Railgun_Quadruplet performance: 437KB/clock
00,322 StrnglenTRAVERSED: 204449346 bytes
00,323
00,324 Searching for Pattern('human consciousness',19bytes) into String(206908949bytes) line-by-line ...
00,325
00,326 LinesEncountered: 2459508
00,327 strstr_Microsoft_hits/strstr_Microsoft_clocks: 32/1010
00,328 strstr_Microsoft performance: 197KB/clock
00,329 StrnglenTRAVERSED: 204422699 bytes
00,330 LinesEncountered: 2459508
00,331 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 32/560
00,332 strstr_GNU_C_Library performance: 356KB/clock
00,333 StrnglenTRAVERSED: 204422699 bytes
00,334 LinesEncountered: 2459508
00,335 Railgun_hits/Railgun_clocks: 32/545
00,336 Railgun performance: 366KB/clock
00,337 StrnglenTRAVERSED: 204422699 bytes
00,338 LinesEncountered: 2459508
00,339 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 32/512
00,340 Railgun_Quadruplet performance: 389KB/clock
00,341 StrnglenTRAVERSED: 204422699 bytes
00,342

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,343 Doing Search for 8x2 Patterns into String(206908949bytes) as-one-line ...
00,344 Found ('an') 1987797 time(s), Railgun performance: 300KB/clock
00,345 Found ('to') 1076629 time(s), Railgun performance: 300KB/clock
00,346 Found ('TDK') 0 time(s), Railgun performance: 515KB/clock
00,347 Found ('the') 2114180 time(s), Railgun performance: 403KB/clock
00,348 Found ('fast') 5945 time(s), Railgun performance: 561KB/clock
00,349 Found ('easy') 5191 time(s), Railgun performance: 612KB/clock
00,350 Found ('grmb1') 0 time(s), Railgun performance: 805KB/clock
00,351 Found ('email') 1 time(s), Railgun performance: 716KB/clock
00,352 Found ('pasting') 2 time(s), Railgun performance: 990KB/clock
00,353 Found ('amazing') 323 time(s), Railgun performance: 990KB/clock
00,354 Found ('underdog') 4 time(s), Railgun performance: 1167KB/clock
00,355 Found ('superdog') 0 time(s), Railgun performance: 1074KB/clock
00,356 Found ('participants') 147 time(s), Railgun performance: 1603KB/clock
00,357 Found ('skilllessness') 0 time(s), Railgun performance: 1422KB/clock
00,358 Found ('I should have known') 1 time(s), Railgun performance: 1603KB/clock
00,359 Found ('human consciousness') 519 time(s), Railgun performance: 1603KB/clock
00,360 Railgun 8x2 i.e. average performance: 682KB/clock
00,361
00,362 [strsr_SHORT-SHOWDOWN_Intel_O3_Qparallel.exe:]
00,363
00,364 Searching for Pattern('an',2bytes) into String(206908949bytes) line-by-line ...
00,365
00,366 LinesEncountered: 2459508
00,367 strsr_Microsoft_hits/strsr_Microsoft_clocks: 75781/917
00,368 strsr_Microsoft performance: 147KB/clock
00,369 StrnglenTRAVERSED: 138478024 bytes
00,370 LinesEncountered: 2459508
00,371 strsr_GNU_C_Library_hits/strsr_GNU_C_Library_clocks: 75781/503
00,372 strsr_GNU_C_Library performance: 268KB/clock
00,373 StrnglenTRAVERSED: 138478024 bytes
00,374 LinesEncountered: 2459508
00,375 Railgun_hits/Railgun_clocks: 75781/526
00,376 Railgun performance: 257KB/clock
00,377 StrnglenTRAVERSED: 138478024 bytes
00,378 LinesEncountered: 2459508
00,379 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 75781/484
00,380 Railgun_Quadruplet performance: 279KB/clock
00,381 StrnglenTRAVERSED: 138478024 bytes
00,382
00,383 Searching for Pattern('to',2bytes) into String(206908949bytes) line-by-line ...
00,384
00,385 LinesEncountered: 2459508
00,386 strsr_Microsoft_hits/strsr_Microsoft_clocks: 48760/1050
00,387 strsr_Microsoft performance: 152KB/clock
00,388 StrnglenTRAVERSED: 164505415 bytes
00,389 LinesEncountered: 2459508
00,390 strsr_GNU_C_Library_hits/strsr_GNU_C_Library_clocks: 48760/562
00,391 strsr_GNU_C_Library performance: 285KB/clock

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,392 StrnglenTRAVERSED: 164505415 bytes
00,393 LinesEncountered: 2459508
00,394 Railgun_hits/Railgun_clocks: 48760/550
00,395 Railgun performance: 292KB/clock
00,396 StrnglenTRAVERSED: 164505415 bytes
00,397 LinesEncountered: 2459508
00,398 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 48760/518
00,399 Railgun_Quadruplet performance: 310KB/clock
00,400 StrnglenTRAVERSED: 164505415 bytes
00,401
00,402 Searching for Pattern('TDK',3bytes) into String(206908949bytes) line-by-line ...
00,403
00,404 LinesEncountered: 2459508
00,405 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1120
00,406 strstr_Microsoft performance: 178KB/clock
00,407 StrnglenTRAVERSED: 204449441 bytes
00,408 LinesEncountered: 2459508
00,409 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/532
00,410 strstr_GNU_C_Library performance: 375KB/clock
00,411 StrnglenTRAVERSED: 204449441 bytes
00,412 LinesEncountered: 2459508
00,413 Railgun_hits/Railgun_clocks: 0/586
00,414 Railgun performance: 340KB/clock
00,415 StrnglenTRAVERSED: 204449441 bytes
00,416 LinesEncountered: 2459508
00,417 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/569
00,418 Railgun_Quadruplet performance: 350KB/clock
00,419 StrnglenTRAVERSED: 204449441 bytes
00,420
00,421 Searching for Pattern('the',3bytes) into String(206908949bytes) line-by-line ...
00,422
00,423 LinesEncountered: 2459508
00,424 strstr_Microsoft_hits/strstr_Microsoft_clocks: 74500/938
00,425 strstr_Microsoft performance: 141KB/clock
00,426 StrnglenTRAVERSED: 135882884 bytes
00,427 LinesEncountered: 2459508
00,428 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 74500/543
00,429 strstr_GNU_C_Library performance: 244KB/clock
00,430 StrnglenTRAVERSED: 135882884 bytes
00,431 LinesEncountered: 2459508
00,432 Railgun_hits/Railgun_clocks: 74500/507
00,433 Railgun performance: 261KB/clock
00,434 StrnglenTRAVERSED: 135882884 bytes
00,435 LinesEncountered: 2459508
00,436 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 74500/488
00,437 Railgun_Quadruplet performance: 271KB/clock
00,438 StrnglenTRAVERSED: 135882884 bytes
00,439
00,440 Searching for Pattern('fast',4bytes) into String(206908949bytes) line-by-line ...

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,441
00,442 LinesEncountered: 2459508
00,443 strstr_Microsoft_hits/strstr_Microsoft_clocks: 336/1135
00,444 strstr_Microsoft performance: 175KB/clock
00,445 StrnglenTRAVERSED: 204186782 bytes
00,446 LinesEncountered: 2459508
00,447 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 336/531
00,448 strstr_GNU_C_Library performance: 375KB/clock
00,449 StrnglenTRAVERSED: 204186782 bytes
00,450 LinesEncountered: 2459508
00,451 Railgun_hits/Railgun_clocks: 336/587
00,452 Railgun performance: 339KB/clock
00,453 StrnglenTRAVERSED: 204186782 bytes
00,454 LinesEncountered: 2459508
00,455 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 336/521
00,456 Railgun_Quadruplet performance: 382KB/clock
00,457 StrnglenTRAVERSED: 204186782 bytes
00,458
00,459 Searching for Pattern('easy',4bytes) into String(206908949bytes) line-by-line ...
00,460
00,461 LinesEncountered: 2459508
00,462 strstr_Microsoft_hits/strstr_Microsoft_clocks: 301/1310
00,463 strstr_Microsoft performance: 152KB/clock
00,464 StrnglenTRAVERSED: 204202166 bytes
00,465 LinesEncountered: 2459508
00,466 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 301/709
00,467 strstr_GNU_C_Library performance: 281KB/clock
00,468 StrnglenTRAVERSED: 204202166 bytes
00,469 LinesEncountered: 2459508
00,470 Railgun_hits/Railgun_clocks: 301/588
00,471 Railgun performance: 339KB/clock
00,472 StrnglenTRAVERSED: 204202166 bytes
00,473 LinesEncountered: 2459508
00,474 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 301/669
00,475 Railgun_Quadruplet performance: 298KB/clock
00,476 StrnglenTRAVERSED: 204202166 bytes
00,477
00,478 Searching for Pattern('grmb1',5bytes) into String(206908949bytes) line-by-line ...
00,479
00,480 LinesEncountered: 2459508
00,481 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1140
00,482 strstr_Microsoft performance: 175KB/clock
00,483 StrnglenTRAVERSED: 204449441 bytes
00,484 LinesEncountered: 2459508
00,485 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/532
00,486 strstr_GNU_C_Library performance: 375KB/clock
00,487 StrnglenTRAVERSED: 204449441 bytes
00,488 LinesEncountered: 2459508
00,489 Railgun_hits/Railgun_clocks: 0/580

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,490 Railgun performance: 344KB/clock
00,491 StrnglenTRAVERSED: 204449441 bytes
00,492 LinesEncountered: 2459508
00,493 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/521
00,494 Railgun_Quadruplet performance: 383KB/clock
00,495 StrnglenTRAVERSED: 204449441 bytes
00,496
00,497 Searching for Pattern('email',5bytes) into String(206908949bytes) line-by-line ...
00,498
00,499 LinesEncountered: 2459508
00,500 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1307
00,501 strstr_Microsoft performance: 152KB/clock
00,502 StrnglenTRAVERSED: 204449414 bytes
00,503 LinesEncountered: 2459508
00,504 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/699
00,505 strstr_GNU_C_Library performance: 285KB/clock
00,506 StrnglenTRAVERSED: 204449414 bytes
00,507 LinesEncountered: 2459508
00,508 Railgun_hits/Railgun_clocks: 0/587
00,509 Railgun performance: 340KB/clock
00,510 StrnglenTRAVERSED: 204449414 bytes
00,511 LinesEncountered: 2459508
00,512 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/666
00,513 Railgun_Quadruplet performance: 299KB/clock
00,514 StrnglenTRAVERSED: 204449414 bytes
00,515
00,516 Searching for Pattern('pasting',7bytes) into String(206908949bytes) line-by-line ...
00,517
00,518 LinesEncountered: 2459508
00,519 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1135
00,520 strstr_Microsoft performance: 175KB/clock
00,521 StrnglenTRAVERSED: 204449363 bytes
00,522 LinesEncountered: 2459508
00,523 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/533
00,524 strstr_GNU_C_Library performance: 374KB/clock
00,525 StrnglenTRAVERSED: 204449363 bytes
00,526 LinesEncountered: 2459508
00,527 Railgun_hits/Railgun_clocks: 0/574
00,528 Railgun performance: 347KB/clock
00,529 StrnglenTRAVERSED: 204449363 bytes
00,530 LinesEncountered: 2459508
00,531 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/513
00,532 Railgun_Quadruplet performance: 389KB/clock
00,533 StrnglenTRAVERSED: 204449363 bytes
00,534
00,535 Searching for Pattern('amazing',7bytes) into String(206908949bytes) line-by-line ...
00,536
00,537 LinesEncountered: 2459508
00,538 strstr_Microsoft_hits/strstr_Microsoft_clocks: 19/1230

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,539 strstr_Microsoft performance: 162KB/clock
00,540 StrnglenTRAVERSED: 204432134 bytes
00,541 LinesEncountered: 2459508
00,542 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 19/610
00,543 strstr_GNU_C_Library performance: 327KB/clock
00,544 StrnglenTRAVERSED: 204432134 bytes
00,545 LinesEncountered: 2459508
00,546 Railgun_hits/Railgun_clocks: 19/574
00,547 Railgun performance: 347KB/clock
00,548 StrnglenTRAVERSED: 204432134 bytes
00,549 LinesEncountered: 2459508
00,550 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 19/586
00,551 Railgun_Quadruplet performance: 340KB/clock
00,552 StrnglenTRAVERSED: 204432134 bytes
00,553
00,554 Searching for Pattern('underdog',8bytes) into String(206908949bytes) line-by-line ...
00,555
00,556 LinesEncountered: 2459508
00,557 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1169
00,558 strstr_Microsoft performance: 170KB/clock
00,559 StrnglenTRAVERSED: 204449185 bytes
00,560 LinesEncountered: 2459508
00,561 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/549
00,562 strstr_GNU_C_Library performance: 363KB/clock
00,563 StrnglenTRAVERSED: 204449185 bytes
00,564 LinesEncountered: 2459508
00,565 Railgun_hits/Railgun_clocks: 0/570
00,566 Railgun performance: 350KB/clock
00,567 StrnglenTRAVERSED: 204449185 bytes
00,568 LinesEncountered: 2459508
00,569 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/531
00,570 Railgun_Quadruplet performance: 376KB/clock
00,571 StrnglenTRAVERSED: 204449185 bytes
00,572
00,573 Searching for Pattern('superdog',8bytes) into String(206908949bytes) line-by-line ...
00,574
00,575 LinesEncountered: 2459508
00,576 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1212
00,577 strstr_Microsoft performance: 164KB/clock
00,578 StrnglenTRAVERSED: 204449441 bytes
00,579 LinesEncountered: 2459508
00,580 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/595
00,581 strstr_GNU_C_Library performance: 335KB/clock
00,582 StrnglenTRAVERSED: 204449441 bytes
00,583 LinesEncountered: 2459508
00,584 Railgun_hits/Railgun_clocks: 0/573
00,585 Railgun performance: 348KB/clock
00,586 StrnglenTRAVERSED: 204449441 bytes
00,587 LinesEncountered: 2459508

