MASAKARI-The_Holy_Axe_Textual_Sidekick

MASAKARI_Vanilla.BAS:

```
00,001 ' Masakari.bas, written in QB64 v1.5 by Kaze (sanmayce@sanmayce.com), 2021-Mar-03
00,002 ' Many thanks go to Galleon, also thanks go to the www.qb64.org/forum members for sharing useful excerpts/etudes
00,003
00,004 ' Note #1: For wrapping set the flag 'WrapFlag = 1' or 'WrapFlag = 0' for vanilla.
00,005 ' 'MASAKARI_General-Purpose_Grade_English_Wordlist.wrd' is the unigram i.e. wordlist to be used here.
00,006
00,007 Const RSHIFTkey& = 100303
00,008 Const LSHIFTkey& = 100304
00,009 Const RCTRLkey& = 100305
00,010 Const LCTRLkey& = 100306
00,011 Const RALTkey& = 100307
00,012 Const LALTkey& = 100308
00,013
00,014 Const BACKSPCkey& = 8
00,015 Const TABkey& = 9
00,016 Const SPACEkey& = 32
00,017 Const ESCkey& = 27
00,018 Const ENTERkey& = 13
00,019
00,020 Const HOMEkey& = 18176
00,021 Const ENDkey& = 20224
00,022
00,023 Const INSkey& = 20992
```

```
Const DELkey& = 21248

Const PGUPkey& = 18688
Const PGDNkey& = 20736

Const LEFTkey& = 19200
Const RIGHTkey& = 19712
Const UPkey& = 18432
Const DOWNkey& = 20480

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.
Dim Shared WrapFlag As Integer

WrapFlag = 0 ' 1 means wrapping; 0 means vanilla
ToLoadOrNotFlag = 0 ' Should be 0; OLD: 1 means fast load but memory greedy; 0 means slow load but memory efficient

If InStr(LCase$(Command$), "/help") Or InStr(LCase$(Command$), "-h") Then
    $ScreenHide
    $Console
    _Console On
    _ConsoleTitle "Masakari console window"
    _Dest _Console

    Print "___  ___                _                 _                   _____"
    Print "|  \/  |               | |               (_)                 |  _  |"
    Print "| .  . |  __ _  ___   __ _ | | __ __ _  _ __  _   _ __   ___ __   __ \ V /     _   "
    Print "| |\/| | / _` |/ __| / _` || |/ // _` || '__|| | | '__| / _ \\ \ / / / _ \   _| |_ "
    Print "| |  | || (_| |\__ \| (_| ||   <| (_| || |   | | | |   |  __/ \ V /_| |_| | |_   _|"
    Print "\_|  |_/ \__,_||___/ \__,_||_|\_\\__,_||_|   |_| |_|    \___|  \_/(_)\_____/   |_|  "
    Print

    '
    ' _____
    '|  _  |
    '| |/' |
    '|  /| |
    '\ |_/ /
    ' \___/
    '
    '  __
    ' /  |
    '/_/ |
    '  | |
    ' _| |__
    '\_____/
    '
    ' _____
    '/ __  \
    '`' / /'
    '  / /
    './ /___
    '\_____/
    '
    ' _____
    '|____ |
    '    / /
    '    \ \
    '.___/ /
    '\____/
    '
    '   ___
    '  /   |
    ' / /| |
    '/ /_| |
    '\___  |
    '    |_/
    '
    ' _____
    '|  ___|
    '|___ \
    '    \ \
    '/\__/ /
    '\____/
    '
    '  ____
    ' / ___|
    '/ /___
    '| ___ \
    '| \_/ |
    '\_____/
    '
    ' ______
    '|___  /
    '   / /
    '  / /
    './ /
    '\_/
    '
    ' _____
    '|  _  |
    ' \ V /
    ' / _ \
    '| |_| |
    '\_____/
    '
    ' _____
    '|  _  |
    '| |_| |
    '\____ |
    '.___/ /
    '\____/

    '
    '   _
    '  (_)

    '
    '   _
    '  (_)
    '
    '                         _             _                  __
    '   _                    | |           | |                / /
    '  _| |_   ______         | |__   _   _ | |_   ___   ___      / /___
```

```
    '|_   _||_____|     | '_ \| | | || __|/ _ \/ __|  / // __|
    '  |_|           _  | |_) || |_| || |_|  __/\__ \ / / \__ \
    '               ( ) |_.__/  \__, | \__|\___||___//_/  |___/
    '               |/           __/ |
    '                           |___/

    ' _   _              _____
    '| \ | | _   _   ___|_   _|___  _ __    __ _  _   _
    '|  \| || | | | / _ \ | |  / _ \| '_ \  / _` || | | |
    '| |\  || |_| || (_) || | |  __/| | | || (_| || |_| |
    '|_| \_| \__, | \___/ |_|  \___||_| |_| \__, | \__,_|
    '        |___/                          |___/

    ' _                  _______ _________ _______  _        _______
    '( (    /||\     /|(  ___  )\__   __/(  ____ \( (    /|(  ____ \|\     /|
    '|  \  ( |( \   / )| (   ) |   ) (   | (    \/|  \  ( || (    \/| )   ( |
    '|   \ | | \ (_) / | |   | |   | |   | (__    |   \ | || |      | |   | |
    '| (\ \) |  \   /  | |   | |   | |   |  __)   | (\ \) || | ____ | |   | |
    '| | \   |   ) (   | |   | |   | |   | (      | | \   || | \_  )| |   | |
    '| )  \  |   | |   | (___) |   | |   | (____/\| )  \  || (___) || (___) |
    '|/    )_)   \_/   (_______)   )_(   (_______/|/    )_)(_______)(_______)

    ShowF1
    System
Else
    _ScreenShow ' Why so?!
    $Console
    _Console Off
    _Title "MASAKARI, revision 8+"
    $ExeIcon:'ColumnChart.ico'
    _Icon
End If

_Define A-Z As _INTEGER64

$If WINDOWS Then
    DECLARE LIBRARY "mem"
    FUNCTION MemInUsePercent~&&
    FUNCTION TotalPhysicalMem~&&
    FUNCTION FreePhysicalMem~&&
    FUNCTION TotalPagingFile~&&
    FUNCTION FreePagingFile~&&
    FUNCTION TotalVirtualMem~&&
    FUNCTION FreeVirtualMem~&&
    FUNCTION FreeExtendedMem~&&
    FUNCTION GetCPULoad#
    END DECLARE
$End If

Dim Shared Mwholefile As _MEM
Dim Shared MhandleOFF As _MEM
Dim Shared MhandleLEN As _MEM 'paired with above one, added for speed, no need to search for the length at offset 'MhandleOFF'

Dim QWORD As _Unsigned _Integer64
'DIM SHARED Byte AS STRING * 1 'not used since r.7
Dim PrevByte As String * 1
Dim LastByte As String * 1

_ScreenMove 102, 3
XdimCOL = 128
YdimROW = 40 ' ensure old laptops with 768pixels vertical will hold the whole window
wide& = _DesktopWidth
high& = _DesktopHeight
If high& > 1000 Then YdimROW = 60
If wide& >= 1920 Then XdimCOL = 198
If XdimCOL = 128 And YdimROW = 40 Then _Title "MASAKARI, revision 8+ 128x40, The 'Holy Axe' English Text Sidekick"
If XdimCOL = 128 And YdimROW = 60 Then _Title "MASAKARI, revision 8+ 128x60, The 'Holy Axe' English Text Sidekick"
If XdimCOL = 198 And YdimROW = 60 Then _Title "MASAKARI, revision 8+ 198x60, The 'Holy Axe' English Text Sidekick"

Mode4K = 0 'Make it 1 or 0, if 1 then 16 becomes 32 i.e. the font is 4x bigger: 8x16 becomes 16x32
If wide& >= 1920 * 2 Then Mode4K = 1

'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!

ASCIIFrame = 0
ASCIIFramePEN = 0
'YdimROW = 40 'debug, test the Search Panel

$If WINDOWS Then
    PSPlike$ = _CWD$ + "\"
$Else
    PSPlike$ = _CWD$ + "/"
$End If

'' SOUND PLAYER [
''s& = _SNDOPEN(PSPlike$+"music_zapsplat_rapid_turnaround.mp3")
's& = _SNDOPEN(PSPlike$+"music_pioxonaq_agressive_lightning.mp3")
'IF s& <> 0 THEN
'    _SNDPLAY s& 'check for valid handle before using!
'    'The _SNDLOOP statement is like _SNDPLAY but the sound is looped. Uses a handle from the _SNDOPEN function.
'    'Example 1: Playing a previously opened sound at half volume.