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,588 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/573
00,589 Railgun_Quadruplet performance: 348KB/clock
00,590 StrnglenTRAVERSED: 204449441 bytes
00,591
00,592 Searching for Pattern('participants',12bytes) into String(206908949bytes) line-by-line ...
00,593
00,594 LinesEncountered: 2459508
00,595 strstr_Microsoft_hits/strstr_Microsoft_clocks: 8/1137
00,596 strstr_Microsoft performance: 175KB/clock
00,597 StrnglenTRAVERSED: 204441500 bytes
00,598 LinesEncountered: 2459508
00,599 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 8/535
00,600 strstr_GNU_C_Library performance: 373KB/clock
00,601 StrnglenTRAVERSED: 204441500 bytes
00,602 LinesEncountered: 2459508
00,603 Railgun_hits/Railgun_clocks: 8/559
00,604 Railgun performance: 357KB/clock
00,605 StrnglenTRAVERSED: 204441500 bytes
00,606 LinesEncountered: 2459508
00,607 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 8/499
00,608 Railgun_Quadruplet performance: 400KB/clock
00,609 StrnglenTRAVERSED: 204441500 bytes
00,610
00,611 Searching for Pattern('skilllessness',12bytes) into String(206908949bytes) line-by-line ...
00,612
00,613 LinesEncountered: 2459508
00,614 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1212
00,615 strstr_Microsoft performance: 164KB/clock
00,616 StrnglenTRAVERSED: 204449441 bytes
00,617 LinesEncountered: 2459508
00,618 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/593
00,619 strstr_GNU_C_Library performance: 336KB/clock
00,620 StrnglenTRAVERSED: 204449441 bytes
00,621 LinesEncountered: 2459508
00,622 Railgun_hits/Railgun_clocks: 0/565
00,623 Railgun performance: 353KB/clock
00,624 StrnglenTRAVERSED: 204449441 bytes
00,625 LinesEncountered: 2459508
00,626 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/558
00,627 Railgun_Quadruplet performance: 357KB/clock
00,628 StrnglenTRAVERSED: 204449441 bytes
00,629
00,630 Searching for Pattern('I should have known',19bytes) into String(206908949bytes) line-by-line ...
00,631
00,632 LinesEncountered: 2459508
00,633 strstr_Microsoft_hits/strstr_Microsoft_clocks: 0/1126
00,634 strstr_Microsoft performance: 177KB/clock
00,635 StrnglenTRAVERSED: 204449346 bytes
00,636 LinesEncountered: 2459508

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
00,637 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 0/534
00,638 strstr_GNU_C_Library performance: 373KB/clock
00,639 StrnglenTRAVERSED: 204449346 bytes
00,640 LinesEncountered: 2459508
00,641 Railgun_hits/Railgun_clocks: 0/537
00,642 Railgun performance: 371KB/clock
00,643 StrnglenTRAVERSED: 204449346 bytes
00,644 LinesEncountered: 2459508
00,645 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 0/470
00,646 Railgun_Quadruplet performance: 424KB/clock
00,647 StrnglenTRAVERSED: 204449346 bytes
00,648
00,649 Searching for Pattern('human consciousness',19bytes) into String(206908949bytes) line-by-line ...
00,650
00,651 LinesEncountered: 2459508
00,652 strstr_Microsoft_hits/strstr_Microsoft_clocks: 32/1200
00,653 strstr_Microsoft performance: 166KB/clock
00,654 StrnglenTRAVERSED: 204422699 bytes
00,655 LinesEncountered: 2459508
00,656 strstr_GNU_C_Library_hits/strstr_GNU_C_Library_clocks: 32/576
00,657 strstr_GNU_C_Library performance: 346KB/clock
00,658 StrnglenTRAVERSED: 204422699 bytes
00,659 LinesEncountered: 2459508
00,660 Railgun_hits/Railgun_clocks: 32/543
00,661 Railgun performance: 367KB/clock
00,662 StrnglenTRAVERSED: 204422699 bytes
00,663 LinesEncountered: 2459508
00,664 Railgun_Quadruplet_hits/Railgun_Quadruplet_clocks: 32/522
00,665 Railgun_Quadruplet performance: 382KB/clock
00,666 StrnglenTRAVERSED: 204422699 bytes
00,667
00,668 Doing Search for 8x2 Patterns into String(206908949bytes) as-one-line ...
00,669 Found ('an') 1987797 time(s), Railgun performance: 461KB/clock
00,670 Found ('to') 1076629 time(s), Railgun performance: 495KB/clock
00,671 Found ('TDK') 0 time(s), Railgun performance: 859KB/clock
00,672 Found ('the') 2114180 time(s), Railgun performance: 585KB/clock
00,673 Found ('fast') 5945 time(s), Railgun performance: 922KB/clock
00,674 Found ('easy') 5191 time(s), Railgun performance: 985KB/clock
00,675 Found ('grmb1') 0 time(s), Railgun performance: 1287KB/clock
00,676 Found ('email') 1 time(s), Railgun performance: 1167KB/clock
00,677 Found ('pasting') 2 time(s), Railgun performance: 1603KB/clock
00,678 Found ('amazing') 323 time(s), Railgun performance: 1603KB/clock
00,679 Found ('underdog') 4 time(s), Railgun performance: 1836KB/clock
00,680 Found ('superdog') 0 time(s), Railgun performance: 1836KB/clock
00,681 Found ('participants') 147 time(s), Railgun performance: 2525KB/clock
00,682 Found ('skilllessness') 0 time(s), Railgun performance: 2149KB/clock
00,683 Found ('I should have known') 1 time(s), Railgun performance: 2126KB/clock
00,684 Found ('human consciousness') 519 time(s), Railgun performance: 2557KB/clock
00,685 Railgun 8x2 i.e. average performance: 1085KB/clock
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,686 */
00,687
00,688 /*
00,689 PUBLIC      _Railgun_Quadruplet
00,690 ; Function compile flags: /Ogtpy
00,691 TEXT      SEGMENT
00,692 tv701 = -1056                      ; size = 4
00,693 tv544 = -1056                      ; size = 4
00,694 $T5950 = -1056                    ; size = 4
00,695 Quadruplet3rd$ = -1052           ; size = 4
00,696 _countSTATIC$ = -1052           ; size = 4
00,697 _pbPattern$GSCopy$ = -1048       ; size = 4
00,698 tv698 = -1044                    ; size = 4
00,699 Quadruplet2nd$ = -1044           ; size = 4
00,700 tv569 = -1040                    ; size = 4
00,701 _ulHashPattern$ = -1040          ; size = 4
00,702 tv392 = -1036                    ; size = 4
00,703 Quadruplet4th$ = -1036           ; size = 4
00,704 _pbTargetMax$ = -1032           ; size = 4
00,705 _bm_bc$ = -1028                  ; size = 1024
00,706 __$ArrayPad$ = -4                ; size = 4
00,707 _pbTarget$ = 8                    ; size = 4
00,708 _pbPattern$ = 12                  ; size = 4
00,709 _cbTarget$ = 16                    ; size = 4
00,710 _cbPattern$ = 20                  ; size = 4
00,711 _Railgun_Quadruplet PROC
00,712
00,713 ; 1001 : {
00,714
00,715 00420      81 ec 20 04 00
00,716 00          sub     esp, 1056      ; 00000420H
00,717 00426      a1 00 00 00 00      mov     eax, DWORD PTR __security_cookie
00,718 0042b      33 c4              xor     eax, esp
00,719 0042d      89 84 24 1c 04      mov     DWORD PTR __$ArrayPad$(esp+1056), eax
00,720 00 00
00,721
00,722 ; 1002 :      char * pbTargetMax = pbTarget + cbTarget;
00,723
00,724 00434      8b 94 24 2c 04
00,725 00 00      mov     edx, DWORD PTR _cbTarget$(esp+1052)
00,726 0043b      53              push    ebx
00,727 0043c      8b 9c 24 28 04
00,728 00 00      mov     ebx, DWORD PTR _pbTarget$(esp+1056)
00,729 00443      55              push    ebp
00,730
00,731 ; 1003 :      register unsigned long ulHashPattern;
00,732 ; 1004 :      unsigned long ulHashTarget;
00,733 ; 1005 :      unsigned long count;
00,734 ; 1006 :      unsigned long countSTATIC;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,735 ; 1007 : // unsigned long countRemainder;
00,736 ; 1008 :
00,737 ; 1009 : /*
00,738 ; 1010 :     const unsigned char SINGLET = *(char *) (pbPattern);
00,739 ; 1011 :     const unsigned long Quadruplet2nd = SINGLET<<8;
00,740 ; 1012 :     const unsigned long Quadruplet3rd = SINGLET<<16;
00,741 ; 1013 :     const unsigned long Quadruplet4th = SINGLET<<24;
00,742 ; 1014 : */
00,743 ; 1015 :     unsigned char SINGLET;
00,744 ; 1016 :     unsigned long Quadruplet2nd;
00,745 ; 1017 :     unsigned long Quadruplet3rd;
00,746 ; 1018 :     unsigned long Quadruplet4th;
00,747 ; 1019 :
00,748 ; 1020 :     unsigned long AdvanceHopperGrass;
00,749 ; 1021 :
00,750 ; 1022 :     long i; //BMH needed
00,751 ; 1023 :     int a, j, bm_bc[ASIZE]; //BMH needed
00,752 ; 1024 :     unsigned char ch; //BMH needed
00,753 ; 1025 : //     unsigned char lastch, firstch; //BMH needed
00,754 ; 1026 :
00,755 ; 1027 :     if (cbPattern > cbTarget)
00,756 :
00,757 00444      8b ac 24 38 04
00,758 00 00      mov     ebp, DWORD PTR _cbPattern$(esp+1060)
00,759 0044b      56      push    esi
00,760 0044c      8b b4 24 34 04
00,761 00 00      mov     esi, DWORD PTR _pbPattern$(esp+1064)
00,762 00453      8d 04 13 lea     eax, DWORD PTR [ebx+edx]
00,763 00456      89 74 24 14 mov     DWORD PTR _pbPattern$GSCopy$(esp+1068), esi
00,764 0045a      89 44 24 24 mov     DWORD PTR _pbTargetMax$(esp+1068), eax
00,765 0045e      3b ea     cmp     ebp, edx
00,766 00460      76 1a     jbe     SHORT $LN36@Bailgun_Qu
00,767 :
00,768 ; 1028 :     return(NULL);
00,769 :
00,770 00462      5e      pop     esi
00,771 00463      5d      pop     ebp
00,772 00464      33 c0     xor     eax, eax
00,773 00466      5b      pop     ebx
00,774 :
00,775 ; 1151 : } //if (cbTarget<961)
00,776 ; 1152 : } //if ( cbPattern<4)
00,777 ; 1153 : }
00,778 :
00,779 00467      8b 8c 24 1c 04
00,780 00 00      mov     ecx, DWORD PTR __ArrayPad$(esp+1056)
00,781 0046e      33 cc     xor     ecx, esp
00,782 00470      e8 00 00 00 00 call    @_security_check_cookie@4
00,783 00475      81 c4 20 04 00

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,784      00          add     esp, 1056      ; 00000420H
00,785      0047b      c3          ret      0
00,786 $LN36@Railgun_Qu:
00,787      0047c      57          push     edi
00,788
00,789 ; 1029 :
00,790 ; 1030 : // Doesn't work when cbPattern = 1
00,791 ; 1031 : // The next IF-fragment works very well with cbPattern>1, OBVIOUSLY IT MUST BE UNROLLED(but crippled with less functionality) SINCE either cbPattern=2 or cbPattern=3!
00,792 ; 1032 : if ( cbPattern<4) { // This IF makes me unhappy: it slows down from 390KB/clock to 367KB/clock for 'fast' pattern. This fragment(for 2..3 pattern lengths) is needed because I need a function different than
strchr but sticking to strstr i.e. lengths above 1 are to be handled.
00,793
00,794      0047d      83 fd 04      cmp      ebp, 4
00,795      00480      0f 83 8b 00 00      jae      $LN35@Railgun_Qu
00,796      00
00,797
00,798 ; 1033 :      pbTarget = pbTarget+cbPattern;
00,799 ; 1034 :      ulHashPattern = ( (*char *) (pbPattern))<<8 ) + *(pbPattern+(cbPattern-1));
00,800
00,801      00486      0f be 0e      movsx     ecx, BYTE PTR [esi]
00,802      00489      0f be 7c 2e ff      movsx     edi, BYTE PTR [esi+ebp-1]
00,803
00,804 ; 1083 :      }
00,805
00,806      0048e      b8 02 00 00 00      mov      eax, 2
00,807      00493      2b c5          sub      eax, ebp
00,808      00495      c1 e1 08      shl      ecx, 8
00,809      00498      89 44 24 1c      mov      DWORD PTR tv698[esp+1072], eax
00,810      0049c      b8 01 00 00 00      mov      eax, 1
00,811      004a1      03 dd          add      ebx, ebp
00,812      004a3      03 f9          add      edi, ecx
00,813      004a5      2b c5          sub      eax, ebp
00,814      004a7      89 44 24 10      mov      DWORD PTR tv701[esp+1072], eax
00,815      004ab      eb 03 8d 49 00      npad     5
00,816 $LL34@Railgun_Qu:
00,817
00,818 ; 1035 :      countSTATIC = cbPattern-2;
00,819 ; 1036 :
00,820 ; 1037 :      for ( ;; )
00,821 ; 1038 :      {
00,822 ; 1039 :          // The line below gives for 'cbPattern'>=1:
00,823 ; 1040 :          // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/543
00,824 ; 1041 :          // Karp_Rabin_Kaze_4_OCTETS performance: 372KB/clock
00,825 ; 1042 : /*
00,826 ; 1043 :          if ( (ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1)) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
00,827 ; 1044 :              return((long)(pbTarget-cbPattern));
00,828 ; 1045 : ~/
00,829 ; 1046 :
00,830 ; 1047 :          // The fragment below gives for 'cbPattern'>=2:
00,831 ; 1048 :          // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/546