'    '_SNDPLAYCOPY applause&, 0.5
'    volumeINT% = 9 '10 means 100% means 1.0, down to 0 i.e. 0..10 or 0,0.1,...1.0
'    StopFlag = 0
'    DO
'        k$ = INKEY$
'        SELECT CASE k$
'            CASE "n"
'                IF volumeINT% >= 1 THEN volumeINT% = volumeINT% - 1
'                _SNDVOL s&, volumeINT% / 10
```

```
00,224 '            CASE "m"
00,225 '                IF volumeINT% <= 8 THEN volumeINT% = volumeINT% + 1 ' don't go 100% but up to 90%
00,226 '                _SNDVOL s&, volumeINT% / 10
00,227 '            CASE "p"
00,228 '                _SNDPAUSE s&
00,229 '            CASE " "
00,230 '                _SNDPLAY s&
00,231 '            CASE CHR$(27)
00,232 '                _SNDSTOP s&
00,233 '                _SNDCLOSE s&
00,234 '                StopFlag = 1
00,235 '        END SELECT
00,236 '        LOCATE 1, 1
00,237 '        IF StopFlag = 0 THEN PRINT "Volume:"; volumeINT%; "; Position:"; INT(_SNDGETPOS(s&)); "second, up to"; INT(_SNDLEN(s&))
00,238 '    LOOP UNTIL StopFlag
00,239 'END IF
00,240 '' SOUND PLAYER ]
00,241
00,242 'spell [
00,243 Dim lfixed As String * 32
00,244 TotalWrd& = 0
00,245 Open PSPlike$ + "masakari.ind" For Binary As #2
00,246 If LOF(2) Then
00,247     TotalWrd& = LOF(2) / 32
00,248     Close #2
00,249 Else
00,250     Close #2
00,251     Kill PSPlike$ + "masakari.ind"
00,252     Open PSPlike$ + "masakari.wrd" For Binary As #2
00,253     If LOF(2) Then
00,254         Close #2
00,255         Open PSPlike$ + "masakari.wrd" For Input As #1
00,256         Open PSPlike$ + "masakari.ind" For Random Access Write As #2 Len = 32
00,257         Do While Not EOF(1)
00,258             Line Input #1, l$
00,259             lfixed$ = l$
00,260             Put #2, , lfixed$: TotalWrd& = TotalWrd& + 1
00,261             FL2wrd& = 1
00,262         Loop
00,263         Close #1, #2
00,264     Else
00,265         Close #2
00,266         Kill PSPlike$ + "masakari.wrd"
00,267     End If
00,268 End If
00,269 'spell ]
00,270
00,271 GoTo SkipFontReviews
00,272 'SCREEN 0 'It is better to define your own mode, as:
00,273 handle& = _NewImage(40, 20, 0)
00,274 Screen handle&
00,275 _Dest handle&
00,276
00,277 '_FONT _LOADFONT("C:\windows\fonts\cour.ttf", 32, "MONOSPACE")
00,278 '_FONT _LOADFONT("C:\windows\fonts\lucon.ttf", 32, "MONOSPACE")
00,279 '_FONT _LOADFONT("C:\windows\fonts\MxPlus_Cordata_PPC-400.ttf", 32, "MONOSPACE")
00,280 '_FONT _LOADFONT("C:\windows\fonts\MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
00,281 _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
00,282
00,283 'RESTORE Microsoft_pc_cp437 'United States MS DOS
00,284 '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.
00,285 '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.
00,286 'RESTORE Microsoft_pc_cpMIK 'Cyrillic Bulgarian Pravetz 16 for MS-DOS
00,287 Restore Microsoft_pc_cpGESCH 'My codepage a.k.a. Gesch
00,288
00,289 For ASCIIcode = 128 To 255
00,290     Read unicode
00,291     _MapUnicode unicode To ASCIIcode
00,292 Next
00,293
00,294 For i = 0 To 255
00,295     Print Chr$(i) + " ";
00,296     cnt = cnt + 1
00,297     If cnt Mod 16 = 0 Then Print: Print cnt;
00,298 Next
00,299 End
00,300
00,301 Microsoft_pc_cp737: 'Greek MS DOS displays Greek alphabet for algebraic formulas.
00,302 Data 913,914,915,916,917,918,919,920,921,922,923,924,925,926,927,928
00,303 Data 929,931,932,933,934,935,936,937,945,946,947,948,949,950,951,952
00,304 Data 953,954,955,956,957,958,959,960,961,963,962,964,965,966,967,968
00,305 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,306 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,307 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,308 Data 969,940,941,942,970,943,972,973,971,974,902,904,905,906,908,910
00,309 Data 911,177,8805,8804,938,939,247,8776,176,8729,183,8730,8319,178,9632,160
00,310
00,311 Microsoft_pc_cp775: 'Estonian, Lithuanian and Latvian languages.
00,312 Data 262,252,233,257,228,291,229,263,322,275,342,343,299,377,196,197
00,313 Data 201,230,198,333,246,290,162,346,347,214,220,248,163,216,215,164
00,314 Data 256,298,243,379,380,378,8221,166,169,174,172,189,188,321,171,187
00,315 Data 9617,9618,9619,9474,9508,260,268,280,278,9571,9553,9559,9565,302,352,9488
00,316 Data 9492,9524,9516,9500,9472,9532,370,362,9562,9556,9577,9574,9568,9552,9580,381
00,317 Data 261,269,281,279,303,353,371,363,382,9496,9484,9608,9604,9612,9616,9600
00,318 Data 211,223,332,323,245,213,181,324,310,311,315,316,326,274,325,8217
00,319 Data 173,177,8220,190,182,167,247,8222,176,8729,183,185,179,178,9632,160
00,320
00,321 Microsoft_pc_cp850: 'Western Europe, Spain, England
```

```
00,322 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,236,196,197
00,323 Data 201,230,198,244,246,242,251,249,255,214,220,248,163,216,215,402
00,324 Data 225,237,243,250,241,209,170,186,191,174,172,189,188,161,171,187
00,325 Data 9617,9618,9619,9474,9508,193,194,192,169,9571,9553,9559,9565,162,165,9488
00,326 Data 9492,9524,9516,9500,9472,9532,227,195,9562,9556,9577,9574,9568,9552,9580,164
00,327 Data 240,208,202,203,200,305,205,206,207,9496,9484,9608,9604,166,204,9600
00,328 Data 211,223,212,210,245,213,181,254,222,218,219,217,253,221,175,180
00,329 Data 173,177,8215,190,182,167,247,184,176,168,183,185,179,178,9632,160
00,330
00,331 Microsoft_pc_cp852: 'Central European languages that use Latin script such as Bosnian, Croatian, Czech, Hungarian, Polish, Romanian, Serbian or Slovak.
00,332 Data 199,252,233,226,228,367,263,231,322,235,336,337,238,377,196,262
00,333 Data 201,313,314,244,246,317,318,346,347,214,220,356,357,321,215,269
00,334 Data 225,237,243,250,260,261,381,382,280,281,172,378,268,351,171,187
00,335 Data 9617,9618,9619,9474,9508,193,194,282,350,9571,9553,9559,9565,379,380,9488
00,336 Data 9492,9524,9516,9500,9472,9532,258,259,9562,9556,9577,9574,9568,9552,9580,164
00,337 Data 273,272,270,203,271,327,205,206,283,9496,9484,9608,9604,354,366,9600
00,338 Data 211,223,212,323,324,328,352,353,340,218,341,368,253,221,355,180
00,339 Data 173,733,731,711,728,167,247,184,176,168,729,369,344,345,9632,160
00,340
00,341 Microsoft_pc_cp855: 'Cyrillic code page to be used under MS-DOS
00,342 Data 1106,1026,1107,1027,1105,1025,1108,1028,1109,1029,1110,1030,1111,1031,1112,1032
00,343 Data 1113,1033,1114,1034,1115,1035,1116,1036,1118,1038,1119,1039,1102,1070,1098,1066
00,344 Data 1072,1040,1073,1041,1094,1062,1076,1044,1077,1045,1092,1060,1075,1043,171,187
00,345 Data 9617,9618,9619,9474,9508,1093,1061,1080,1048,9571,9553,9559,9565,1081,1049,9488
00,346 Data 9492,9524,9516,9500,9472,9532,1082,1050,9562,9556,9577,9574,9568,9552,9580,164
00,347 Data 1083,1051,1084,1052,1085,1053,1086,1054,1087,9496,9484,9608,9604,1055,1103,9600
00,348 Data 1071,1088,1056,1089,1057,1090,1058,1091,1059,1078,1046,1074,1042,1100,1068,8470
00,349 Data 173,1099,1067,1079,1047,1096,1064,1101,1069,1097,1065,1095,1063,167,9632,160
00,350
00,351 Microsoft_pc_cp857: 'Turkish MS DOS
00,352 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,305,196,197
00,353 Data 201,230,198,244,246,242,251,249,304,214,220,248,163,216,350,351
00,354 Data 225,237,243,250,241,209,286,287,191,174,172,189,188,161,171,187
00,355 Data 9617,9618,9619,9474,9508,193,194,192,169,9571,9553,9559,9565,162,165,9488
00,356 Data 9492,9524,9516,9500,9472,9532,227,195,9562,9556,9577,9574,9568,9552,9580,164
00,357 Data 186,170,202,203,200,0,205,206,207,9496,9484,9608,9604,166,204,9600
00,358 Data 211,223,212,210,245,213,181,0,215,218,219,217,236,255,175,180
00,359 Data 173,177,0,190,182,167,247,184,176,168,183,185,179,178,9632,160
00,360
00,361 Microsoft_pc_cp860: 'Portuguese language. MS DOS
00,362 Data 199,252,233,226,227,224,193,231,234,202,232,205,212,236,195,194
00,363 Data 201,192,200,244,245,242,218,249,204,213,220,162,163,217,8359,211
00,364 Data 225,237,243,250,241,209,170,186,191,210,172,189,188,161,171,187
00,365 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,366 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,367 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,368 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,369 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,370
00,371 Microsoft_pc_cp861: 'Icelandic language (as well as other Nordic languages). MS DOS
00,372 Data 199,252,233,226,228,224,229,231,234,235,232,208,240,222,196,197
00,373 Data 201,230,198,244,246,254,251,221,253,214,220,248,163,216,8359,402
00,374 Data 225,237,243,250,193,205,211,218,191,8976,172,189,188,161,171,187
00,375 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,376 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,377 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,378 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,379 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,380
00,381 Microsoft_pc_cp862: 'Hebrew letters in positions 809A hex, but otherwise it is identical to CP437. Now obsolete, see CP1255
00,382 Data 1488,1489,1490,1491,1492,1493,1494,1495,1496,1497,1498,1499,1500,1501,1502,1503
00,383 Data 1504,1505,1506,1507,1508,1509,1510,1511,1512,1513,1514,162,163,165,8359,402
00,384 Data 225,237,243,250,241,209,170,186,191,8976,172,189,188,161,171,187
00,385 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,386 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,387 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,388 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,389 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,390
00,391 Microsoft_pc_cp863: 'French language (mainly in Canada). MS DOS
00,392 Data 199,252,233,226,194,224,182,231,234,235,232,239,238,8215,192,167
00,393 Data 201,200,202,244,203,207,251,249,164,212,220,162,163,217,219,402
00,394 Data 166,180,243,250,168,184,179,175,206,8976,172,189,188,190,171,187
00,395 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,396 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,397 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,398 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,399 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,400
00,401 Microsoft_pc_cp864: 'Arabic MS DOS
00,402 Data 176,183,8729,8730,9618,9472,9474,9532,9508,9516,9500,9524,9488,9484,9492,9496
00,403 Data 946,8734,966,177,189,188,8776,171,187,65271,65272,0,0,65275,65276,0
00,404 Data 160,173,65154,163,164,65156,0,0,65166,65167,65173,65177,1548,65181,65185,65189
00,405 Data 1632,1633,1634,1635,1636,1637,1638,1639,1640,1641,65233,1563,65201,65205,65209,1567
00,406 Data 162,65152,65153,65155,65157,65226,65163,65165,65169,65171,65175,65179,65183,65187,65191,65193
00,407 Data 65195,65197,65199,65203,65207,65211,65215,65217,65221,65227,65231,166,172,247,215,65225
00,408 Data 1600,65235,65239,65243,65247,65251,65255,65259,65261,65263,65267,65213,65228,65230,65229,65249
00,409 Data 65149,1617,65253,65257,65260,65264,65266,65232,65237,65269,65270,65245,65241,65265,9632,0
00,410
00,411 Microsoft_pc_cp865: 'Nordic languages (except Icelandic, for which CP861 is used). MS DOS
00,412 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,236,196,197
00,413 Data 201,230,198,244,246,242,251,249,255,214,220,248,163,216,8359,402
00,414 Data 225,237,243,250,241,209,170,186,191,8976,172,189,188,161,171,164
00,415 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,416 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,417 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,418 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,419 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,420
00,421 Microsoft_pc_cp866: 'Cyrillic alphabetical order code page to be used with MS-DOS
```

```
00,422 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,423 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,424 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,425 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,426 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,427 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,428 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
00,429 Data 1025,1105,1028,1108,1031,1111,1038,1118,176,8729,183,8730,8470,164,9632,160
00,430
00,431 Microsoft_pc_cp869: 'Greek MS DOS. Less popular than Code page 737
00,432 Data 0,0,0,0,0,0,902,0,183,172,166,8216,8217,904,8213,905
00,433 Data 906,938,908,0,0,910,939,169,911,178,179,940,163,941,942,943
00,434 Data 970,912,972,973,913,914,915,916,917,918,919,189,920,921,171,187
00,435 Data 9617,9618,9619,9474,9508,922,923,924,925,9571,9553,9559,9565,926,927,9488
00,436 Data 9492,9524,9516,9500,9472,9532,928,929,9562,9556,9577,9574,9568,9552,9580,931
00,437 Data 932,933,934,935,936,937,945,946,947,9496,9484,9608,9604,948,949,9600
00,438 Data 950,951,952,953,954,955,956,957,958,959,960,961,963,962,964,900
00,439 Data 173,177,965,966,967,167,968,901,176,168,969,971,944,974,9632,160
00,440
00,441 Microsoft_pc_cp874: 'Thai MS DOS and Windows
00,442 Data 8364,0,0,0,0,8230,0,0,0,0,0,0,0,0,0,0
00,443 Data 0,8216,8217,8220,8221,8226,8211,8212,0,0,0,0,0,0,0,0
00,444 Data 160,3585,3586,3587,3588,3589,3590,3591,3592,3593,3594,3595,3596,3597,3598,3599
00,445 Data 3600,3601,3602,3603,3604,3605,3606,3607,3608,3609,3610,3611,3612,3613,3614,3615
00,446 Data 3616,3617,3618,3619,3620,3621,3622,3623,3624,3625,3626,3627,3628,3629,3630,3631
00,447 Data 3632,3633,3634,3635,3636,3637,3638,3639,3640,3641,3642,0,0,0,0,3647
00,448 Data 3648,3649,3650,3651,3652,3653,3654,3655,3656,3657,3658,3659,3660,3661,3662,3663
00,449 Data 3664,3665,3666,3667,3668,3669,3670,3671,3672,3673,3674,3675,0,0,0,0
00,450
00,451 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.
00,452 Data 8364,0,8218,0,8222,8230,8224,8225,0,8240,352,8249,346,356,381,377
00,453 Data 0,8216,8217,8220,8221,8226,8211,8212,0,8482,353,8250,347,357,382,378
00,454 Data 160,711,728,321,164,260,166,167,168,169,350,171,172,173,174,379
00,455 Data 176,177,731,322,180,181,182,183,184,261,351,187,317,733,318,380
00,456 Data 340,193,194,258,196,313,262,199,268,201,280,203,282,205,206,270
00,457 Data 272,323,327,211,212,336,214,215,344,366,218,368,220,221,354,223
00,458 Data 341,225,226,259,228,314,263,231,269,233,281,235,283,237,238,271
00,459 Data 273,324,328,243,244,337,246,247,345,367,250,369,252,253,355,729
00,460
00,461 Microsoft_windows_cp1252: 'Windows Western languages with Latin alphabet, used by default in the legacy components of Microsoft Windows in English.
00,462 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,352,8249,338,0,381,0
00,463 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,353,8250,339,0,382,376
00,464 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175
00,465 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191
00,466 Data 192,193,194,195,196,197,198,199,200,201,202,203,204,205,206,207
00,467 Data 208,209,210,211,212,213,214,215,216,217,218,219,220,221,222,223
00,468 Data 224,225,226,227,228,229,230,231,232,233,234,235,236,237,238,239
00,469 Data 240,241,242,243,244,245,246,247,248,249,250,251,252,253,254,255
00,470
00,471 Microsoft_windows_cp1253: 'Greek (but not polytonic Greek) Not fully compatible with ISO 8859-7 (? is located differently).
00,472 Data 8364,0,8218,402,8222,8230,8224,8225,0,8240,0,8249,0,0,0,0
00,473 Data 0,8216,8217,8220,8221,8226,8211,8212,0,8482,0,8250,0,0,0,0
00,474 Data 160,901,902,163,164,165,166,167,168,169,0,171,172,173,174,8213
00,475 Data 176,177,178,179,900,181,182,183,904,905,906,187,908,189,910,911
00,476 Data 912,913,914,915,916,917,918,919,920,921,922,923,924,925,926,927
00,477 Data 928,929,0,931,932,933,934,935,936,937,938,939,940,941,942,943
00,478 Data 944,945,946,947,948,949,950,951,952,953,954,955,956,957,958,959
00,479 Data 960,961,962,963,964,965,966,967,968,969,970,971,972,973,974,0
00,480
00,481 Microsoft_windows_cp1254: 'Turkish
00,482 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,352,8249,338,0,0,0
00,483 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,353,8250,339,0,0,376
00,484 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175
00,485 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191
00,486 Data 192,193,194,195,196,197,198,199,200,201,202,203,204,205,206,207
00,487 Data 286,209,210,211,212,213,214,215,216,217,218,219,220,304,350,223
00,488 Data 224,225,226,227,228,229,230,231,232,233,234,235,236,237,238,239
00,489 Data 287,241,242,243,244,245,246,247,248,249,250,251,252,305,351,255
00,490
00,491 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.
00,492 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,0,8249,0,0,0,0
00,493 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,0,8250,0,0,0,0
00,494 Data 160,161,162,163,8362,165,166,167,168,169,215,171,172,173,174,175
00,495 Data 176,177,178,179,180,181,182,183,184,185,247,187,188,189,190,191
00,496 Data 1456,1457,1458,1459,1460,1461,1462,1463,1464,1465,0,1467,1468,1469,1470,1471
00,497 Data 1472,1473,1474,1475,1520,1521,1522,1523,1524,0,0,0,0,0,0,0
00,498 Data 1488,1489,1490,1491,1492,1493,1494,1495,1496,1497,1498,1499,1500,1501,1502,1503
00,499 Data 1504,1505,1506,1507,1508,1509,1510,1511,1512,1513,1514,0,0,8206,8207,0
00,500
00,501 Microsoft_windows_cp1256: 'Arabic Latin Windows
00,502 Data 8364,1662,8218,402,8222,8230,8224,8225,710,8240,1657,8249,338,1670,1688,1672
00,503 Data 1711,8216,8217,8220,8221,8226,8211,8212,1705,8482,1681,8250,339,8204,8205,1722
00,504 Data 160,1548,162,163,164,165,166,167,168,169,1726,171,172,173,174,175
00,505 Data 176,177,178,179,180,181,182,183,184,185,1563,187,188,189,190,1567
00,506 Data 1729,1569,1570,1571,1572,1573,1574,1575,1576,1577,1578,1579,1580,1581,1582,1583
00,507 Data 1584,1585,1586,1587,1588,1589,1590,215,1591,1592,1593,1594,1600,1601,1602,1603
00,508 Data 224,1604,226,1605,1606,1607,1608,231,232,233,234,235,1609,1610,238,239
00,509 Data 1611,1612,1613,1614,244,1615,1616,247,1617,249,1618,251,252,8206,8207,1746
00,510
00,511 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.