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,832 ; 1049 :      // Karp_Rabin_Kaze_4_OCTETS performance: 370KB/clock
00,833 ; 1050 :
00,834 ; 1051 :      if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern)) << 8 ) + *(pbTarget-1) ) {
00,835
00,836 004b0      0f be 54 18 ff      movsx     edx, BYTE PTR [eax+ebx-1]
00,837 004b5      0f be 4b ff      movsx     ecx, BYTE PTR [ebx-1]
00,838 004b9      c1 e2 08      shl      edx, 8
00,839 004bc      03 d1      add      edx, ecx
00,840 004be      3b fa      cmp      edi, edx
00,841 004c0      75 2d      jne      SHORT $LN64@Railgun_Qu
00,842
00,843 ; 1052 :      count = countSTATIC;
00,844
00,845 004c2      8d 45 fe      lea      eax, DWORD PTR [ebp-2]
00,846
00,847 ; 1053 :      while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
00,848
00,849 004c5      85 c0      test     eax, eax
00,850 004c7      74 14      je       SHORT $LN58@Railgun_Qu
00,851
00,852 ; 1052 :      count = countSTATIC;
00,853
00,854 004c9      8b 54 24 1c      mov      edx, DWORD PTR tv698[esp+1072]
00,855 004cd      46      inc      esi
00,856 004ce      8d 4c 1a ff      lea      ecx, DWORD PTR [edx+ebx-1]
00,857 $LL31@Railgun_Qu:
00,858
00,859 ; 1053 :      while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
00,860
00,861 004d2      8a 16      mov      dl, BYTE PTR [esi]
00,862 004d4      3a 11      cmp      dl, BYTE PTR [ecx]
00,863 004d6      75 0b      jne      SHORT $LN30@Railgun_Qu
00,864
00,865 ; 1054 :      count--;
00,866
00,867 004d8      41      inc      ecx
00,868 004d9      46      inc      esi
00,869 004da      48      dec      eax
00,870 004db      75 f5      jne      SHORT $LL31@Railgun_Qu
00,871 $LN58@Railgun_Qu:
00,872
00,873 ; 1055 :      }
00,874 ; 1056 :      if ( count == 0 ) return((pbTarget-cbPattern));
00,875
00,876 004dd      8b c3      mov      eax, ebx
00,877 004df      2b c5      sub      eax, ebp
00,878
00,879 ; 1082 :      return(NULL);
00,880

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,881 004e1 eb 15 jmp SHORT $LN65@Railgun_Qu
00,882 $LN30@Railgun_Qu:
00,883
00,884 ; 1055 : }
00,885 ; 1056 : if ( count == 0) return((pbTarget-cbPattern));
00,886
00,887 004e3 85 c0 test eax, eax
00,888 004e5 74 f6 je SHORT $LN58@Railgun_Qu
00,889 004e7 8b 74 24 18 mov esi, DWORD PTR _pbPattern$SGSCopy$(esp+1072)
00,890 004eb 8b 44 24 10 mov eax, DWORD PTR tv701[esp+1072]
00,891 $LN64@Railgun_Qu:
00,892
00,893 ; 1057 : }
00,894 ; 1058 :
00,895 ; 1059 : // The fragment below gives for 'cbPattern'>=2:
00,896 ; 1060 : // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/554
00,897 ; 1061 : // Karp_Rabin_Kaze_4_OCTETS performance: 364KB/clock
00,898 ; 1062 : /*
00,899 ; 1063 : if ( ulHashPattern == ( (* (char *) (pbTarget-cbPattern)) << 8 ) + *(pbTarget-1) ) {
00,900 ; 1064 : count = countSTATIC>>2;
00,901 ; 1065 : countRemainder = countSTATIC % 4;
00,902 ; 1066 :
00,903 ; 1067 : while ( count && *((unsigned long *) (pbPattern+1+((count-1)<<2))) == *((unsigned long *) (pbTarget-cbPattern+1+((count-1)<<2))) ) {
00,904 ; 1068 : count--;
00,905 ; 1069 : }
00,906 ; 1070 : //if (count == 0) { // Disastrous degradation only from this line(317KB/clock when 1+2x4+2+1 bytes pattern: 'skillessness'; 312KB/clock when 1+1x4+2+1 bytes pattern: 'underdog'), otherwise 368KB/clock.
00,907 ; 1071 : while ( countRemainder && *((char *) (pbPattern+1+(countSTATIC-countRemainder))) == *((char *) (pbTarget-cbPattern+1+(countSTATIC-countRemainder))) ) {
00,908 ; 1072 : countRemainder--;
00,909 ; 1073 : }
00,910 ; 1074 : //if ( countRemainder == 0) return((long)(pbTarget-cbPattern));
00,911 ; 1075 : if ( count+countRemainder == 0) return((long)(pbTarget-cbPattern));
00,912 ; 1076 : //}
00,913 ; 1077 : }
00,914 ; 1078 : ~/
00,915 ; 1079 :
00,916 ; 1080 : pbTarget++;
00,917
00,918 004ef 43 inc ebx
00,919
00,920 ; 1081 : if (pbTarget > pbTargetMax)
00,921
00,922 004f0 3b 5c 24 28 cmp ebx, DWORD PTR _pbTargetMax$(esp+1072)
00,923 004f4 76 ba jbe SHORT $LL34@Railgun_Qu
00,924 $LN5@Railgun_Qu:
00,925
00,926 ; 1149 : }
00,927 ; 1150 : return(NULL);
00,928
00,929 004f6 33 c0 xor eax, eax

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,930 $LN65@Railgun_Qu:
00,931
00,932 ; 1151 : } //if (cbTarget<961)
00,933 ; 1152 : } //if ( cbPattern<4)
00,934 ; 1153 : }
00,935
00,936 004f8      8b 8c 24 2c 04
00,937 00 00      mov     ecx, DWORD PTR __$ArrayPad$(esp+1072)
00,938 004ff      5f      pop     edi
00,939 00500      5e      pop     esi
00,940 00501      5d      pop     ebp
00,941 00502      5b      pop     ebx
00,942 00503      33 cc   xor     ecx, esp
00,943 00505      e8 00 00 00 00 call    @_security_check_cookie@4
00,944 0050a      81 c4 20 04 00
00,945 00      add     esp, 1056      ; 00000420H
00,946 00510      c3      ret     0
00,947 $LN35@Railgun_Qu:
00,948
00,949 ; 1084 : } else { //if ( cbPattern<4)
00,950 ; 1085 : if (cbTarget<961) // This value is arbitrary(don't know how exactly), it ensures(at least must) better performance than 'Boyer_Moore_Horspool'.
00,951
00,952 00511      81 fa c1 03 00
00,953 00      cmp     edx, 961      ; 000003c1H
00,954 00517      0f 83 dd 00 00
00,955 00      jae     $LN26@Railgun_Qu
00,956
00,957 ; 1086 : {
00,958 ; 1087 :      pbTarget = pbTarget+cbPattern;
00,959 ; 1088 :      ulHashPattern = *(unsigned long *) (pbPattern);
00,960
00,961 0051d      8b 16      mov     edx, DWORD PTR [esi]
00,962
00,963 ; 1089 :      countSTATIC = cbPattern-1;
00,964 ; 1090 :
00,965 ; 1091 :      //SINGLET = *(char *) (pbPattern);
00,966 ; 1092 :      SINGLET = ulHashPattern & 0xFF;
00,967 ; 1093 :      Quadruplet2nd = SINGLET<<8;
00,968
00,969 0051f      0f b6 c2   movzx   eax, dl
00,970 00522      8b c8      mov     ecx, eax
00,971 00524      c1 e1 08   shl     ecx, 8
00,972 00527      89 4c 24 1c mov     DWORD PTR _Quadruplet2nd$(esp+1072), ecx
00,973
00,974 ; 1094 :      Quadruplet3rd = SINGLET<<16;
00,975
00,976 0052b      8b c8      mov     ecx, eax
00,977 0052d      03 dd      add     ebx, ebp
00,978 0052f      c1 e1 10   shl     ecx, 16      ; 00000010H

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

00,979 00532      89 44 24 10      mov     DWORD PTR tv544[esp+1072], eax
00,980
00,981 ; 1095 :      Quadruplet4th = SINGLET<<24;
00,982
00,983 00536      c1 e0 18  shl     eax, 24                ; 00000018H
00,984 00539      89 9c 24 34 04
00,985 00 00      mov     DWORD PTR _pbTarget$[esp+1068], ebx
00,986 00540      89 54 24 20      mov     DWORD PTR _ulHashPattern$[esp+1072], edx
00,987 00544      89 4c 24 14      mov     DWORD PTR _Quadruplet3rd$[esp+1072], ecx
00,988 00548      89 44 24 24      mov     DWORD PTR _Quadruplet4th$[esp+1072], eax
00,989 0054c      8d 64 24 00      npad    4
00,990 $LL250Railgun_Qu:
00,991
00,992 ; 1096 :
00,993 ; 1097 :      for ( ;; )
00,994 ; 1098 :      {
00,995 ; 1099 :      AdvanceHopperGrass = 0;
00,996 ; 1100 :      ulHashTarget = *(unsigned long *) (pbTarget-cbPattern);
00,997
00,998 00550      8b cb      mov     ecx, ebx
00,999 00552      2b cd      sub     ecx, ebp
01,000 00554      8b 01      mov     eax, DWORD PTR [ecx]
01,001 00556      33 f6      xor     esi, esi
01,002
01,003 ; 1101 :
01,004 ; 1102 :      if ( ulHashPattern == ulHashTarget ) { // Three unnecessary comparisons here, but 'AdvanceHopperGrass' must be calculated - it has a higher priority.
01,005
01,006 00558      3b d0      cmp     edx, eax
01,007 0055a      75 4a      jne     SHORT $LN230Railgun_Qu
01,008
01,009 ; 1103 :      count = countSTATIC;
01,010
01,011 0055c      8d 45 ff  lea     eax, DWORD PTR [ebp-1]
01,012
01,013 ; 1104 :      while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,014
01,015 0055f      85 c0      test    eax, eax
01,016 00561      0f 84 76 ff ff  je     $LN580Railgun_Qu
01,017 ff
01,018
01,019 ; 1103 :      count = countSTATIC;
01,020
01,021 00567      8b 54 24 18      mov     edx, DWORD PTR _pbPattern$GSCopy$[esp+1072]
01,022 0056b      42      inc     edx
01,023 0056c      8d 79 01  lea     edi, DWORD PTR [ecx+1]
01,024 0056f      90      npad    1
01,025 $LL220Railgun_Qu:
01,026
01,027 ; 1104 :      while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,028
01,029 00570      8a 0f          mov     cl, BYTE PTR [edi]
01,030 00572      38 0a          cmp     BYTE PTR [edx], cl
01,031 00574      75 26          jne     SHORT $LN21@Railgun_Qu
01,032
01,033 ; 1105 :          if ( countSTATIC==AdvanceHopperGrass+count && SINGLET != *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) AdvanceHopperGrass++;
01,034
01,035 00576      8d 1c 06   lea     ebx, DWORD PTR [esi+eax]
01,036 00579      8d 4d ff   lea     ecx, DWORD PTR [ebp-1]
01,037 0057c      3b cb          cmp     ecx, ebx
01,038 0057e      75 0a          jne     SHORT $LN62@Railgun_Qu
01,039 00580      0f be 0f   movsx   ecx, BYTE PTR [edi]
01,040 00583      39 4c 24 10   cmp     DWORD PTR tv544[esp+1072], ecx
01,041 00587      74 01          je      SHORT $LN62@Railgun_Qu
01,042 00589      46          inc     esi
01,043 $LN62@Railgun_Qu:
01,044
01,045 ; 1104 :          while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,046
01,047 0058a      8b 9c 24 34 04   mov     ebx, DWORD PTR _pbTarget$[esp+1068]
01,048 00 00
01,049
01,050 ; 1106 :          count--;
01,051
01,052 00591      42          inc     edx
01,053 00592      47          inc     edi
01,054 00593      48          dec     eax
01,055 00594      0f 84 43 ff ff   je      $LN58@Railgun_Qu
01,056 ff
01,057
01,058 ; 1104 :          while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,059
01,060 0059a      eb d4          jmp     SHORT $LL22@Railgun_Qu
01,061 $LN21@Railgun_Qu:
01,062
01,063 ; 1107 :          }
01,064 ; 1108 :          if ( count == 0 ) return((pbTarget-cbPattern));
01,065
01,066 0059c      85 c0          test    eax, eax
01,067 0059e      0f 84 39 ff ff   je      $LN58@Railgun_Qu
01,068 ff
01,069
01,070 ; 1109 :          } else { // The goal here: to avoid memory accesses by stressing the registers.
01,071
01,072 005a4      eb 36          jmp     SHORT $LN15@Railgun_Qu
01,073 $LN23@Railgun_Qu:
01,074
01,075 ; 1110 :          if ( Quadruplet2nd != (ulHashTarget & 0x0000FF00) ) {
01,076
01,077

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,077 005a6 8b d0      mov     edx, eax
01,078 005a8 81 e2 00 ff 00      and     edx, 65280 ; 0000ff00H
01,079 00      and     edx, 65280 ; 0000ff00H
01,080 005ae 39 54 24 1c      cmp     DWORD PTR _Quadruplet2nd$[esp+1072], edx
01,081 005b2 74 28      je      SHORT $LN15@Railgun_Qu
01,082
01,083 ; 1111 :      AdvanceHopperGrass++;
01,084 ; 1112 :      if ( Quadruplet3rd != (ulHashTarget & 0x00FF0000) ) {
01,085
01,086 005b4 8b c8      mov     ecx, eax
01,087 005b6 81 e1 00 00 ff      and     ecx, 16711680 ; 00ff0000H
01,088 00      and     ecx, 16711680 ; 00ff0000H
01,089 005bc be 01 00 00 00      mov     esi, 1
01,090 005c1 39 4c 24 14      cmp     DWORD PTR _Quadruplet3rd$[esp+1072], ecx
01,091 005c5 74 15      je      SHORT $LN15@Railgun_Qu
01,092
01,093 ; 1113 :      AdvanceHopperGrass++;
01,094 ; 1114 :      if ( Quadruplet4th != (ulHashTarget & 0xFF000000) ) AdvanceHopperGrass++;
01,095
01,096 005c7 25 00 00 00 ff      and     eax, -16777216 ; ff000000H
01,097 005cc be 02 00 00 00      mov     esi, 2
01,098 005d1 39 44 24 24      cmp     DWORD PTR _Quadruplet4th$[esp+1072], eax
01,099 005d5 74 05      je      SHORT $LN15@Railgun_Qu
01,100 005d7 be 03 00 00 00      mov     esi, 3
01,101 $LN15@Railgun_Qu:
01,102
01,103 ; 1115 :      }
01,104 ; 1116 :      }
01,105 ; 1117 :      }
01,106 ; 1118 :
01,107 ; 1119 :      AdvanceHopperGrass++;
01,108 ; 1120 :
01,109 ; 1121 :      pbTarget = pbTarget + AdvanceHopperGrass;
01,110
01,111 005dc 8d 5c 33 01      lea     ebx, DWORD PTR [ebx+esi+1]
01,112 005e0 89 9c 24 34 04      mov     DWORD PTR _pbTarget$[esp+1068], ebx
01,113 00 00
01,114
01,115 ; 1122 :      if (pbTarget > pbTargetMax)
01,116
01,117 005e7 3b 5c 24 28      cmp     ebx, DWORD PTR _pbTargetMax$[esp+1072]
01,118 005eb 0f 87 05 ff ff      ja      $LN5@Railgun_Qu
01,119 ff
01,120
01,121 ; 1123 :      return(NULL);
01,122 ; 1124 :      }
01,123
01,124 005f1 8b 54 24 20      mov     edx, DWORD PTR _ulHashPattern$[esp+1072]
01,125 005f5 e9 56 ff ff ff      jmp     $LL25@Railgun_Qu

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,126 $LN26@Railgun_Qu:
01,127
01,128 ; 1125 : } else { //if (cbTarget<961)
01,129 ; 1126 :     countSTATIC = cbPattern-2;
01,130
01,131 005fa      8d 45 fe  lea     eax, DWORD PTR [ebp-2]
01,132 005fd      89 44 24 14     mov     DWORD PTR _countSTATIC$[esp+1072], eax
01,133
01,134 ; 1127 :     /* Preprocessing */
01,135 ; 1128 :     for (a=0; a < ASIZE; a++) hm_bc[a]=cbPattern;
01,136
01,137 00601      8b c5             mov     eax, ebp
01,138 00603      b9 00 01 00 00     mov     ecx, 256             ; 00000100H
01,139 00608      8d 7c 24 2c     lea     edi, DWORD PTR _hm_bc$[esp+1072]
01,140 0060c      f3 ab             rep stosd
01,141
01,142 ; 1129 :     for (j=0; j < cbPattern-1; j++) hm_bc[pbPattern[j]]=cbPattern-j-1;
01,143
01,144 0060e      8d 4d ff  lea     ecx, DWORD PTR [ebp-1]
01,145 00611      33 c0             xor     eax, eax
01,146 00613      85 c9             test    ecx, ecx
01,147 00615      74 1a             je      SHORT $LN7@Railgun_Qu
01,148 00617      8d 7d ff  lea     edi, DWORD PTR [ebp-1]
01,149 0061a      8d 9b 00 00 00     npad     6
01,150 00
01,151 $LN9@Railgun_Qu:
01,152 00620      0f be 0c 06     movsx   ecx, BYTE PTR [esi+eax]
01,153 00624      89 7c 8c 2c     mov     DWORD PTR _hm_bc$[esp+ecx*4+1072], edi
01,154 00628      40              inc     eax
01,155 00629      8d 4d ff  lea     ecx, DWORD PTR [ebp-1]
01,156 0062c      4f              dec     edi
01,157 0062d      3b c1             cmp     eax, ecx
01,158 0062f      72 ef             jb      SHORT $LN9@Railgun_Qu
01,159 $LN7@Railgun_Qu:
01,160
01,161 ; 1130 :
01,162 ; 1131 :     /* Searching */
01,163 ; 1132 :     //lastch=pbPattern[cbPattern-1];
01,164 ; 1133 :     //firstch=pbPattern[0];
01,165 ; 1134 :     i=0;
01,166 ; 1135 :     while (i <= cbTarget-cbPattern) {
01,167
01,168 00631      0f be 44 2e ff     movsx   eax, BYTE PTR [esi+ebp-1]
01,169 00636      33 ff             xor     edi, edi
01,170 00638      2b d5             sub     edx, ebp
01,171 0063a      89 54 24 10     mov     DWORD PTR $T5950$[esp+1072], edx
01,172 0063e      8d 54 2b ff     lea     edx, DWORD PTR [ebx+ebp-1]
01,173 00642      89 54 24 20     mov     DWORD PTR tv569$[esp+1072], edx
01,174 00646      89 44 24 24     mov     DWORD PTR tv392$[esp+1072], eax

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,175 0064a 8d 9b 00 00 00
01,176 00 npad 6
01,177 $LL6@Railgun_Qu:
01,178
01,179 ; 1136 : ch=pbTarget[i+cbPattern-1];
01,180 ; 1137 : //if (ch ==lastch)
01,181 ; 1138 : //if (memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) OUTPUT(i);
01,182 ; 1139 : //if (ch == lastch && pbTarget[i] == firstch && memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e.
to perform a slower check only when the target "looks like".
01,183 ; 1140 : if (ch == pbPattern[cbPattern-1] && pbTarget[i] == pbPattern[0])
01,184
01,185 00650 8b 4c 24 20 mov ecx, DWORD PTR tv569[esp+1072]
01,186 00654 0f b6 2c 39 movzx ebp, BYTE PTR [ecx+edi]
01,187 00658 3b 6c 24 24 cmp ebp, DWORD PTR tv392[esp+1072]
01,188 0065c 75 2f jne SHORT $LN1@Railgun_Qu
01,189 0065e 8a 14 1f mov dl, BYTE PTR [edi+ebx]
01,190 00661 3a 16 cmp dl, BYTE PTR [esi]
01,191 00663 75 28 jne SHORT $LN1@Railgun_Qu
01,192
01,193 ; 1141 : {
01,194 ; 1142 : count = countSTATIC;
01,195
01,196 00665 8b 44 24 14 mov eax, DWORD PTR _countSTATIC$[esp+1072]
01,197
01,198 ; 1143 : while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *)(&pbTarget[i]+1+(countSTATIC-count)) ) {
01,199
01,200 00669 85 c0 test eax, eax
01,201 0066b 74 10 je SHORT $LN51@Railgun_Qu
01,202
01,203 ; 1141 : {
01,204 ; 1142 : count = countSTATIC;
01,205
01,206 0066d 46 inc esi
01,207 0066e 8d 4c 1f 01 lea ecx, DWORD PTR [edi+ebx+1]
01,208 $LL3@Railgun_Qu:
01,209
01,210 ; 1143 : while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *)(&pbTarget[i]+1+(countSTATIC-count)) ) {
01,211
01,212 00672 8a 16 mov dl, BYTE PTR [esi]
01,213 00674 3a 11 cmp dl, BYTE PTR [ecx]
01,214 00676 75 0d jne SHORT $LN2@Railgun_Qu
01,215
01,216 ; 1144 : count--;
01,217
01,218 00678 41 inc ecx
01,219 00679 46 inc esi
01,220 0067a 48 dec eax
01,221 0067b 75 f5 jne SHORT $LL3@Railgun_Qu
01,222 $LN51@Railgun_Qu:

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,223
01,224 ; 1145 :      }
01,225 ; 1146 :      if ( count == 0) return(pbTarget+i);
01,226
01,227 0067d      8d 04 1f lea      eax, DWORD PTR [edi+ebx]
01,228 00680      e9 73 fe ff ff      jmp      $LN65@Railgun_Qu
01,229 $LN2@Railgun_Qu:
01,230 00685      85 c0          test     eax, eax
01,231 00687      74 f4          je       SHORT $LN51@Railgun_Qu
01,232 00689      8b 74 24 18     mov      esi, DWORD PTR _pbPattern$GSCopy$(esp+1072)
01,233 $LN1@Railgun_Qu:
01,234
01,235 ; 1147 :      }
01,236 ; 1148 :      i+=bm_bc[ch];
01,237
01,238 0068d      03 7c ac 2c     add      edi, DWORD PTR _bm_bc$(esp+ebp*4+1072)
01,239 00691      3b 7c 24 10     cmp      edi, DWORD PTR $T5950(esp+1072)
01,240 00695      76 b9          jbe      SHORT $LL6@Railgun_Qu
01,241
01,242 ; 1130 :
01,243 ; 1131 :      /* Searching ~/
01,244 ; 1132 :      //lastch=pbPattern[chPattern-1];
01,245 ; 1133 :      //firstch=pbPattern[0];
01,246 ; 1134 :      i=0;
01,247 ; 1135 :      while (i <= chTarget-chPattern) {
01,248
01,249 00697      e9 5a fe ff ff      jmp      $LN5@Railgun_Qu
01,250 _Railgun_Quadruplet ENDP
01,251 */
01,252
01,253 /*
01,254 On Intel T3400 CPU: IntelC 11.1 mashes MicrosoftC 13.10.3077 with ((1072-681)/681)*100=57%, le-le.
01,255
01,256 strstr_SHORT-SHOWDOWN_Microsoft.exe:
01,257 Doing Search for 8x2 Patterns into String(206908949bytes) as-one-line ...
01,258 Found ('an') 1987797 time(s), Railgun performance: 300KB/clock
01,259 Found ('to') 1076629 time(s), Railgun performance: 300KB/clock
01,260 Found ('TDK') 0 time(s), Railgun performance: 496KB/clock
01,261 Found ('the') 2114180 time(s), Railgun performance: 403KB/clock
01,262 Found ('fast') 5945 time(s), Railgun performance: 585KB/clock
01,263 Found ('easy') 5191 time(s), Railgun performance: 614KB/clock
01,264 Found ('grmb1') 0 time(s), Railgun performance: 805KB/clock
01,265 Found ('email') 1 time(s), Railgun performance: 716KB/clock
01,266 Found ('pasting') 2 time(s), Railgun performance: 990KB/clock
01,267 Found ('amazing') 323 time(s), Railgun performance: 990KB/clock
01,268 Found ('underdog') 4 time(s), Railgun performance: 1069KB/clock
01,269 Found ('superdog') 0 time(s), Railgun performance: 1167KB/clock
01,270 Found ('participants') 147 time(s), Railgun performance: 1433KB/clock
01,271 Found ('skilllessness') 0 time(s), Railgun performance: 1422KB/clock

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,272 Found ('I should have known') 1 time(s), Railgun performance: 1433KB/clock
01,273 Found ('human consciousness') 519 time(s), Railgun performance: 1820KB/clock
01,274 Railgun 8x2 i.e. average performance: 681KB/clock
01,275
01,276 strstr_SHORT-SHOWDOWN_Intel.exe:
01,277 Doing Search for 8x2 Patterns into String(206908949bytes) as-one-line ...
01,278 Found ('an') 1987797 time(s), Railgun performance: 460KB/clock
01,279 Found ('to') 1076629 time(s), Railgun performance: 477KB/clock
01,280 Found ('TDK') 0 time(s), Railgun performance: 859KB/clock
01,281 Found ('the') 2114180 time(s), Railgun performance: 585KB/clock
01,282 Found ('fast') 5945 time(s), Railgun performance: 918KB/clock
01,283 Found ('easy') 5191 time(s), Railgun performance: 990KB/clock
01,284 Found ('grmbl') 0 time(s), Railgun performance: 1167KB/clock
01,285 Found ('email') 1 time(s), Railgun performance: 1167KB/clock
01,286 Found ('pasting') 2 time(s), Railgun performance: 1603KB/clock
01,287 Found ('amazing') 323 time(s), Railgun performance: 1603KB/clock
01,288 Found ('underdog') 4 time(s), Railgun performance: 1603KB/clock
01,289 Found ('superdog') 0 time(s), Railgun performance: 1836KB/clock
01,290 Found ('participants') 147 time(s), Railgun performance: 2126KB/clock
01,291 Found ('skilllessness') 0 time(s), Railgun performance: 2149KB/clock
01,292 Found ('I should have known') 1 time(s), Railgun performance: 2126KB/clock
01,293 Found ('human consciousness') 519 time(s), Railgun performance: 2557KB/clock
01,294 Railgun 8x2 i.e. average performance: 1072KB/clock
01,295 */
01,296
01,297 // Revision 2:
01,298 /*
01,299 DANNII MINOGUE:
01,300 Where do we go now?
01,301 I don't know
01,302 Innocence over
01,303 Fading fast
01,304 ...
01,305 You're still promising perfection, perfection
01,306 With empty words
01,307 With empty words
01,308 With empty words
01,309 With empty words
01,310 And it's hard to break a habit
01,311 You're lost inside it
01,312 ...
01,313 A moment of coldness
01,314 Cuts through me (cuts through me)
01,315 I've tried to remember
01,316 Why I don't leave (I don't leave)
01,317 And you're the cause of my confusion
01,318 Closing down the way I feel
01,319 How come I don't see so clearly, so clearly
01,320 ...