00,512 Data 8364,0,8218,0,8222,8230,8224,8225,0,8240,0,8249,0,168,711,184
00,513 Data 0,8216,8217,8220,8221,8226,8211,8212,0,8482,0,8250,0,175,731,0
00,514 Data 160,0,162,163,164,0,166,167,216,169,342,171,172,173,174,198
00,515 Data 176,177,178,179,180,181,182,183,248,185,343,187,188,189,190,230
00,516 Data 260,302,256,262,196,197,280,274,268,201,377,278,290,310,298,315
00,517 Data 352,323,325,211,332,213,214,215,370,321,346,362,220,379,381,223
00,518 Data 261,303,257,263,228,229,281,275,269,233,378,279,291,311,299,316
```

```
00,519 Data 353,324,326,243,333,245,246,247,371,322,347,363,252,380,382,729
00,520
00,521 Microsoft_windows_cp1258: 'Vietnamese. [https://en.wikipedia.org/wiki/UTF-8 UTF-8] is the preferred encoding for Vietnamese in modern applications.
00,522 Data 8364,0,8218,402,8222,8230,8224,8225,710,8240,0,8249,338,0,0,0
00,523 Data 0,8216,8217,8220,8221,8226,8211,8212,732,8482,0,8250,339,0,0,376
00,524 Data 160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175
00,525 Data 176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191
00,526 Data 192,193,194,258,196,197,198,199,200,201,202,203,768,205,206,207
00,527 Data 272,209,777,211,212,416,214,215,216,217,218,219,220,431,771,223
00,528 Data 224,225,226,259,228,229,230,231,232,233,234,235,769,237,238,239
00,529 Data 273,241,803,243,244,417,246,247,248,249,250,251,252,432,8363,255
00,530
00,531 Microsoft_pc_cp437: 'United States MS DOS
00,532 Data 199,252,233,226,228,224,229,231,234,235,232,239,238,236,196,197
00,533 Data 201,230,198,244,246,242,251,249,255,214,220,162,163,165,8359,402
00,534 Data 225,237,243,250,241,209,170,186,191,8976,172,189,188,161,171,187
00,535 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,536 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,537 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,538 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,539 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,540
00,541 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.
00,542 Data 1026,1027,8218,1107,8222,8230,8224,8225,8364,8240,1033,8249,1034,1036,1035,1039
00,543 Data 1106,8216,8217,8220,8221,8226,8211,8212,0,8482,1113,8250,1114,1116,1115,1119
00,544 Data 160,1038,1118,1032,164,1168,166,167,1025,169,1028,171,172,173,174,1031
00,545 Data 176,177,1030,1110,1169,181,182,183,1105,8470,1108,187,1112,1029,1109,1111
00,546 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,547 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,548 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,549 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
00,550
00,551 Microsoft_pc_cpMIK: 'Cyrillic Bulgarian Pravetz 16 for MS-DOS
00,552 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,553 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,554 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,555 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
00,556 Data 9492,9524,9516,9500,9472,9532,9571,9553,9562,9566,9577,9574,9568,9552,9580,9488
00,557 Data 9617,9618,9619,9474,9508,8470,167,9559,9565,9496,9484,9608,9604,9612,9616,9600
00,558 Data 945,223,915,960,931,963,181,964,934,920,937,948,8734,966,949,8745
00,559 Data 8801,177,8805,8804,8992,8993,247,8776,176,8729,183,8730,8319,178,9632,160
00,560
00,561 Microsoft_pc_cpGESCH: 'Gesch is Sanmayce's layout, combining the MIK and 437, in this way:
00,562 'First half of big Cyrillic letters:
00,563 Data 1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,1053,1054,1055
00,564 'Second half of big Cyrillic letters:
00,565 Data 1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,1066,1067,1068,1069,1070,1071
00,566 'First half of small Cyrillic letters:
00,567 Data 1072,1073,1074,1075,1076,1077,1078,1079,1080,1081,1082,1083,1084,1085,1086,1087
00,568 '1st third of CP437 drawing symbols:
00,569 Data 9617,9618,9619,9474,9508,9569,9570,9558,9557,9571,9553,9559,9565,9564,9563,9488
00,570 '2nd third of CP437 drawing symbols:
00,571 Data 9492,9524,9516,9500,9472,9532,9566,9567,9562,9556,9577,9574,9568,9552,9580,9575
00,572 '3rd third of CP437 drawing symbols:
00,573 Data 9576,9572,9573,9561,9560,9554,9555,9579,9578,9496,9484,9608,9604,9612,9616,9600
00,574 'Second half of small Cyrillic letters:
00,575 Data 1088,1089,1090,1091,1092,1093,1094,1095,1096,1097,1098,1099,1100,1101,1102,1103
00,576 'Last 16 of CP437 symbols:
00,577 '                                          u-kr E:   e:   << >>  Up_ ...
00,578 Data 8216,8217,8218,8219,8220,8221,8222,8223,176,1118,1025,1105,171,187,175,8230
00,579
00,580 SkipFontReviews:
00,581 handle& = _NewImage(XdimCOL + IndigoField, YdimROW + 1, 0)
00,582 Screen handle&
00,583 _Dest handle&
00,584
00,585 ' _DEST 0 refers to the present program SCREEN. You can use 0 to refer to the present program
00,586 '  SCREEN.
00,587
00,588 '_FONT 16 'wish we could use the old 8x16 bitmap/raster fonts from DOS times...
00,589
00,590 '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)
00,591 'IF _FILEEXISTS("C:\windows\fonts\MxPlus_ToshibaTxL2_8x16.ttf") = 0 THEN
00,592 If _FileExists(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf") = 0 Then
00,593     If Mode4K = 0 Then
00,594         '_FONT _LOADFONT(PSPlike$ +"cour.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
00,595         '_FONT _LOADFONT(PSPlike$ +"cour.ttf", 16, "MONOSPACE")
00,596         '_FONT _LOADFONT(PSPlike$ +"lucon.ttf", 14, "MONOSPACE") ' 8x14 is not bad at all!
00,597         '_FONT _LOADFONT(PSPlike$ +"lucon.ttf", 16, "MONOSPACE")
00,598         'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
00,599     Else
00,600         '_FONT _LOADFONT(PSPlike$ +"cour.ttf", 32, "MONOSPACE")
00,601         _Font _LoadFont(PSPlike$ + "lucon.ttf", 32, "MONOSPACE")
00,602         'If not a single line is uncommented, above, then the EMULATED (Trident?) font is in use.
00,603     End If
00,604 Else
00,605     If Mode4K = 0 Then
00,606         _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 16, "MONOSPACE") 'Toshiba rules...
00,607         '_FONT _LOADFONT(PSPlike$ +"MxPlus_Cordata_PPC-400.ttf", 16, "MONOSPACE") 'Beautiful, yet the '2' has a missing dot - up-right
00,608     Else
00,609         _Font _LoadFont(PSPlike$ + "MxPlus_ToshibaTxL2_8x16.ttf", 32, "MONOSPACE")
00,610         '_FONT _LOADFONT(PSPlike$ +"MxPlus_Cordata_PPC-400.ttf", 32, "MONOSPACE")
00,611     End If
00,612 End If
00,613 Restore Microsoft_pc_cpGESCH 'My codepage a.k.a. Gesch
00,614 'RESTORE Microsoft_windows_cp1251
00,615 For ASCIIcode = 128 To 255
00,616     Read unicode
00,617     _MapUnicode unicode To ASCIIcode
```

```
Next

' Either one must be uncommented:
_Display
'_AUTODISPLAY 'no need of refreshing

PurpleFlag = 0

NormalFRGr = 3
If PurpleFlag Then NormalFRGr = 9
NormalBCKGr = 0
If PurpleFlag Then NormalBCKGr = 0

InverseFRGr = 0
If PurpleFlag Then InverseFRGr = 8
InverseBCKGr = 3
If PurpleFlag Then InverseBCKGr = 0

Color NormalFRGr, NormalBCKGr

Print "___  ___                _                _                  _____"
Print "|  \/  |               | |              (_)                |  _  |"
Print "| .  . | __ _ ___  __ _| | ____ _ _ __ _   _ __ _____   __ \ V /    _   "
Print "| |\/| |/ _` / __|/ _` | |/ / _` | '__| | | '__/ _ \ \ / / / _ \  _| |_ "
Print "| |  | | (_| \__ \ (_| |   < (_| | |  | | | | |  __/\ V /_| |_| ||_   _|"
Print "\_|  |_/\__,_|___/\__,_|_|\_\__,_|_|  |_| |_|  \___| \_/(_)\_____/  |_|  "
Print

$If WINDOWS Then
    i = 0
    DO
    i = i + 1
    setting$ = ENVIRON$(i) ' get a setting from the list
    IF INSTR(setting$, "NUMBER_OF_PROCESSORS") THEN NUMBER_OF_PROCESSORS$ = setting$
    LOOP UNTIL setting$ = ""
    PRINT NUMBER_OF_PROCESSORS$

    ' For some reason the Drag-and-Drop stops working when running as Administrator?! [
    'C$ = ENVIRON$("ALLUSERSPROFILE") 'try desktop for all users
    'SHORTCUT$ = C$ + "\Desktop\" + "Masakari.log.txt" 'create filename for the desktop
    ' For some reason the Drag-and-Drop stops working when running as Administrator?! ]
    SHORTCUT$ = _STARTDIR$ + "\" + "Masakari.log.txt"
    OPEN SHORTCUT$ FOR APPEND AS #13
    PRINT #13,
    PRINT #13, "[Written on DATE: " + DATE$ + ", TIME: " + TIME$ + "]:"
    PRINT #13,
    'CLOSE #13
    PRINT "The log/output file, opened in APPEND mode: "; CHR$(34); SHORTCUT$; CHR$(34)
$End If

'f = FREEFILE
'SHELL _HIDE "cd > PRGMDIR.INF" 'get the current program path
'OPEN "PRGMDIR.INF" FOR INPUT AS #f
'LINE INPUT #f, current_program_path$
'CLOSE #f
'KILL "PRGMDIR.INF"
'PRINT "Current working directory path (obtained via SHELL _HIDE): "; CHR$(34); current_program_path$; CHR$(34)

'Ugh, they are reversed?!
'PRINT "Current working directory path: "; CHR$(34); _CWD$; CHR$(34)
'PRINT "User's program calling path: "; CHR$(34); _STARTDIR$; CHR$(34)
Print "Current working directory path: "; Chr$(34); _StartDir$; Chr$(34)
Print "User's program calling path: "; Chr$(34); _CWD$; Chr$(34)
'PRINT "Command line parameters sent when a program is started: "; CHR$(34); COMMAND$; CHR$(34)
'PRINT "_OS$="; _OS$

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

$If WINDOWS Then
    _ACCEPTFILEDROP 'enables drag/drop functionality
    IF COMMAND$ = "" THEN PRINT: PRINT "Drag files from a folder and drop them in this window... May press Alt+X or Alt+Q to eXit/Quit..."
$End If

_Display

a$ = ""
PostfixedToHeader = CsrLin
Do

    Locate PostfixedToHeader, 1

    'FOR d = 1 TO _DEVICES 'number of input devices found
    '    dev$ = _DEVICE$(d)
    '    IF INSTR(dev$, "[MOUSE]") THEN buttons = _LASTBUTTON(d): wheels = _LASTWHEEL(d): EXIT FOR
    'NEXT
    'PRINT
    'PRINT "Mouse Buttons:"; buttons
    'PRINT "Mouse Wheels:"; wheels
    ''d = _DEVICES '  always read number of devices to enable device input
    ''DO WHILE _DEVICEINPUT(2) 'loop only runs during a device 2 mouse event
    ''LOOP

    $If WINDOWS Then
        ' MEMUSAGE by Steve [
        ' GetCPULoad = 0 is idle, 1 is fully used.
        '  Multiply by 100 for a percentage
        PRINT
        PRINT "CPU used:              "; LTRIM$(STR$(INT((GetCPULoad * 10000) / 100))); "%"; SPACE$(18)
        PRINT "Memory used:           "; LTRIM$(STR$(MemInUsePercent)); "%"; SPACE$(18)
```

```
00,718          PRINT "Total Physical Memory: ";: PRINT AddCommas$(TotalPhysicalMem); " bytes"; SPACE$(18)
00,719          PRINT "Free Physical Memory:  ";: PRINT AddCommas$(FreePhysicalMem); " bytes"; SPACE$(18)
00,720          PRINT "Total Paging File:     ";: PRINT AddCommas$(TotalPagingFile); " bytes"; SPACE$(18)
00,721          PRINT "Free Paging File:      ";: PRINT AddCommas$(FreePagingFile); " bytes"; SPACE$(18)
00,722          '       PRINT "Total Virtual Memory:  ";: PRINT AddCommas$(TotalVirtualMem); " bytes"; SPACE$(18)
00,723          '       PRINT "Free Virtual Memory:   ";: PRINT AddCommas$(FreeVirtualMem); " bytes"; SPACE$(18)
00,724          ' MEMUSAGE by Steve ]
00,725      $End If
00,726
00,727      'IF COMMAND$ = "" THEN PRINT: PRINT "May press Alt+X or Alt+Q to eXit/Quit..."
00,728      ReturnCOMBO
00,729
00,730      If YdimROW = 60 Then
00,731          cryFrame = CsrLin
00,732          NextFrame cryFrame + 1
00,733          Locate cryFrame, 1
00,734      End If
00,735
00,736      _Display
00,737
00,738      $If WINDOWS Then
00,739          IF _TOTALDROPPEDFILES THEN
00,740          'FOR i = 1 TO _TOTALDROPPEDFILES
00,741          'a$ = _DROPPEDFILE(i)
00,742          a$ = _DROPPEDFILE(1)
00,743          'NEXT'
00,744          _FINISHDROP 'If _FINISHDROP isn't called here then _TOTALDROPPEDFILES never gets reset.
00,745          'ELSE
00,746          'a$ = "Scroller.$$$"
00,747          'SHELL _HIDE "DIR /B *.* > Scroller.$$$"
00,748          END IF
00,749      $End If
00,750      $If WINDOWS Then
00,751          IF COMMAND$ <> "" THEN a$ = _STARTDIR$ + "\" + COMMAND$
00,752      $Else
00,753          If Command$ <> "" Then a$ = _StartDir$ + "/" + Command$
00,754      $End If
00,755
00,756      'WrapFlag = 0
00,757      'Wwidth% = XdimCOL
00,758      'ul& = 0
00,759      'LongestLine = 0
00,760      'DO WHILE NOT EOF(1)
00,761      '    LINE INPUT #1, l$: ul& = ul& + 1
00,762      '    ExpandTabs l$
00,763      '    IF LEN(l$) > LongestLine THEN LongestLine = LEN(l$)
00,764      '    IF WrapFlag THEN
00,765      '        lX$ = l$
00,766      '        DO WHILE LEN(lX$) > Wwidth%
00,767      '            Glupak% = Wwidth%
00,768      '            DO UNTIL MID$(lX$, Glupak% + 1, 1) = " " AND MID$(lX$, Glupak%, 1) <> " "
00,769      '                Glupak% = Glupak% - 1
00,770      '                IF Glupak% = 0 THEN
00,771      '                    PRINT "Rejecting line(#"; LTRIM$(STR$(ul&)); ", "; LTRIM$(STR$(LEN(l$))); "chars) that cannot be wrapped!"
00,772      '                    _DISPLAY: SYSTEM
00,773      '                    GOTO B4Txpanar
00,774      '                END IF
00,775      '            LOOP
00,776      '            PRINT #3, LEFT$(lX$, Glupak%)
00,777      '            lX$ = STRING$(1, " ") + LTRIM$(MID$(lX$, Glupak% + 1, LEN(lX$) - (Glupak%)))
00,778      '        LOOP
00,779      '        PRINT #3, lX$
00,780      '        B4Txpanar:
00,781      '    END IF
00,782      'LOOP
00,783
00,784      SecondTime = 1
00,785      GettingStarted:
00,786
00,787      If WrapFlag = 1 Then
00,788          'Creating the wrapped temporary file [
00,789          If _FileExists(a$ + ".wrapped") = 0 Then
00,790              If _FileExists(a$) Then
00,791                  If SecondTime <> 2 Then Print: Print "Writing wrapped variant..."
00,792                  PostfixedToHeader = CsrLin
00,793
00,794                  StatuLine$ = Space$(XdimCOL)
00,795                  Mid$(StatuLine$, 1, 1) = "["
00,796                  Mid$(StatuLine$, XdimCOL, 1) = "]"
00,797
00,798                  _Display
00,799                  Open a$ For Binary As #1
00,800                  Open a$ + ".wrapped" For Output As #3
00,801                  Open a$ + ".unwrappable" For Output As #2
00,802                  LongestLine = 0
00,803                  UnwrappableLines = 0
00,804                  FileSize = LOF(1)
00,805                  If FileSize Then
00,806                      ChunkBeingOneUnit = FileSize \ (XdimCOL - 1 - 1) + 1: LastPercentage = 0
00,807                      NumberOfLFs = 0
00,808                      Wwidth% = XdimCOL
00,809                      ul& = 0
00,810                      ReadBytes = 0
00,811                      Do While Not EOF(1)
00,812                          Line Input #1, l$: ul& = ul& + 1: ReadBytes = ReadBytes + Len(l$)
00,813                          ExpandTabs l$
00,814                          If Len(l$) > LongestLine Then LongestLine = Len(l$)
00,815                          If WrapFlag Then
00,816                              ' 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:
```

```
00,817                          AssumeLineIsWrappable = 1
00,818                          lx$ = l$
00,819                          Do While Len(lx$) > Wwidth%
00,820                              Glupak% = Wwidth%
00,821                              Do 'UNTIL (MID$(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$, Glupak%, 1) <> " "
00,822                                  InvokeOnce$ = Mid$(lx$, Glupak% + 1, 1)
00,823                                  If InStr(" \/_:;,.-|+%=:", InvokeOnce$) And Mid$(lx$, Glupak%, 1) <> " " Then Exit Do
00,824                                  'IF (InvokeOnce$ = " " 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
00,825                                  Glupak% = Glupak% - 1
00,826                                  If Glupak% = 0 Then
00,827                                      AssumeLineIsWrappable = 0
00,828                                      GoTo B4TxpanarVIRTUAL
00,829                                  End If
00,830                              Loop
00,831                              lx$ = LTrim$(Mid$(lx$, Glupak% + 1, Len(lx$) - (Glupak%)))
00,832                          Loop
00,833                          B4TxpanarVIRTUAL:
00,834                          If AssumeLineIsWrappable = 1 Then
00,835                              lx$ = l$
00,836                              Do While Len(lx$) > Wwidth%
00,837                                  Glupak% = Wwidth%
00,838                                  Do 'UNTIL (MID$(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$, Glupak%, 1) <> " "
00,839                                      InvokeOnce$ = Mid$(lx$, Glupak% + 1, 1)
00,840                                      If InStr(" \/_:;,.-|+%=:", InvokeOnce$) And Mid$(lx$, Glupak%, 1) <> " " Then Exit Do
00,841                                      'IF (InvokeOnce$ = " " 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
00,842                                      Glupak% = Glupak% - 1
00,843                                      If Glupak% = 0 Then
00,844                                          Print "Rejecting line(#"; LTrim$(Str$(ul&)); ", "; LTrim$(Str$(Len(l$))); "chars) that cannot be wrapped!"