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,321 */
01,322
01,323 #include <stdio.h>
01,324 #include <stdlib.h>
01,325 #include <string.h>
01,326 #include <time.h>
01,327
01,328 #ifndef NULL
01,329 #define NULL ((void*)0)
01,330 #endif
01,331
01,332 clock_t clocks1, clocks2;
01,333 clock_t clocks3, clocks4;
01,334 double TotalRoughSearchTime = 0;
01,335
01,336 long Railgunhits=0;
01,337
01,338 #define ASIZE 256
01,339
01,340 // ### Boyer-Moore-Horspool algorithm [
01,341 long HORSPOOL(y, x, n, m)
01,342     char *y, *x;
01,343     long n;
01,344     int m;
01,345     {
01,346         long i;
01,347         int a, j, bm_bc[ASIZE];
01,348         unsigned char ch, lastch;
01,349
01,350         /* Preprocessing */
01,351         for (a=0; a < ASIZE; a++) bm_bc[a]=m;
01,352         for (j=0; j < m-1; j++) bm_bc[x[j]]=m-j-1;
01,353
01,354         /* Searching */
01,355         lastch=x[m-1];
01,356         i=0;
01,357         while (i <= n-m) {
01,358             ch=y[i+m-1];
01,359             if (ch == lastch)
01,360                 //if (memcmp(&y[i],x,m-1) == 0) OUTPUT(i);
01,361                 if (memcmp(&y[i],x,m-1) == 0) return(i);
01,362             i+=bm_bc[ch];
01,363         }
01,364         return(-1);
01,365     }
01,366 long Boyer_Moore_Horspool_Kaze(y, x, n, m) // m>=2
01,367     char *y, *x;
01,368     long n;
01,369     int m;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,370 {
01,371     long i;
01,372     int a, j, bm_bc[ASIZE];
01,373     unsigned char ch;
01,374     unsigned char lastch;
01,375     unsigned char firstch;
01,376
01,377     unsigned long count;
01,378     unsigned long countSTATIC;
01,379
01,380     /* Preprocessing */
01,381     for (a=0; a < ASIZE; a++) bm_bc[a]=m;
01,382     for (j=0; j < m-1; j++) bm_bc[x[j]]=m-j-1;
01,383
01,384     /* Searching */
01,385     lastch=x[m-1];
01,386     firstch=x[0];
01,387     i=0;
01,388     countSTATIC = m-2;
01,389     while (i <= n-m) {
01,390         ch=y[i+m-1];
01,391         //if (ch ==lastch)
01,392             //if (memcmp(&y[i],x,m-1) == 0) OUTPUT(i);
01,393 // Below line gives: 315KB/clock
01,394         //if (ch ==lastch && y[i] == firstch && memcmp(&y[i],x,m-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e. to perform a slower check only when the
target "looks like".
01,395 // Below line gives: 328KB/clock
01,396 //         if (ch == lastch && y[i] == firstch && memcmp(&y[i+1],&x[0+1],m-1-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e. to perform a slower check
only when the target "looks like".
01,397
01,398         if (ch == lastch && y[i] == firstch)
01,399             {
01,400                 count = countSTATIC;
01,401                 while ( count && *(char *) (x+1+(countSTATIC-count)) == *(char *) (&y[i+1+(countSTATIC-count)) ) {
01,402                     count--;
01,403                 }
01,404                 if ( count == 0) return(i);
01,405             }
01,406
01,407         i+=bm_bc[ch];
01,408     }
01,409     return(-1);
01,410 }
01,411
01,412 // ### Boyer-Moore-Horspool algorithm ]
01,413
01,414
01,415 // ### Brute force 'Dummy' algorithm [
01,416     long Brute_Force_Dummy(char *y, char *x, long n, int m) {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,417     long i, j;
01,418
01,419     /* Searching */
01,420     for (i=0; i <= n-m; i++) {
01,421         j=0;
01,422         while (j < m && y[i+j] == x[j]) j++;
01,423         if (j >= m) return(i);
01,424     }
01,425     return(-1);
01,426 }
01,427 // ### Brute force 'Dummy' algorithm ]
01,428
01,429
01,430 // ### Karp-Rabin algorithm [
01,431 #define REHASH(a, b, h) (((h) - (a)*d) << 1) + (b))
01,432 long Karp_Rabin(char *y, char *x, long n, int m) {
01,433     int d, hx, hy, i, j;
01,434
01,435     /* Preprocessing */
01,436     /* computes d = 2^(m-1) with
01,437        the left-shift operator */
01,438     for (d = i = 1; i < m; ++i)
01,439         d = (d<<1);
01,440
01,441     for (hy = hx = i = 0; i < m; ++i) {
01,442         hx = ((hx<<1) + x[i]);
01,443         hy = ((hy<<1) + y[i]);
01,444     }
01,445
01,446     /* Searching */
01,447     j = 0;
01,448     while (j <= n-m) {
01,449         if (hx == hy && memcmp(x, y + j, m) == 0) return(j);
01,450         hy = REHASH(y[j], y[j + m], hy);
01,451         ++j;
01,452     }
01,453     return(-1);
01,454 }
01,455 // ### Karp-Rabin algorithm ]
01,456
01,457
01,458 // ### Karp-Rabin-Kaze algorithm [
01,459 char * KarpRabinKaze (char * pbTarget,
01,460     char * pbPattern,
01,461     unsigned long cbTarget,
01,462     unsigned long cbPattern)
01,463 {
01,464     unsigned int i;
01,465     char * pbTargetMax = pbTarget + cbTarget;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1++); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,466 char * pbPatternMax = pbPattern + cbPattern;
01,467 unsigned long ulBaseToPowerMod = 1;
01,468 register unsigned long ulHashPattern = 0;
01,469 unsigned long ulHashTarget = 0;
01,470 long hits = 0;
01,471 //unsigned long count;
01,472 //char * buf1;
01,473 //char * buf2;
01,474
01,475 if (cbPattern > cbTarget)
01,476     return(NULL);
01,477
01,478 // Compute the power of the left most character in base ulBase
01,479 //for (i = 1; i < cbPattern; i++) ulBaseToPowerMod = (ulBase * ulBaseToPowerMod);
01,480
01,481 // Calculate the hash function for the src (and the first dst)
01,482 while (pbPattern < pbPatternMax)
01,483 {
01,484     // Below lines give 366KB/clock for 'underdog':
01,485     //ulHashPattern = (ulHashPattern*ulBase + *pbPattern);
01,486     //ulHashTarget = (ulHashTarget*ulBase + *pbTarget);
01,487     pbPattern++;
01,488     pbTarget++;
01,489 }
01,490 // Below lines give 436KB/clock for 'underdog' + requirement pattern to be 4 chars min.:
01,491 //ulHashPattern = ( ( *(long *) (pbPattern-cbPattern) ) & 0xffffffff ) + *(pbPattern-1);
01,492 //ulHashTarget = ( ( *(long *) (pbTarget-cbPattern) ) & 0xffffffff ) + *(pbTarget-1);
01,493 // Below lines give 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,494 //ulHashPattern = ( ( *(unsigned short *) (pbPattern-cbPattern) ) ! *(pbPattern-1) );
01,495 //ulHashTarget = ( ( *(unsigned short *) (pbTarget-cbPattern) ) ! *(pbTarget-1) );
01,496 // Below lines give 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,497 //ulHashPattern = ( ( *(unsigned short *) (pbPattern-cbPattern) ) & 0xff00 ) + *(pbPattern-1);
01,498 //ulHashTarget = ( ( *(unsigned short *) (pbTarget-cbPattern) ) & 0xff00 ) + *(pbTarget-1);
01,499 // Below lines give 605KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,500 //ulHashPattern = ( ( *(unsigned short *) (pbPattern-cbPattern) ) << 8 ) + *(pbPattern-1);
01,501 //ulHashTarget = ( ( *(unsigned short *) (pbTarget-cbPattern) ) << 8 ) + *(pbTarget-1);
01,502 // Below lines give 668KB/clock for 'underdog':
01,503 ulHashPattern = ( ( *(char *) (pbPattern-cbPattern) ) << 8 ) + *(pbPattern-1);
01,504 ulHashTarget = ( ( *(char *) (pbTarget-cbPattern) ) << 8 ) + *(pbTarget-1);
01,505
01,506 // Dynamically produce hash values for the string as we go
01,507 for ( ;; )
01,508 {
01,509     if ( (ulHashPattern == ulHashTarget) && !memcmp(pbPattern-cbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
01,510         // if ( ulHashPattern == ulHashTarget ) {
01,511         //
01,512         // count = cbPattern;
01,513         // buf1 = pbPattern-cbPattern;
01,514         // buf2 = pbTarget-cbPattern;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,515 // while ( --count && *(char *)buf1 == *(char *)buf2 ) {
01,516 //     buf1 = (char *)buf1 + 1;
01,517 //     buf2 = (char *)buf2 + 1;
01,518 // }
01,519 //
01,520 // if ( *((unsigned char *)buf1) - *((unsigned char *)buf2) == 0) hits++;
01,521 // }
01,522     return((pbTarget-cbPattern));
01,523 //hits++;
01,524
01,525 if (pbTarget == pbTargetMax)
01,526     return(NULL);
01,527
01,528 // Below line gives 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,529 //ulHashTarget = ( (*unsigned short *) (pbTarget+1-cbPattern)) ! *pbTarget );
01,530 // Below line gives 436KB/clock for 'underdog' + requirement pattern to be 4 chars min.:
01,531 //ulHashTarget = ( (*long *) (pbTarget+1-cbPattern)) & 0xfffff00 ) + *pbTarget;
01,532 //; Line 696
01,533 //     movsx     esi, BYTE PTR [ebx]
01,534 //     mov     ecx, DWORD PTR [edx+1]
01,535 //     and     ecx, -256                                ; fffff00H
01,536 //     add     ecx, esi
01,537 // Below line gives 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,538 //ulHashTarget = ( (*unsigned short *) (pbTarget+1-cbPattern)) & 0xff00 ) + *pbTarget;
01,539 //; Line 691
01,540 //     movsx     esi, BYTE PTR [ebx]
01,541 //     xor     ecx, ecx
01,542 //     mov     cx, WORD PTR [edx+1]
01,543 //     and     ecx, 65280                                ; 0000ff00H
01,544 //     add     ecx, esi
01,545 // Below line gives 605KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
01,546 //ulHashTarget = ( (*unsigned short *) (pbTarget+1-cbPattern))<<8 ) + *pbTarget;
01,547 // Below line gives 668KB/clock for 'underdog':
01,548 ulHashTarget = ( (*char *) (pbTarget+1-cbPattern))<<8 ) + *pbTarget;
01,549 //; Line 718
01,550 //     movsx     ecx, BYTE PTR [eax+1]
01,551 //     movsx     edx, BYTE PTR [ebp]
01,552 //     shl     ecx, 8
01,553 //     add     ecx, edx
01,554 // Below line gives 366KB/clock for 'underdog':
01,555 //ulHashTarget = (ulHashTarget - *(pbTarget-cbPattern)*ulBaseToPowerMod)*ulBase + *pbTarget;
01,556 pbTarget++;
01,557 }
01,558 }
01,559 // ### Karp-Rabin-Kaze algorithm ]
01,560
01,561
01,562 // ### Mix(2in1) of Karp-Rabin & Boyer-Moore-Horspool algorithm [
01,563 // Caution: For better speed the case 'if (cbPattern==1)' was removed, so Pattern must be longer than 1 char.

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,564 char * Railgun_Quadruplet (char * pbTarget,
01,565     char * pbPattern,
01,566     unsigned long cbTarget,
01,567     unsigned long cbPattern)
01,568 {
01,569     char * pbTargetMax = pbTarget + cbTarget;
01,570     register unsigned long ulHashPattern;
01,571     unsigned long ulHashTarget;
01,572     unsigned long count;
01,573     unsigned long countSTATIC;
01,574 // unsigned long countRemainder;
01,575
01,576 /*
01,577     const unsigned char SINGLET = *(char *) (pbPattern);
01,578     const unsigned long Quadruplet2nd = SINGLET<<8;
01,579     const unsigned long Quadruplet3rd = SINGLET<<16;
01,580     const unsigned long Quadruplet4th = SINGLET<<24;
01,581 */
01,582     unsigned char SINGLET;
01,583     unsigned long Quadruplet2nd;
01,584     unsigned long Quadruplet3rd;
01,585     unsigned long Quadruplet4th;
01,586
01,587     unsigned long AdvanceHopperGrass;
01,588
01,589     long i; //BMH needed
01,590     int a, j, bm_bc[ASIZE]; //BMH needed
01,591     unsigned char ch; //BMH needed
01,592 // unsigned char lastch, firstch; //BMH needed
01,593
01,594     if (cbPattern > cbTarget)
01,595         return(NULL);
01,596
01,597 // Doesn't work when cbPattern = 1
01,598 // The next IF-fragment works very well with cbPattern>1, OBVIOUSLY IT MUST BE UNROLLED(but crippled with less functionality) SINCE either cbPattern=2 or cbPattern=3!
01,599 if ( cbPattern<4) { // This IF makes me unhappy: it slows down from 390KB/clock to 367KB/clock for 'fast' pattern. This fragment(for 2..3 pattern lengths) is needed because I need a function different than strchr but
sticking to strstr i.e. lengths above 1 are to be handled.
01,600     pbTarget = pbTarget+cbPattern;
01,601     ulHashPattern = ( *(char *) (pbPattern)<<8 ) + *(pbPattern+(cbPattern-1));
01,602     countSTATIC = cbPattern-2;
01,603
01,604     for ( ;; )
01,605     {
01,606         // The line below gives for 'cbPattern'=1:
01,607         // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/543
01,608         // Karp_Rabin_Kaze_4_OCTETS performance: 372KB/clock
01,609 /*
01,610         if ( (ulHashPattern == ( *(char *) (pbTarget-cbPattern)<<8 ) + *(pbTarget-1)) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
01,611             return((long) (pbTarget-cbPattern));

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,612 */
01,613
01,614 // The fragment below gives for 'cbPattern'>=2:
01,615 // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/546
01,616 // Karp_Rabin_Kaze_4_OCTETS performance: 370KB/clock
01,617
01,618 if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) {
01,619     count = countSTATIC;
01,620     while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,621         count--;
01,622     }
01,623     if ( count == 0 ) return((pbTarget-cbPattern));
01,624 }
01,625
01,626 // The fragment below gives for 'cbPattern'>=2:
01,627 // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/554
01,628 // Karp_Rabin_Kaze_4_OCTETS performance: 364KB/clock
01,629 /*
01,630 if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) {
01,631     count = countSTATIC>>2;
01,632     countRemainder = countSTATIC % 4;
01,633
01,634     while ( count && *(unsigned long *) (pbPattern+1+((count-1)<<2)) == *(unsigned long *) (pbTarget-cbPattern+1+((count-1)<<2)) ) {
01,635         count--;
01,636     }
01,637 //if (count == 0) { // Disastrous degradation only from this line(317KB/clock when 1+2x4+2+1 bytes pattern: 'skilllessness'; 312KB/clock when 1+1x4+2+1 bytes pattern: 'underdog'), otherwise 368KB/clock.
01,638     while ( countRemainder && *(char *) (pbPattern+1+(countSTATIC-countRemainder)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-countRemainder)) ) {
01,639         countRemainder--;
01,640     }
01,641     //if ( countRemainder == 0 ) return((long)(pbTarget-cbPattern));
01,642     if ( count+countRemainder == 0 ) return((long)(pbTarget-cbPattern));
01,643     //}
01,644 }
01,645 */
01,646
01,647 pbTarget++;
01,648 if (pbTarget > pbTargetMax)
01,649     return(NULL);
01,650 }
01,651 } else { //if ( cbPattern<4)
01,652 if (cbTarget<961) // This value is arbitrary(don't know how exactly), it ensures(at least must) better performance than 'Boyer_Moore_Horspool'.
01,653 {
01,654     pbTarget = pbTarget+cbPattern;
01,655     ulHashPattern = *(unsigned long *) (pbPattern);
01,656     countSTATIC = cbPattern-1;
01,657
01,658 //SINGLET = *(char *) (pbPattern);
01,659 SINGLET = ulHashPattern & 0xFF;
01,660 Quadruplet2nd = SINGLET<<8;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,661   Quadruplet3rd = SINGLET<<16;
01,662   Quadruplet4th = SINGLET<<24;
01,663
01,664   for ( ;; )
01,665   {
01,666   AdvanceHopperGrass = 0;
01,667   ulHashTarget = *(unsigned long *) (pbTarget-cbPattern);
01,668
01,669       if ( ulHashPattern == ulHashTarget ) { // Three unnecessary comparisons here, but 'AdvanceHopperGrass' must be calculated - it has a higher priority.
01,670           count = countSTATIC;
01,671           while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,672               if ( countSTATIC==AdvanceHopperGrass+count && SINGLET != *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) AdvanceHopperGrass++;
01,673               count--;
01,674           }
01,675           if ( count == 0 ) return((pbTarget-cbPattern));
01,676       } else { // The goal here: to avoid memory accesses by stressing the registers.
01,677       if ( Quadruplet2nd != (ulHashTarget & 0x0000FF00) ) {
01,678           AdvanceHopperGrass++;
01,679           if ( Quadruplet3rd != (ulHashTarget & 0x00FF0000) ) {
01,680               AdvanceHopperGrass++;
01,681               if ( Quadruplet4th != (ulHashTarget & 0xFF000000) ) AdvanceHopperGrass++;
01,682           }
01,683       }
01,684   }
01,685
01,686   AdvanceHopperGrass++;
01,687
01,688   pbTarget = pbTarget + AdvanceHopperGrass;
01,689   if (pbTarget > pbTargetMax)
01,690       return(NULL);
01,691   }
01,692 } else { //if (cbTarget<961)
01,693     countSTATIC = cbPattern-2;
01,694     /* Preprocessing */
01,695     for (a=0; a < ASIZE; a++) km_bc[a]=cbPattern;
01,696     for (j=0; j < cbPattern-1; j++) km_bc[pbPattern[j]]=cbPattern-j-1;
01,697
01,698     /* Searching */
01,699     //lastch=pbPattern[cbPattern-1];
01,700     //firstch=pbPattern[0];
01,701     i=0;
01,702     while (i <= cbTarget-cbPattern) {
01,703         ch=pbTarget[i+cbPattern-1];
01,704         //if (ch ==lastch)
01,705             //if (memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) OUTPUT(i);
01,706             //if (ch == lastch && pbTarget[i] == firstch && memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e. to
perform a slower check only when the target "looks like".
01,707         if (ch == pbPattern[cbPattern-1] && pbTarget[i] == pbPattern[0])
01,708             {