00,845                                          Close #1, #3: Kill a$ + ".wrapped"
00,846                                          _Display: End
00,847                                          GoTo B4Txpanar
00,848                                      End If
00,849                                  Loop
00,850                                  Print #3, Left$(lx$, Glupak%)
00,851                                  'lx$ = STRING$(1, " ") + LTRIM$(MID$(lx$, Glupak% + 1, LEN(lx$) - (Glupak%)))
00,852                                  lx$ = LTrim$(Mid$(lx$, Glupak% + 1, Len(lx$) - (Glupak%)))
00,853                              Loop
00,854                              Print #3, lx$
00,855                              B4Txpanar:
00,856                          Else
00,857                              Print #2, l$: UnwrappableLines = UnwrappableLines + 1
00,858                          End If
00,859                      End If
00,860                      If (ReadBytes \ ChunkBeingOneUnit) > LastPercentage Then
00,861                          LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
00,862                          Mid$(StatuLine$, 2, LastPercentage) = String$(LastPercentage, Chr$(176))
00,863                          Locate YdimROW + 1, 1: Color 9, 0: Print StatuLine$;
00,864                          _Display
00,865                      End If
00,866                  Loop
00,867              Else
00,868                  Close #1 'when filesize is 0
00,869                  System
00,870              End If 'IF FileSize THEN
00,871              Close #1, #3
00,872              a$ = a$ + ".wrapped"
00,873              Locate PostfixedToHeader, 1
00,874              Color NormalFRGr, NormalBCKGr
00,875          End If
00,876      Else
00,877          a$ = a$ + ".wrapped"
00,878      End If
00,879      'Creating the wrapped temporary file ]
00,880  End If
00,881
00,882  TimeA = Timer
00,883
00,884  ' Below fragment is r.4+ with 64bit pointers... to external/internal RAM!
00,885  ' On i5-7200u, SSD Kingston 256GB it loads OED in ????/??? seconds, respectively.
00,886  If _FileExists(a$) Then
00,887      If SecondTime <> 2 Then Print: Print "Loading...": Print
00,888
00,889      PostfixedToHeader = CsrLin
00,890
00,891      StatuLine$ = Space$(XdimCOL)
00,892      Mid$(StatuLine$, 1, 1) = "["
00,893      Mid$(StatuLine$, XdimCOL, 1) = "]"
00,894
00,895      _Display
00,896      Open a$ For Binary As #1
00,897      LongestLine = 0
00,898      FileSize = LOF(1)
00,899      If FileSize Then
00,900          ChunkBeingOneUnit = FileSize \ (XdimCOL - 1 - 1) + 1: LastPercentage = 0
00,901          NumberOfLFs = 0
00,902          If ToLoadOrNotFlag Then
00,903              Mwholefile = _MemNew(FileSize + 1)
00,904              'TheWholeFile$ = SPACE$(FileSize + 1) ' Grmbl, this comes with limitation probably 2GB, which prompts for buffered e.g. 128KB approach as my old LineWordReporter.c tool.
00,905              'GET #1, , TheWholeFile$
00,906              ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
00,907              chunk128KB$ = Space$(128 * 1024 * 1024)
00,908              Do While ReadBytes + (128 * 1024 * 1024) < FileSize
00,909                  Get #1, , chunk128KB$
00,910                  i = 1
00,911                  Do While i <= (128 * 1024 * 1024)
00,912                      i = InStr(i, chunk128KB$, Chr$(10))
```

```
00,913                      If i = 0 Then Exit Do
00,914                      NumberOfLFs = NumberOfLFs + 1
00,915                      i = i + 1
00,916                  Loop
00,917                  'FOR i = 1 TO LEN(chunk128KB$)
00,918                  '    IF MID$(chunk128KB$, i, 1) = CHR$(10) THEN NumberOfLFs = NumberOfLFs + 1 ' Boost it with INSTR...
00,919                  'NEXT i
00,920                  _MemPut Mwholefile, Mwholefile.OFFSET + ReadBytes, chunk128KB$
00,921                  ReadBytes = ReadBytes + (128 * 1024 * 1024)
00,922              Loop
00,923              If (FileSize - ReadBytes) Then
00,924                  RemainingChunk$ = Space$(FileSize - ReadBytes)
00,925                  Get #1, , RemainingChunk$
00,926                  i = 1
00,927                  Do While i <= Len(RemainingChunk$)
00,928                      i = InStr(i, RemainingChunk$, Chr$(10))
00,929                      If i = 0 Then Exit Do
00,930                      NumberOfLFs = NumberOfLFs + 1
00,931                      i = i + 1
00,932                  Loop
00,933                  'FOR i = 1 TO LEN(RemainingChunk$)
00,934                  '    IF MID$(RemainingChunk$, i, 1) = CHR$(10) THEN NumberOfLFs = NumberOfLFs + 1 ' Boost it with INSTR...
00,935                  'NEXT i
00,936                  _MemPut Mwholefile, Mwholefile.OFFSET + ReadBytes, RemainingChunk$
00,937              End If
00,938          End If
00,939          'Bugfix from 2021-Jan-29, inhere problem exists, in efficient mode we don't know 'NumberOfLFs' therefore should read-it-without-loading-it:
00,940          If ToLoadOrNotFlag = 0 Then
00,941              ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
00,942              chunk128KB$ = Space$(128 * 1024 * 1024)
00,943              Do While ReadBytes + (128 * 1024 * 1024) < FileSize
00,944                  Get #1, , chunk128KB$
00,945                  i = 1
00,946                  Do While i <= (128 * 1024 * 1024)
00,947                      i = InStr(i, chunk128KB$, Chr$(10))
00,948                      If i = 0 Then Exit Do
00,949                      NumberOfLFs = NumberOfLFs + 1
00,950                      i = i + 1
00,951                  Loop
00,952                  ReadBytes = ReadBytes + (128 * 1024 * 1024)
00,953              Loop
00,954              If (FileSize - ReadBytes) Then
00,955                  RemainingChunk$ = Space$(FileSize - ReadBytes)
00,956                  Get #1, , RemainingChunk$
00,957                  i = 1
00,958                  Do While i <= Len(RemainingChunk$)
00,959                      i = InStr(i, RemainingChunk$, Chr$(10))
00,960                      If i = 0 Then Exit Do
00,961                      NumberOfLFs = NumberOfLFs + 1
00,962                      i = i + 1
00,963                  Loop
00,964              End If
00,965          End If
00,966          Seek #1, 1
00,967          'IF ASC(Byte) < 32 AND Byte <> CHR$(10) AND Byte <> CHR$(13) AND Byte <> CHR$(9) THEN Byte = CHR$(32)
00,968          filecount = 0
00,969          'Ugh, buggy, in r.5 below two "malloc" lines should be using 'NumberOfLFs' not 'FileSize'...
00,970          MhandleOFF = _MemNew(8&& * (NumberOfLFs + 1)) 'create new memory block of 8*NumberOfLFs bytes - each line has its own 64bit Offset
00,971          MhandleLEN = _MemNew(8&& * (NumberOfLFs + 1)) 'create new memory block of 8*NumberOfLFs bytes - each line has its own 64bit Length, kinda overkill, could be 32bit
00,972
00,973          ' Parser for r.6 [[[
00,974          '           LineLen13 = 0
00,975          '           Byte = ""
00,976          '           'Caution: Many files DON'T end with CRLF, or LF or CR - kinda the last line is not always postixed correctly!