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,709         count = countSTATIC;
01,710         while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (&pbTarget[i]+1+(countSTATIC-count)) ) {
01,711             count--;
01,712         }
01,713         if ( count == 0) return(pbTarget+i);
01,714     }
01,715     i+=bm_bc[ch];
01,716 }
01,717 return(NULL);
01,718 } //if (cbTarget<961)
01,719 } //if ( cbPattern<4)
01,720 }
01,721 // ### Mix(2in1) of Karp-Rabin & Boyer-Moore-Horspool algorithm ]
01,722
01,723
01,724 // ### Mix(2in1) of Karp-Rabin & Boyer-Moore-Horspool algorithm [
01,725 // Caution: For better speed the case 'if (cbPattern==1)' was removed, so Pattern must be longer than 1 char.
01,726 char * Railgun (char * pbTarget,
01,727                 char * pbPattern,
01,728                 unsigned long cbTarget,
01,729                 unsigned long cbPattern)
01,730 {
01,731     char * pbTargetMax = pbTarget + cbTarget;
01,732     register unsigned long ulHashPattern;
01,733     unsigned long ulHashTarget;
01,734     unsigned long count;
01,735     unsigned long countSTATIC, countRemainder;
01,736
01,737     long i; //BMH needed
01,738     int a, j, bm_bc[ASIZE]; //BMH needed
01,739     unsigned char ch; //BMH needed
01,740 //     unsigned char lastch, firstch; //BMH needed
01,741
01,742     if (cbPattern > cbTarget)
01,743         return(NULL);
01,744
01,745     countSTATIC = cbPattern-2;
01,746
01,747 // Doesn't work when cbPattern = 1
01,748 if (cbTarget<961) // This value is arbitrary(don't know how exactly), it ensures(at least must) better performance than 'Boyer_Moore_Horspool'.
01,749 {
01,750     pbTarget = pbTarget+cbPattern;
01,751     ulHashPattern = ( (*(char *) (pbPattern))<<8 ) + *(pbPattern+(cbPattern-1));
01,752
01,753     for ( ;; )
01,754     {
01,755         // The line below gives for 'cbPattern'>=1:
01,756         // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/543
01,757         // Karp_Rabin_Kaze_4_OCTETS performance: 372KB/clock

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,758 /*
01,759     if ( (ulHashPattern == ( (*char *) (pbTarget-cbPattern)) << 8 ) + *(pbTarget-1) ) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int) cbPattern) )
01,760         return((long)(pbTarget-cbPattern));
01,761 */
01,762
01,763 // The fragment below gives for 'cbPattern'>=2:
01,764 // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/546
01,765 // Karp_Rabin_Kaze_4_OCTETS performance: 370KB/clock
01,766
01,767 if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern)) << 8 ) + *(pbTarget-1) ) {
01,768     count = countSTATIC;
01,769     while ( count && (*char *) (pbPattern+1+(countSTATIC-count)) == (*char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,770         count--;
01,771     }
01,772     if ( count == 0 ) return((pbTarget-cbPattern));
01,773 }
01,774
01,775 // The fragment below gives for 'cbPattern'>=2:
01,776 // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/554
01,777 // Karp_Rabin_Kaze_4_OCTETS performance: 364KB/clock
01,778 /*
01,779     if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern)) << 8 ) + *(pbTarget-1) ) {
01,780         count = countSTATIC >> 2;
01,781         countRemainder = countSTATIC % 4;
01,782
01,783         while ( count && *(unsigned long *) (pbPattern+1+((count-1)<<2)) == *(unsigned long *) (pbTarget-cbPattern+1+((count-1)<<2)) ) {
01,784             count--;
01,785         }
01,786 //if (count == 0) { // Disastrous degradation only from this line(317KB/clock when 1+2x4+2+1 bytes pattern: 'skilllessness'; 312KB/clock when 1+1x4+2+1 bytes pattern: 'underdog'), otherwise 368KB/clock.
01,787     while ( countRemainder && (*char *) (pbPattern+1+(countSTATIC-countRemainder)) == (*char *) (pbTarget-cbPattern+1+(countSTATIC-countRemainder)) ) {
01,788         countRemainder--;
01,789     }
01,790     //if ( countRemainder == 0 ) return((long)(pbTarget-cbPattern));
01,791     if ( count+countRemainder == 0 ) return((long)(pbTarget-cbPattern));
01,792     //}
01,793 }
01,794 */
01,795
01,796 pbTarget++;
01,797 if (pbTarget > pbTargetMax)
01,798     return(NULL);
01,799 }
01,800 }
01,801 else
01,802 {
01,803     /* Preprocessing */
01,804     for (a=0; a < ASIZE; a++) bm_bc[a]=cbPattern;
01,805     for (j=0; j < cbPattern-1; j++) bm_bc[pbPattern[j]]=cbPattern-j-1;
01,806

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,807  /* Searching */
01,808  //lastch=pbPattern[cbPattern-1];
01,809  //firstch=pbPattern[0];
01,810  i=0;
01,811  while (i <= cbTarget-cbPattern) {
01,812      ch=pbTarget[i+cbPattern-1];
01,813      //if (ch ==lastch)
01,814          //if (memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) OUTPUT(i);
01,815          //if (ch == lastch && pbTarget[i] == firstch && memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e. to
perform a slower check only when the target "looks like".
01,816          if (ch == pbPattern[cbPattern-1] && pbTarget[i] == pbPattern[0])
01,817              {
01,818                  count = countSTATIC;
01,819                  while ( count && *(char *)(&pbPattern+1+(countSTATIC-count)) == *(char *)(&pbTarget[i]+1+(countSTATIC-count)) ) {
01,820                      count--;
01,821                  }
01,822                  if ( count == 0) return(pbTarget+i);
01,823              }
01,824          i+=bm_bc[ch];
01,825      }
01,826      return(NULL);
01,827 }
01,828 }
01,829 // ### Mix(2in1) of Karp-Rabin & Boyer-Moore-Horspool algorithm ]
01,830
01,831
01,832 // ### Railgun_totalhits [
01,833 char * Railgun_totalhits (char * pbTarget,
01,834     char * pbPattern,
01,835     unsigned long cbTarget,
01,836     unsigned long cbPattern)
01,837 {
01,838     char * pbTargetMax = pbTarget + cbTarget;
01,839     register unsigned long ulHashPattern;
01,840     unsigned long ulHashTarget;
01,841     unsigned long count;
01,842     unsigned long countSTATIC, countRemainder;
01,843
01,844     long i; //BMH needed
01,845     int a, j, bm_bc[ASIZE]; //BMH needed
01,846     unsigned char ch; //BMH needed
01,847 //     unsigned char lastch, firstch; //BMH needed
01,848
01,849     if (cbPattern > cbTarget)
01,850         return(NULL);
01,851
01,852     countSTATIC = cbPattern-2;
01,853
01,854 // Doesn't work when cbPattern = 1

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```
01,855 if (cbTarget<961) // This value is arbitrary(don't know how exactly), it ensures(at least must) better performance than 'Boyer_Moore_Horspool'.
01,856 {
01,857     pbTarget = pbTarget+cbPattern;
01,858     ulHashPattern = ( (*char *) (pbPattern))<<8 ) + *(pbPattern+(cbPattern-1));
01,859
01,860 for ( ;; )
01,861 {
01,862     // The line below gives for 'cbPattern'>=1:
01,863     // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/543
01,864     // Karp_Rabin_Kaze_4_OCTETS performance: 372KB/clock
01,865 /*
01,866     if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
01,867         return((long)(pbTarget-cbPattern));
01,868 */
01,869
01,870     // The fragment below gives for 'cbPattern'>=2:
01,871     // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/546
01,872     // Karp_Rabin_Kaze_4_OCTETS performance: 370KB/clock
01,873
01,874     if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) {
01,875         count = countSTATIC;
01,876         while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-count)) ) {
01,877             count--;
01,878         }
01,879         if ( count == 0) Railgunhits++; //return((pbTarget-cbPattern));
01,880     }
01,881
01,882     // The fragment below gives for 'cbPattern'>=2:
01,883     // Karp_Rabin_Kaze_4_OCTETS_hits/Karp_Rabin_Kaze_4_OCTETS_clocks: 4/554
01,884     // Karp_Rabin_Kaze_4_OCTETS performance: 364KB/clock
01,885 /*
01,886     if ( ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) {
01,887         count = countSTATIC>>2;
01,888         countRemainder = countSTATIC % 4;
01,889
01,890         while ( count && *(unsigned long *) (pbPattern+1+((count-1)<<2)) == *(unsigned long *) (pbTarget-cbPattern+1+((count-1)<<2)) ) {
01,891             count--;
01,892         }
01,893         //if (count == 0) { // Disastrous degradation only from this line(317KB/clock when 1+2x4+2+1 bytes pattern: 'skilllessness'; 312KB/clock when 1+1x4+2+1 bytes pattern: 'underdog'), otherwise 368KB/clock.
01,894         while ( countRemainder && *(char *) (pbPattern+1+(countSTATIC-countRemainder)) == *(char *) (pbTarget-cbPattern+1+(countSTATIC-countRemainder)) ) {
01,895             countRemainder--;
01,896         }
01,897         //if ( countRemainder == 0) return((long)(pbTarget-cbPattern));
01,898         if ( count+countRemainder == 0) return((long)(pbTarget-cbPattern));
01,899         //}
01,900     }
01,901 */
01,902
01,903     pbTarget++;
```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

01,904         if (pbTarget > pbTargetMax)
01,905             return(NULL);
01,906     }
01,907 }
01,908 else
01,909 {
01,910     /* Preprocessing */
01,911     for (a=0; a < ASIZE; a++) bm_bc[a]=cbPattern;
01,912     for (j=0; j < cbPattern-1; j++) bm_bc[pbPattern[j]]=cbPattern-j-1;
01,913
01,914     /* Searching */
01,915     //lastch=pbPattern[cbPattern-1];
01,916     //firstch=pbPattern[0];
01,917     i=0;
01,918     while (i <= cbTarget-cbPattern) {
01,919         ch=pbTarget[i+cbPattern-1];
01,920         //if (ch ==lastch)
01,921             //if (memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) OUTPUT(i);
01,922             //if (ch == lastch && pbTarget[i] == firstch && memcmp(&pbTarget[i],pbPattern,cbPattern-1) == 0) return(i); // Kaze: The idea(to prevent execution of slower 'memcmp') is borrowed from Karp-Rabin i.e. to
perform a slower check only when the target "looks like".
01,923             if (ch == pbPattern[cbPattern-1] && pbTarget[i] == pbPattern[0])
01,924                 {
01,925                     count = countSTATIC;
01,926                     while ( count && *(char *) (pbPattern+1+(countSTATIC-count)) == *(char *) (&pbTarget[i]+1+(countSTATIC-count)) ) {
01,927                         count--;
01,928                     }
01,929                     if ( count == 0) Railgunhits++; //return(pbTarget+i);
01,930                 }
01,931             i+=bm_bc[ch];
01,932     }
01,933     return(NULL);
01,934 }
01,935 }
01,936 // ### Railgun_totalhits ]
01,937
01,938
01,939 // ### Karp-Rabin-Kaze_BOOSTED algorithm [
01,940 char * KarpRabinKaze_BOOSTED (char * pbTarget,
01,941     char * pbPattern,
01,942     unsigned long cbTarget,
01,943     unsigned long cbPattern)
01,944 {
01,945     char * pbTargetMax = pbTarget + cbTarget;
01,946     register unsigned long ulHashPattern;
01,947     unsigned long ulHashTarget;
01,948
01,949     if (cbPattern > cbTarget)
01,950         return(NULL);
01,951

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



```

01,952         pbTarget = pbTarget+cbPattern;
01,953         ulHashPattern = ( (*char *) (pbPattern))<<8 ) + *(pbPattern+(cbPattern-1));
01,954
01,955     for ( ;; )
01,956     {
01,957         // Kaze: The idea(FAILED) here is to add an additional(second) layer in order to prevent execution of slower hash calculation(i.e. first layer) which(hash) prevents execution of even slower 'memcmp'.
01,958         // The line below gives: 314KB/clock
01,959         //if ( *pbPattern == *(char *) (pbTarget-cbPattern) && (ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
01,960         // The line below gives: 370KB/clock
01,961         if ( (ulHashPattern == ( (*char *) (pbTarget-cbPattern))<<8 ) + *(pbTarget-1) ) && !memcmp(pbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
01,962             return((pbTarget-cbPattern));
01,963
01,964         pbTarget++;
01,965         if (pbTarget > pbTargetMax)
01,966             return(NULL);
01,967     }
01,968 }
01,969 // ### Karp-Rabin-Kaze_BOOSTED algorithm ]
01,970
01,971 char * strstr_Microsoft (
01,972     const char * str1,
01,973     const char * str2
01,974 )
01,975 {
01,976     char *cp = (char *) str1;
01,977     char *s1, *s2;
01,978
01,979     if ( !*str2 )
01,980         return((char *)str1);
01,981
01,982     while (*cp)
01,983     {
01,984         s1 = cp;
01,985         s2 = (char *) str2;
01,986
01,987         while ( *s1 && *s2 && !(*s1-*s2) )
01,988             s1++, s2++;
01,989
01,990         if (!*s2)
01,991             return(cp);
01,992
01,993         cp++;
01,994     }
01,995     return(NULL);
01,996 }
01,997
01,998 char *
01,999 strstr_GNU_C_Library (phystack, pneedle)
02,000     const char *phystack;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,001     const char *pneedle;
02,002 {
02,003     const unsigned char *haystack, *needle;
02,004     char b;
02,005     const unsigned char *rneedle;
02,006
02,007     haystack = (const unsigned char *) phystack;
02,008
02,009     if ((b = *(needle = (const unsigned char *) pneedle)))
02,010     {
02,011         char c;
02,012         haystack--;           /* possible ANSI violation */
02,013
02,014         {
02,015             char a;
02,016             do
02,017                 if (!(a = **haystack))
02,018                     goto ret0;
02,019             while (a != b);
02,020         }
02,021
02,022         if (!(c = **needle))
02,023             goto foundneedle;
02,024         ++needle;
02,025         goto jin;
02,026
02,027         for (;;)
02,028         {
02,029             {
02,030                 char a;
02,031                 if (!0)
02,032                 jin:{
02,033                     if ((a = **haystack) == c)
02,034                         goto crest;
02,035                 }
02,036                 else
02,037                     a = **haystack;
02,038                 do
02,039                 {
02,040                     for (; a != b; a = **haystack)
02,041                     {
02,042                         if (!a)
02,043                             goto ret0;
02,044                         if ((a = **haystack) == b)
02,045                             break;
02,046                         if (!a)
02,047                             goto ret0;
02,048                     }
02,049                 }

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,050     while ((a = **haystack) != c);
02,051     }
02,052     crest:
02,053     {
02,054         char a;
02,055         {
02,056             const unsigned char *rhaystack;
02,057             if (*(rhaystack = haystack-- + 1) == (a = *(rneedle = needle)))
02,058                 do
02,059                     {
02,060                         if (!a)
02,061                             goto foundneedle;
02,062                         if (**rhaystack != (a = **needle))
02,063                             break;
02,064                         if (!a)
02,065                             goto foundneedle;
02,066                     }
02,067                     while (**rhaystack == (a = **needle));
02,068                     needle = rneedle; /* took the register-poor aproach */
02,069                 }
02,070             if (!a)
02,071                 break;
02,072         }
02,073     }
02,074 }
02,075 foundneedle:
02,076     return (char *) haystack;
02,077 ret0:
02,078     return 0;
02,079 }
02,080
02,081 void x64toaKAZE ( /* stdcall is faster and smaller... Might as well use it for the helper. */
02,082     unsigned long long val,
02,083     char *buf,
02,084     unsigned radix,
02,085     int is_neg
02,086 )
02,087 {
02,088     char *p; /* pointer to traverse string */
02,089     char *firstdig; /* pointer to first digit */
02,090     char temp; /* temp char */
02,091     unsigned digval; /* value of digit */
02,092
02,093     p = buf;
02,094
02,095     if ( is_neg )
02,096     {
02,097         *p++ = '-'; /* negative, so output '-' and negate */
02,098         val = (unsigned long long)(- (long long)val);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,099     }
02,100
02,101     firstdig = p;          /* save pointer to first digit */
02,102
02,103     do {
02,104         digval = (unsigned) (val % radix);
02,105         val /= radix;        /* get next digit */
02,106
02,107         /* convert to ascii and store */
02,108         if (digval > 9)
02,109             *p++ = (char) (digval - 10 + 'a'); /* a letter */
02,110         else
02,111             *p++ = (char) (digval + '0');      /* a digit */
02,112     } while (val > 0);
02,113
02,114     /* We now have the digit of the number in the buffer, but in reverse
02,115        order. Thus we reverse them now. */
02,116
02,117     *p-- = '\0';            /* terminate string; p points to last digit */
02,118
02,119     do {
02,120         temp = *p;
02,121         *p = *firstdig;
02,122         *firstdig = temp; /* swap *p and *firstdig */
02,123         --p;
02,124         ++firstdig;        /* advance to next two digits */
02,125     } while (firstdig < p); /* repeat until halfway */
02,126 }
02,127
02,128 /* Actual functions just call conversion helper with neg flag set correctly,
02,129    and return pointer to buffer. */
02,130
02,131 char * _i64toaKAZE (
02,132     long long val,
02,133     char *buf,
02,134     int radix
02,135 )
02,136 {
02,137     x64toaKAZE((unsigned long long)val, buf, radix, (radix == 10 && val < 0));
02,138     return buf;
02,139 }
02,140
02,141 char * _ui64toaKAZE (
02,142     unsigned long long val,
02,143     char *buf,
02,144     int radix
02,145 )
02,146 {
02,147     x64toaKAZE(val, buf, radix, 0);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,148         return buf;
02,149 }
02,150
02,151 char * _ui64toaKAZEzerocomma (
02,152     unsigned long long val,
02,153     char *buf,
02,154     int radix
02,155 )
02,156 {
02,157     char *p;
02,158     char temp;
02,159     int txpman;
02,160     int pxnman;
02,161     x64toaKAZE(val, buf, radix, 0);
02,162     p = buf;
02,163     do {
02,164     } while (*++p != '\0');
02,165     p--; // p points to last digit
02,166         // buf points to first digit
02,167     buf[26] = 0;
02,168     txpman = 1;
02,169     pxnman = 0;
02,170     do
02,171     { if (buf <= p)
02,172         { temp = *p;
02,173             buf[26-txpman] = temp; pxnman++;
02,174             p--;
02,175             if (pxnman % 3 == 0)
02,176             { txpman++;
02,177                 buf[26-txpman] = (char) (',' );
02,178             }
02,179             }
02,180         else
02,181         { buf[26-txpman] = (char) ('0'); pxnman++;
02,182             if (pxnman % 3 == 0)
02,183             { txpman++;
02,184                 buf[26-txpman] = (char) (',' );
02,185             }
02,186             }
02,187         txpman++;
02,188     } while (txpman <= 26);
02,189     return buf;
02,190 }
02,191
02,192 char * _ui64toaKAZEcomma (
02,193     unsigned long long val,
02,194     char *buf,
02,195     int radix
02,196 )

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,197 {
02,198     char *p;
02,199     char temp;
02,200     int txpman;
02,201     int pxnman;
02,202     x64toaKAZE(val, buf, radix, 0);
02,203     p = buf;
02,204     do {
02,205     } while (*++p != '\0');
02,206     p--; // p points to last digit
02,207     // buf points to first digit
02,208     buf[26] = 0;
02,209     txpman = 1;
02,210     pxnman = 0;
02,211     while (buf <= p)
02,212     { temp = *p;
02,213       buf[26-txpman] = temp; pxnman++;
02,214       p--;
02,215       if (pxnman % 3 == 0 && buf <= p)
02,216       { txpman++;
02,217         buf[26-txpman] = (char) (',');
02,218       }
02,219       txpman++;
02,220     }
02,221     return buf+26-(txpman-1);
02,222 }
02,223
02,224 #define _WIN32_ENVIRONMENT_
02,225 // #define _POSIX_ENVIRONMENT_
02,226
02,227
02,228 #ifndef NULL
02,229 #ifdef __cplusplus
02,230 #define NULL 0
02,231 #else
02,232 #define NULL ((void*)0)
02,233 #endif
02,234 #endif
02,235
02,236 // To do #1: Put this 31 in MAXw1: 'int MAXw1 = 31;'
02,237 // To do #2: No need of flushing unsorted words to file: make backup[] array
02,238 //         instead of writing. And mostly sort 26 times!
02,239 // HEAVY BUG in r.7: unsigned long Hill(unsigned long n)
02,240 //         is NOT identical with
02,241 //         unsigned long GRMBLhill[32]; // 00 not used, only 01..31
02,242 //         BECAUSE DUMBEST DUMB Array GRMBLhill expects 'int' not
02,243 //         'unsigned long' !!!
02,244
02,245 #include <stdio.h>

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,246 #include <ctype.h>
02,247 #include <time.h>
02,248 #if defined( WIN32_ENVIRONMENT_)
02,249 #include <io.h> // needed for Windows' 'lseeki64' and 'telli64'
02,250 //Above line must be commented in order to compile with Intel C compiler: an error "can't find io.h" occurs.
02,251 #else
02,252 #endif /* defined( WIN32_ENVIRONMENT_) */
02,253
02,254 typedef unsigned char char_t;
02,255 typedef char_t *string;
02,256
02,257
02,258 int main( int argc, char *argv[])
02,259 {
02,260 FILE *fp_in;
02,261 FILE *fp_inLINE;
02,262 FILE *fp_outLINE;
02,263 int Bozan;
02,264 long ThunderwithL, ThunderwithR;
02,265 char *Strng;
02,266 long Strnglen;
02,267 long StrnglenTRAVERSED;
02,268 long TRUEStrnglenTRAVERSED;
02,269 char Pattern[20*2000]; // skilllessness=12 human consciousness=19 I should have known=19
02,270 // In the East, enlightenment is described as a state of ultimate=62
02,271 int Patternlen;
02,272 long LinesEncountered=0;
02,273 long LinesWritten=0;
02,274 long LinesWrittenChunk=0;
02,275 long BruteForceDummyhits=0;
02,276 long KarpRabinKazehits=0;
02,277 long KarpRabinKaze_BOOSTEDhits=0;
02,278 long Karp_Rabin_Kaze_4_OCTETShits=0;
02,279 long Karp_Rabin_Kaze_4_OCTETShits_DOUBLET=0;
02,280 long KarpRabinhits=0;
02,281 long HORSPPOOLhits=0;
02,282 long HORSPPOOL_Kazehits=0;
02,283 long strstrMicrosofthits=0;
02,284 long strstrGNUCLibraryhits=0;
02,285
02,286     char workK[1024*128];
02,287     long workKoffset = -1;
02,288
02,289     unsigned long long FilesLEN;
02,290     unsigned long k, LINE10len, wrdlen, TRUEwrdlen;
02,291 unsigned long LINE10xgramlen;
02,292     unsigned long long NumberOfLines; // rev. 12+
02,293     unsigned long long NumberOfWords; // rev. 12+
02,294     unsigned long size_in; // rev. 12+

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,295 #if defined(_WIN32_ENVIRONMENT_)
02,296     unsigned long long size_inLINESIXFOUR;
02,297 #else
02,298     size_t size_inLINESIXFOUR;
02,299 #endif /* defined(_WIN32_ENVIRONMENT_) */
02,300
02,301     char LINE10[257]; // 000..255, 256 = 0
02,302 char LINE10xgram[257]; // 000..255, 256 = 0
02,303     char ZEROS[4]; // 0..3, 0 = 0, 1 = 0, 2 = 0, 3 = 0
02,304     char CRdLFa[2]; // 0..1, 0 = 13, 1 = 10
02,305     char workbyte;
02,306
02,307 //int i, j; //BUG fixed in r.1+
02,308 unsigned long long int i, j; //BUG fixed in r.1+
02,309
02,310 char *DumboBox[8][2] = { "an\0", "to\0",
02,311                          "TDK\0", "the\0",
02,312                          "fast\0", "easy\0",
02,313                          "grmb1\0", "email\0",
02,314                          "pasting\0", "amazing\0",
02,315                          "underdog\0", "superdog\0",
02,316                          "participants\0", "skillessness\0",
02,317                          "I should have known\0", "human consciousness\0"
02,318 };
02,319
02,320 long FoundIn;
02,321 char *FoundInPTR;
02,322
02,323 char *ChunkName = { ".1000\0"};
02,324 char l1ToaDigits[27]; // 9,223,372,036,854,775,807: 1(sign or carry)+19(digits)+1('\0')+6(,)
02,325 char TheWholeName[256]; // 0..30, 31 = 0
02,326 char *DelimiterUnderscore = ".\0";
02,327 char *PADext = ".PAD\0";
02,328 char SpacePADs[200];
02,329 int PositionToBePadded = 0;
02,330 int PAGODAorder = 5;
02,331 int UnderscoresToSkip;
02,332 int ChunkNumber = 0;
02,333 int LongestLINE = 2048;
02,334 int LINESinChunk = 400000;
02,335
02,336     for( k = 0; k < 200; k++ )
02,337         SpacePADs[k]=' ';
02,338
02,339 printf("LineJustify_PAGODAo5, revision 1, written by Kaze.\n");
02,340 printf("Purpose: Padding the left side of x-grams with SPACES in order to form the main pillar.\n");
02,341 printf("Example:\n");
02,342 printf("D:\\LineWordreporter.exe LineJustify_PAGODAo5.lst\n");
02,343 printf("Note: Files can exceed 4GB limit.\n");

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)



Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,344
02,345 if (argc == 1) exit (1);
02,346 if (argc != 2) exit (2);
02,347
02,348 if( ( fp_in = fopen( argv[1], "rb" ) ) == NULL )
02,349 { printf( "Linereporter: Can't open file %s \n", argv[1] ); return( 1 ); }
02,350
02,351 fseek( fp_in, 0L, SEEK_END );
02,352 size_in = ftell( fp_in );
02,353 fseek( fp_in, 0L, SEEK_SET );
02,354
02,355 printf( "Buffered PADDING ... \n" );
02,356
02,357     NumberOfLines = 0;
02,358     FilesLEN = 0;
02,359     LINE10len = 0;
02,360     LINE10xgramlen = 0;
02,361
02,362     StrnglenTRAVERSED=0;
02,363     TRUEStrnglenTRAVERSED=0;
02,364     NumberOfWords = 0;
02,365
02,366     for( k = 0; k < size_in; k++ )
02,367     {
02,368         fread( &workbyte, 1, 1, fp_in );
02,369         if( workbyte != 10 )
02,370         { if( workbyte != 13 ) // NON UNIX
02,371             { if( LINE10len < 255 ) { LINE10[ LINE10len ] = workbyte; }
02,372                 LINE10len++;
02,373             }
02,374             else
02,375             {
02,376                 }
02,377             }
02,378             else
02,379             { if( 1 <= LINE10len && LINE10len <= 255 )
02,380                 { LINE10[ LINE10len ] = 0;
02,381 if( ( fp_inLINE = fopen( LINE10, "rb" ) ) == NULL )
02,382 { printf( "LineJustify_PAGODAo5: Can't open file %s \n", LINE10 ); return( 1 ); }
02,383
02,384 //fseek( fp_inLINE, 0L, SEEK_END ); //Rev. 12
02,385 //size_inLINE = ftell( fp_inLINE ); //Rev. 12
02,386 //fseek( fp_inLINE, 0L, SEEK_SET ); //Rev. 12
02,387
02,388 #if defined(_WIN32_ENVIRONMENT_)
02,389     // 64bit:
02,390 _lseeki64( fileno(fp_inLINE), 0L, SEEK_END );
02,391 size_inLINEIXFOUR = _telli64( fileno(fp_inLINE) );
02,392 _lseeki64( fileno(fp_inLINE), 0L, SEEK_SET );