00,977          '           'This means if after the last CRLF, or LF or CR there is at least one byte then the number of total lines should be +1
00,978          '           FOR j = 1 TO FileSize + 1
00,979          '
00,980          '               IF (j \ ChunkBeingOneUnit) > LastPercentage THEN
00,981          '                   LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
00,982          '                   MID$(StatuLine$, 2, LastPercentage) = STRING$(LastPercentage, CHR$(176))
00,983          '                   LOCATE YdimROW + 1, 1: COLOR 9, 0: PRINT StatuLine$;
00,984          '                   _DISPLAY
00,985          '               END IF
00,986          '
00,987          '               PrevByte = Byte
00,988          '               IF j <= FileSize THEN
00,989          '                   IF ToLoadOrNotFlag THEN
00,990          '                       'Byte = MID$(TheWholeFile$, j, 1)
00,991          '                       _MEMGET Mwholefile, Mwholefile.OFFSET + (j - 1), Byte
00,992          '                   ELSE
00,993          '                       GET #1, , Byte
00,994          '                   END IF
00,995          '               ELSE
00,996          '                   Byte = ""
00,997          '                   IF PrevByte <> CHR$(10) THEN Byte = CHR$(10) ' enforce last line to be postfixed ONLY if last char was not LF
00,998          '               END IF
00,999          '               QWORD = j
01,000          '               IF Byte = CHR$(10) THEN 'For now, won't work for MacOS...
01,001          '                   QWORD = QWORD - LineLen13 'have to write the current offset minus the length of the line
01,002          '                   _MEMPUT MhandleOFF, MhandleOFF.OFFSET + 8&& * filecount, QWORD
01,003          '                   IF PrevByte = CHR$(13) THEN LineLen13 = LineLen13 - 1
01,004          '                   _MEMPUT MhandleLEN, MhandleLEN.OFFSET + 8&& * filecount, LineLen13
01,005          '                   IF LineLen13 > LongestLine THEN LongestLine = LineLen13
01,006          '                   LineLen13 = 0
01,007          '                   filecount = filecount + 1
01,008          '               ELSE
01,009          '                   LineLen13 = LineLen13 + 1
01,010          '               END IF
01,011          '           NEXT j
01,012          ' Parser for r.6 ]]]
```

```
01,013
01,014            ' Parser for r.7 [[[[[[
01,015            If ToLoadOrNotFlag = 0 Then
01,016                ReadBytes = 0 ' Have to load the 2+GB "malloc" in chunks...
01,017                ChunkLen = 128 * 1024
01,018                chunk128KB$ = Space$(ChunkLen) '+1 for sentinel
01,019                j = 1 ' is the current offset where new GET reads
01,020                QWORDlast = 1
01,021                PrevByte = ""
01,022                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)
01,023
01,024                    If (j \ ChunkBeingOneUnit) > LastPercentage Then
01,025
01,026                        If SecondTime <> 2 Then
01,027                            $If WINDOWS Then
01,028                                LOCATE PostfixedToHeader, 1
01,029                                ' MEMUSAGE by Steve [
01,030                                ' GetCPULoad = 0 is idle, 1 is fully used.
01,031                                '  Multiply by 100 for a percentage
01,032                                PRINT "CPU used:              "; LTRIM$(STR$(INT((GetCPULoad * 10000) / 100))); "%"; SPACE$(18)
01,033                                PRINT "Memory used:           "; LTRIM$(STR$(MemInUsePercent)); "%"; SPACE$(18)
01,034                                PRINT "Total Physical Memory: ";: PRINT AddCommas$(TotalPhysicalMem); " bytes"; SPACE$(18)
01,035                                PRINT "Free Physical Memory:  ";: PRINT AddCommas$(FreePhysicalMem); " bytes"; SPACE$(18)
01,036                                PRINT "Total Paging File:     ";: PRINT AddCommas$(TotalPagingFile); " bytes"; SPACE$(18)
01,037                                PRINT "Free Paging File:      ";: PRINT AddCommas$(FreePagingFile); " bytes"; SPACE$(18)
01,038                                PRINT "Total Virtual Memory:  ";: PRINT AddCommas$(TotalVirtualMem); " bytes"; SPACE$(18)
01,039                                PRINT "Free Virtual Memory:   ";: PRINT AddCommas$(FreeVirtualMem); " bytes"; SPACE$(18)
01,040                                ' MEMUSAGE by Steve ]
01,041                            $End If
01,042                        End If
01,043
01,044                        LastPercentage = LastPercentage + 1 ' in range 1..(XdimCOL-1)
01,045                        Mid$(StatuLine$, 2, LastPercentage) = String$(LastPercentage, Chr$(176))
01,046                        Locate YdimROW + 1, 1: Color 9, 0: Print StatuLine$;
01,047
01,048                        If YdimROW = 60 Then
01,049                            cryFrame = CsrLin
01,050                            NextFramePEN 1
01,051                            Locate cryFrame, 1
01,052                        End If
01,053
01,054                        _Display
01,055                    End If
01,056
01,057                    LastByte = Right$(chunk128KB$, 1) 'to handle eventual CR, left behind i.e. in previous chunk
01,058                    Get #1, j, chunk128KB$
01,059                    i = 1
01,060                    FoundAt = InStr(i, chunk128KB$, Chr$(10))
01,061                    If FoundAt Then
01,062                        Do While FoundAt
01,063                            If FoundAt = 1 Then PrevByte = LastByte Else PrevByte = Mid$(chunk128KB$, FoundAt - 1, 1)
01,064                            QWORD = (j - 1) + FoundAt
01,065                            LineLen13 = QWORD - QWORDlast
01,066                            _MemPut MhandleOFF, MhandleOFF.OFFSET + &&& * filecount, QWORDlast
01,067                            QWORDlast = QWORD + 1
01,068                            If PrevByte = Chr$(13) Then LineLen13 = LineLen13 - 1
01,069                            _MemPut MhandleLEN, MhandleLEN.OFFSET + &&& * filecount, LineLen13
01,070                            If LineLen13 > LongestLine Then LongestLine = LineLen13
01,071                            filecount = filecount + 1
01,072                            i = FoundAt + 1
01,073                            If i > (ChunkLen) Then Exit Do 'could use sentinel (buffer+1), in order this line to drop out
01,074                            FoundAt = InStr(i, chunk128KB$, Chr$(10))
01,075                        Loop
01,076                    End If
01,077                    j = j + (ChunkLen)
01,078                    ReadBytes = ReadBytes + (ChunkLen)
01,079                Loop
01,080                If (FileSize - ReadBytes) Then
01,081                    RemainingChunk$ = Space$(FileSize - ReadBytes) '+1 for sentinel
01,082                    LastByte = Right$(chunk128KB$, 1) 'to handle eventual CR, left behind i.e. in previous chunk
01,083                    Get #1, , RemainingChunk$
01,084                    If Right$(RemainingChunk$, 1) <> Chr$(10) Then RemainingChunk$ = RemainingChunk$ + Chr$(10) ' dirty, enforcing not missing the last line (if it is not postfixed with
LF)
01,085                    'Beware, yes be aware that above line should have been applied for above/first fragment because the filesize could be multiple of the chunk length i.e. no remaining
chunk, however it was feinted by '<'
01,086                    i = 1
01,087                    FoundAt = InStr(i, RemainingChunk$, Chr$(10))
01,088                    If FoundAt Then
01,089                        Do While FoundAt
01,090                            If FoundAt = 1 Then PrevByte = LastByte Else PrevByte = Mid$(RemainingChunk$, FoundAt - 1, 1)
01,091                            QWORD = (j - 1) + FoundAt
01,092                            LineLen13 = QWORD - QWORDlast
01,093                            _MemPut MhandleOFF, MhandleOFF.OFFSET + &&& * filecount, QWORDlast
01,094                            QWORDlast = QWORD + 1
01,095                            If PrevByte = Chr$(13) Then LineLen13 = LineLen13 - 1
01,096                            _MemPut MhandleLEN, MhandleLEN.OFFSET + &&& * filecount, LineLen13
01,097                            If LineLen13 > LongestLine Then LongestLine = LineLen13
01,098                            filecount = filecount + 1
01,099                            i = FoundAt + 1
01,100                            If i > (FileSize - ReadBytes) Then Exit Do 'could use sentinel (buffer+1), in order this line to drop out
01,101                            FoundAt = InStr(i, RemainingChunk$, Chr$(10))
01,102                        Loop
01,103                    End If
01,104                End If
01,105            End If
01,106            Seek #1, 1
01,107            ' Parser for r.7 ]]]]]]
01,108
01,109        Else
```

```
01,110            Close #1 'when filesize is 0
01,111            System
01,112        End If 'IF FileSize THEN
01,113    End If
01,114
01,115    ' Below fragment is r.4 with 64bit pointers... to external RAM!
01,116    ' 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.
01,117    'IF _FILEEXISTS(a$) THEN
01,118    '    PRINT: PRINT "Loading..."
01,119    '    _DISPLAY
01,120    '    OPEN a$ FOR BINARY AS #1
01,121    '    FileSize = LOF(1)
01,122    '    filecount = 0
01,123    '    MhandleOFF = _MEMNEW(&&& * FileSize) 'create new memory block of 8*filesize bytes - each line has its own 64bit Offset
01,124    '    MhandleLEN = _MEMNEW(&&& * FileSize) 'create new memory block of 8*filesize bytes - each line has its own 64bit Length, kinda overkill, could be 32bit
01,125    '    LineLen13 = 0
01,126    '    Byte = ""
01,127    '    'Caution: Many files DON'T end with CRLF, or LF or CR - kinda the last line is not always postixed correctly!
01,128    '    '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
01,129    '    FOR j = 1 TO FileSize + 1
01,130    '        PrevByte = Byte
01,131    '        IF j <= FileSize THEN
01,132    '            GET #1, , Byte
01,133    '        ELSE
01,134    '            Byte = ""
01,135    '            IF PrevByte <> CHR$(10) THEN Byte = CHR$(10) ' enforce last line to be postfixed ONLY if last char was not LF
01,136    '        END IF
01,137    '        QWORD = j
01,138    '        IF Byte = CHR$(10) THEN 'For now, won't work for MacOS...