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,393 #else
02,394 // 64bit:
02,395 fseeko( fp_inLINE, 0L, SEEK_END );
02,396 size_inLINESIXFOUR = ftello( fp_inLINE );
02,397 fseeko( fp_inLINE, 0L, SEEK_SET );
02,398 #endif /* defined(_WIN32_ENVIRONMENT_) */
02,399
02,400
02,401 if (LINE10[LINE10len-1-4]-'0' == 1) UnderscoresToSkip = 0;
02,402 else UnderscoresToSkip = (LINE10[LINE10len-1-4-2]-'0')-(LINE10[LINE10len-1-4]-'0') + 1; //PAGODAorder
02,403 //printf ("%d\n",UnderscoresToSkip);
02,404 // Kazahana_exascale.1.txt      '1.txt' which is LINE10[LINE10len-1] = 't' i.e. LINE10[LINE10len-1-4] = '1'
02,405 // Kazahana_exascale.2-1.txt    '1.txt'
02,406 // Kazahana_exascale.2-2.txt    '2.txt'
02,407 // Kazahana_exascale.3-1.txt    '1.txt'
02,408 // Kazahana_exascale.3-2.txt    '2.txt'
02,409 // Kazahana_exascale.3-3.txt    '3.txt'
02,410 // Kazahana_exascale.4-1.txt
02,411 // Kazahana_exascale.4-2.txt
02,412 // Kazahana_exascale.4-3.txt
02,413 // Kazahana_exascale.4-4.txt
02,414 // Kazahana_exascale.5-1.txt
02,415 // Kazahana_exascale.5-2.txt
02,416 // Kazahana_exascale.5-3.txt
02,417 // Kazahana_exascale.5-4.txt
02,418 // Kazahana_exascale.5-5.txt    '5.txt'
02,419
02,420 strcpy(TheWholeName, LINE10);
02,421 strcat(TheWholeName, PADext);
02,422 if ( ( fp_outLINE = fopen( TheWholeName, "wb" ) ) == NULL )
02,423 { printf( "LineJustify PAGODAo5: Can't open file %s\n", TheWholeName ); return( 1 ); }
02,424 //~~~~~
02,425 wrdlen = 0;
02,426 TRUEwrdlen = 0;
02,427 for( i = 0; i < size_inLINESIXFOUR; i++ )
02,428 {
02,429
02,430     // ~~~~~ Buffering fread, 10x faster [
02,431     if (workKoffset == -1) {
02,432         if (i + 1024*128 < size_inLINESIXFOUR) {
02,433             fread( &workK[0], 1, 1024*128, fp_inLINE );
02,434             workKoffset = 0;
02,435             workbyte = workK[workKoffset];
02,436         } else
02,437             fread( &workbyte, 1, 1, fp_inLINE );
02,438     } else {
02,439         workKoffset++;
02,440         workbyte = workK[workKoffset];
02,441         if (workKoffset == 1024*128 - 1) workKoffset = -1;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,442     }
02,443     // ~~~~~ Buffering fread, 10x faster ]
02,444
02,445     // ~~~~~ UnBuffered fread, 10x slower [
02,446         //fread( &workbyte, 1, 1, fp_inLINE );
02,447     // ~~~~~ UnBuffered fread, 10x slower ]
02,448
02,449 // [[[ PADDING [[[
02,450     if( workbyte != 10 )
02,451     { //if( workbyte != 13 ) // NON UNIX
02,452         { if( LINE10xgramlen < 255 ) { LINE10xgram[ LINE10xgramlen ] = workbyte; }
02,453             LINE10xgramlen++;
02,454         }
02,455     //else
02,456     //{
02,457     //}
02,458 }
02,459 else
02,460 { if( 1 <= LINE10xgramlen && LINE10xgramlen <= 255 )
02,461 { LINE10xgram[ LINE10xgramlen ] = workbyte;
02,462     LINE10xgramlen++;
02,463     LINE10xgram[ LINE10xgramlen ] = 0;
02,464
02,465 //2021-Apr-21 [
02,466 // Returning back the TAB delimiter by changing 10th position to \t
02,467 LINE10xgram[9] = '\t';
02,468 //2021-Apr-21 ]
02,469
02,470 // Instead of finding the exact LongestLeftPadder it can be set to 61-1=60
02,471 //ifdef quintupleton
02,472 //define LongestLineInclusive 61
02,473 // 0,000,001\tA      : padding with 60 - ( POS(A)-POS(\t)-1 ) SPACES after the '\t'
02,474 // 0,000,001\tA_B_C_D_E : padding with 60 - ( POS(E)-POS(\t)-1 ) SPACES after the '\t'
02,475 // 9..61 i.e. 61-'_E'=60
02,476 // UnderscoresToSkip + 1 points out '_'
02,477         //fwrite(LINE10xgram, strlen(LINE10xgram), 1, fp_outLINE);
02,478         fwrite(LINE10xgram, 9, 1, fp_outLINE); // Was fwrite(LINE10xgram, 10, 1, fp_outLINE); but TAB used needed space.
02,479 PositionToBePadded = strlen(LINE10xgram);
02,480 PositionToBePadded--;
02,481 // Now points to LF
02,482
02,483 //UnderscoresToSkip can be maximum 3 for order 5, because MAX(5-1,4-1,3-1,2-1,1-1) is not processed!
02,484
02,485 j=UnderscoresToSkip;
02,486 if (j == 0) {
02,487     while (LINE10xgram[PositionToBePadded] != '\t') {PositionToBePadded--;}
02,488 } else {
02,489     for( ; j > 0; ) {
02,490         j--;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,491         while (LINE10xgram[PositionToBePadded] != '_' ) {PositionToBePadded--;}
02,492         if (j) {PositionToBePadded--;}
02,493     }
02,494 }
02,495 PositionToBePadded++;
02,496         fwrite(SpacePADs, 60 - ((PositionToBePadded+1)-10-1), 1, fp_outLINE);
02,497         fwrite(&LINE10xgram[10], strlen(LINE10xgram)-10, 1, fp_outLINE);
02,498         LINE10xgramlen = 0;
02,499 LINE10xgram[ LINE10xgramlen ] = 0;
02,500     }
02,501 }
02,502 // ]]] PADDING ]]]
02,503
02,504     } // i 'for'
02,505     //~~~~~
02,506 fclose (fp_outLINE);
02,507
02,508     LINE10len = 0;
02,509 LINE10[ LINE10len ] = 0;
02,510 fclose( fp_inLINE );
02,511     }
02,512     }
02,513     } // k 'for'
02,514
02,515 //printf( "LineWordreporter: Encountered lines in all files: %s\n", _ui64toaKAZEcomma(NumberOfLines, 11ToaDigits, 10) );
02,516 //printf( "LineWordreporter: Encountered words in all files: %s\n", _ui64toaKAZEcomma(NumberOfWords, 11ToaDigits, 10) );
02,517 //printf( "LineWordreporter: Longest line: %s\n", _ui64toaKAZEcomma(StrnglenTRAVERSED, 11ToaDigits, 10) );
02,518 //printf( "LineWordreporter: Longest word: %s\n", _ui64toaKAZEcomma(TRUEStrnglenTRAVERSED, 11ToaDigits, 10) );
02,519
02,520 return(0);
02,521 // -----
02,522
02,523
02,524 strcpy(TheWholeName, argv[1]);
02,525 strcat(TheWholeName, DelimiterUnderscore);
02,526 strcat(TheWholeName, _ui64toaKAZEzerocomma(ChunkNumber++, 11ToaDigits, 10)+(26-6));
02,527 if( ( fp_outLINE = fopen( TheWholeName, "wb" ) ) == NULL )
02,528 { printf( "strstr_SHORT-SHOWDOWN: Can't open 'OSHO.TXT' file.\n" ); return( 1 ) ; }
02,529
02,530 printf( "Writing in %s ...\n", TheWholeName);
02,531
02,532 //Search area is between Strng[0] .. Strng[n-1]
02,533 StrnglenTRAVERSED=0; // Only traversed chars i.e. real
02,534 ThunderwithL=0;ThunderwithR=0;
02,535     for (;;)
02,536     {
02,537         while (Strng[ThunderwithR] != 10 && ThunderwithR < Strnglen-1) {ThunderwithR++;}
02,538         if (ThunderwithR - ThunderwithL + 1 <= LongestLINE + 2) {
02,539             fwrite(&Strng[ThunderwithL], ThunderwithR - ThunderwithL + 1, 1, fp_outLINE);

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,540 LinesWritten++; LinesWrittenChunk++;
02,541 }
02,542 if (LinesWrittenChunk == LINESinChunk) {
02,543 LinesWrittenChunk=0;
02,544 fclose (fp_outLINE);
02,545 strcpy(TheWholeName, argv[1]);
02,546 strcat(TheWholeName, DelimiterUnderscore);
02,547 strcat(TheWholeName, _ui64toaKAZEzerocomma(ChunkNumber++, 11ToaDigits, 10)*(26-6));
02,548 if( ( fp_outLINE = fopen( TheWholeName, "wb" ) ) == NULL )
02,549 { printf( "strsr_SHORT-SHOWDOWN: Can't open 'OSHO.TXT' file.\n" ); return( 1 ); }
02,550 printf( "Writing in %s ...\n", TheWholeName);
02,551 }
02,552 LinesEncountered++;
02,553 ThunderwithR++; ThunderwithL=ThunderwithR;
02,554 if (ThunderwithR >= Strnglen-1) break;
02,555 }
02,556
02,557 printf( "LinesEncountered: %lu\n", LinesEncountered);
02,558 printf( "LinesWritten : %lu\n", LinesWritten);
02,559 return(0);
02,560 // -----
02,561
02,562 printf("Input Pattern(up to 19+2000 chars): "); gets(Pattern); // char * __cdecl gets(char *);
02,563 Patternlen = strlen(&Pattern[0]);
02,564
02,565 // Replacing CR with NULL i.e. 13->0
02,566 for (ThunderwithL=0; ThunderwithL<Strnglen; ThunderwithL++)
02,567     if (Strng[ThunderwithL] == 13) Strng[ThunderwithL] = 0;
02,568 ThunderwithL=0;ThunderwithR=0;
02,569
02,570 printf( "Doing Search for Pattern(%dbytes) into String(%dbytes) line-by-line ...\n", Patternlen, Strnglen);
02,571
02,572 // 7[
02,573 clocks1 = clock();
02,574 for (Bozan=0; Bozan < (1<<4); Bozan++) // 16 times, at end >>4
02,575 {
02,576 //Search area is between Strng[0] .. Strng[n-1]
02,577 StrnglenTRAVERSED=0; // Only traversed chars i.e. real
02,578 ThunderwithL=0;ThunderwithR=0;
02,579 for (;;)
02,580 {
02,581 while (Strng[ThunderwithR] != 10 && ThunderwithR < Strnglen-1) {ThunderwithR++;}
02,582 FoundInPTR = strstr_Microsoft(&Strng[ThunderwithL], &Pattern[0]);
02,583 if ( FoundInPTR != NULL) {strstr_Microsofthits++; StrnglenTRAVERSED=StrnglenTRAVERSED+(FoundInPTR-&Strng[ThunderwithL]);} else StrnglenTRAVERSED=StrnglenTRAVERSED+(ThunderwithR - ThunderwithL);
02,584 LinesEncountered++;
02,585 ThunderwithR++; ThunderwithL=ThunderwithR;
02,586 if (ThunderwithR >= Strnglen-1) break;
02,587 }
02,588 if (Bozan != (1<<4)-1) LinesEncountered=0;

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxL2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)

```

02,589 }
02,590 clocks2 = clock(); TotalRoughSearchTime = clocks2 - clocks1; TotalRoughSearchTime++;
02,591 printf( "LinesEncountered: %lu\n", LinesEncountered);
02,592 printf( "strstr_Microsoft_hits/strstr_Microsoft_clocks: %lu/%lu\n", strstrMicrosoftHits>>4, (long)(TotalRoughSearchTime)>>4);
02,593 printf( "strstr_Microsoft performance: %luKB/clock\n", (StrnglenTRAVERSED/((long)(TotalRoughSearchTime)>>4))>>10);
02,594 printf( "StrnglenTRAVERSED: %lu bytes\n", StrnglenTRAVERSED);
02,595 // 7]
02,596
02,597 //printf("%s",ChunkName);
02,598 //ChunkName[3]='1';
02,599 //printf("%s",ChunkName);
02,600
02,601 /*
02,602 strcpy(TheWholeName, argv[1]);
02,603 strcat(TheWholeName, DelimiterUnderscore);
02,604 strcat(TheWholeName, _ui64toaKAZEzerocomma(ChunkNumber, 11T0aDigits, 10)+(26-6));
02,605
02,606 printf( "ChunkName: %s\n", TheWholeName);
02,607 ChunkNumber++;
02,608
02,609 strcpy(TheWholeName, argv[1]);
02,610 strcat(TheWholeName, DelimiterUnderscore);
02,611 strcat(TheWholeName, _ui64toaKAZEzerocomma(ChunkNumber, 11T0aDigits, 10)+(26-6));
02,612
02,613 printf( "ChunkName: %s\n", TheWholeName);
02,614 */
02,615 return(0);
02,616 }

```

Listing: MASAKARI\_Vanilla.BAS (r.8.1+); Last version: 2022-Jan-27; Font: MxPlus\_ToshibaTxl2\_8x16.ttf; Downloadable at: [www.sanmayce.com/Masakari.zip](http://www.sanmayce.com/Masakari.zip)