01,139    '            QWORD = QWORD - LineLen13 'have to write the current offset minus the length of the line
01,140    '            _MEMPUT MhandleOFF, MhandleOFF.OFFSET + &&& * filecount, QWORD
01,141    '            IF PrevByte = CHR$(13) THEN LineLen13 = LineLen13 - 1
01,142    '            _MEMPUT MhandleLEN, MhandleLEN.OFFSET + &&& * filecount, LineLen13
01,143    '            IF LineLen13 > LongestLine THEN LongestLine = LineLen13
01,144    '            LineLen13 = 0
01,145    '            filecount = filecount + 1
01,146    '        ELSE
01,147    '            LineLen13 = LineLen13 + 1
01,148    '        END IF
01,149    '    NEXT j
01,150    'END IF
01,151
01,152    ''PRINT "filecount= "; filecount
01,153    ''PRINT "LongestLine= "; LongestLine
01,154    ''exercise in reproducing the indexed file:
01,155    ''Debugo = CHR$(13) + CHR$(10)
01,156    'Debugo = CHR$(10)
01,157    'OPEN a$ + ".dump" FOR BINARY AS #2
01,158    'FOR i = 1 TO filecount
01,159    '    _MEMGET MhandleOFF, MhandleOFF.OFFSET + &&& * (i - 1), QWORD
01,160    '    _MEMGET MhandleLEN, MhandleLEN.OFFSET + &&& * (i - 1), LineLen13
01,161    '    PRINT QWORD, LineLen13:
01,162    '    SEEK #1, QWORD
01,163    '    BufferForLine$ = ""
01,164    '    FOR j = 1 TO LineLen13
01,165    '        GET #1, , Byte 'grmbl, how to load in one call?! As fread() is capable.
01,166    '        PUT #2, , Byte
01,167    '        BufferForLine$ = BufferForLine$ + Byte
01,168    '    NEXT j
01,169    '    PUT #2, , Debugo
01,170    '    PRINT BufferForLine$
01,171    'NEXT i
01,172    '_DISPLAY
01,173    'END
01,174
01,175    ' Commented fragment is r.3+ with slow strings
01,176    'LongestLine = 0
01,177    'IF _FILEEXISTS(a$) THEN
01,178    '    PRINT: PRINT "Loading..."
01,179    '    _DISPLAY
01,180    '    filecount = 0
01,181    '    '       OPEN a$ FOR INPUT AS #1
01,182    '    OPEN a$ FOR BINARY AS #1 ' As suggested by SMcNeill and Pete in https://www.qb64.org/forum/index.php?topic=3518.msg128631#msg128631
01,183    '    DO UNTIL EOF(1)
01,184    '        LINE INPUT #1, l$
01,185    '        filecount = filecount + 1
01,186    '    LOOP
01,187    '    CLOSE #1
01,188    '    REDIM FileArray$(1 TO filecount)
01,189    '    filecount = 0
01,190    '    '       OPEN a$ FOR INPUT AS #1
01,191    '    OPEN a$ FOR BINARY AS #1 ' As suggested by SMcNeill and Pete in https://www.qb64.org/forum/index.php?topic=3518.msg128631#msg128631
01,192    '    FileSize = LOF(1)
01,193    '    DO UNTIL EOF(1)
01,194    '        LINE INPUT #1, l$
01,195    '        ExpandTabs l$
01,196    '        FOR j = 1 TO LEN(l$)
01,197    '            IF MID$(l$, j, 1) < CHR$(32) THEN MID$(l$, j, 1) = CHR$(32)
01,198    '        NEXT j
01,199    '        IF LEN(l$) > LongestLine THEN LongestLine = LEN(l$)
01,200    '        filecount = filecount + 1
01,201    '        FileArray$(filecount) = l$
01,202    '    LOOP
01,203    '    CLOSE #1
01,204    'END IF
01,205
01,206    _Limit 5
01,207    If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(81) Or _KeyDown(113)) Then System
01,208    If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(88) Or _KeyDown(120)) Then System
01,209
```

```
01,210      '                           QB64 _KEYHIT and _KEYDOWN Values
01,211      '
01,212      'Esc  F1    F2    F3    F4    F5    F6    F7    F8    F9    F10   F11   F12   Sys  ScL Pause
01,213      ' 27 15104 15360 15616 15872 16128 16384 16640 16896 17152 17408 34048 34304 +316 +302 +019
01,214      '`~  1!  2@  3#  4$  5%  6^  7&  8*  9(  0) -_ =+ BkSp   Ins   Hme   PUp   NumL   /     *    -
01,215      '126 33  64  35  36  37  94  38  42  40  41 95 43   8   20992 18176 18688 +300   47    42   45
01,216      ' 96 49  50  51  52  53  54  55  56  57  48 45 61
01,217      'Tab Q   W   E   R   T   Y   U   I   O   P  [{  ]} \|   Del   End   PDn   7Hme  8/?   9PU   +
01,218      ' 9  81  87  69  82  84  89  85  73  79  80 123 125 124 21248 20224 20736 18176 18432 18688 43
01,219      '   113 119 101 114 116 121 117 105 111 112  91  93  92                    55    56    57
01,220      'CapL   A   S   D   F   G   H   J   K   L   ;:  '" Enter                  4/?-   5    6/-?
01,221      '+301   65  83  68  70  71  72  74  75  76  58  34  13                   19200 19456 19712  E
01,222      '       97 115 100 102 103 104 106 107 108  59  39                     52    53    54    n
01,223      'Shift   Z   X   C   V   B   N   M   ,<  .>  /?    Shift       ?          1End  2/?   3PD   t
01,224      '+304   90  88  67  86  66  78  77  60  62  63    +303       18432       20224 20480 20736  e
01,225      '      122 120  99 118  98 110 109  44  46  47                           49    50    51    r
01,226      'Ctrl   Win  Alt     Spacebar      Alt  Win  Menu  Ctrl   ?-   ?   -?      0Ins        .Del
01,227      '+306  +311 +308       32         +307 +312 +319  +305 19200 20480 19712  20992       21248 13
01,228      '                                                                        48          46
01,229      '         Lower value = LCase/NumLock On _________________ + = add 100000
01,230
01,231 Loop While a$ = ""
01,232
01,233 If XdimCOL = 128 And YdimROW = 40 Then _Title "MASAKARI, revision 8+ 128x40, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,234 If XdimCOL = 128 And YdimROW = 60 Then _Title "MASAKARI, revision 8+ 128x60, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,235 If XdimCOL = 198 And YdimROW = 60 Then _Title "MASAKARI, revision 8+ 198x60, The 'Holy Axe' English Text Sidekick" + ": " + a$
01,236 LoadedFile$ = a$
01,237
01,238 TimeB = Timer: SecondTime = 2
01,239 'PLAY "L8V2a-c-" 'removed in order to get rid of audio dependency
01,240 CurrentLine = 1
01,241 File_Frame_x = 1
01,242 File_Frame_y = 1
01,243 Color NormalFRGr, NormalBCKGr
01,244 UpdateWindowFrame NormalFRGr, NormalBCKGr
01,245
01,246 ' Here is the layout:
01,247 ' The window-frame is 1..crx or 1..128 |  The file-frame is 1..LongestLine
01,248 '                     .        .     |                  .
01,249 '                     .        .     |                  .
01,250 '                     cry      60    |                  filecount
01,251 '             FileArrayWINDOW$(60)   |         FileArray(filecount)
01,252 ' if File_Frame_x < 128 then PADDING to 128 else File_Frame_x = 1..LongestLine-(128-1)
01,253 ' if File_Frame_y < 60 then PADDING to 60 else File_Frame_y = 1..filecount-(60-1)
01,254
01,255 UpdateCline 1, 1, InverseFRGr, InverseBCKGr, 1
01,256 StatuLine$ = Space$(XdimCOL)
01,257 Mid$(StatuLine$, 1, 1) = "["
01,258 Mid$(StatuLine$, XdimCOL, 1) = "]"
01,259
01,260 FieldLineNum = Len(AddCommas$(filecount))
01,261 FieldLineLen = Len(AddCommas$(LongestLine))
01,262
01,263 Dumbo$ = "File Size: " + AddCommas$(FileSize) + "; Longest Line: " + AddCommas$(LongestLine)
01,264 Mid$(StatuLine$, 2, Len(Dumbo$)) = Dumbo$
01,265 Locate YdimROW + 1, 1: Color 9, 0: Print StatuLine$;
01,266 UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1 ' +1 due to [
01,267 MostRightField = Len(Dumbo$) + 1 + 1 + Len("; Line Number: ") + Len("; Line Length: ") + FieldLineNum + FieldLineLen
01,268 UpdateNextToCCL_DONE MostRightField
01,269
01,270 If WrapFlag = 1 Then
01,271     UpdateNextToCCL_UNWRAPPABLEcount MostRightField
01,272 End If
01,273
01,274 _Display
01,275 Locate 1, 1, 1, 30, 31
01,276 crx = Pos(0)
01,277 cry = CsrLin
01,278 crxOLD = crx
01,279 cryOLD = cry
01,280 PageScrollBenchmark = 0
01,281 LineScrollBenchmark = 0
01,282
01,283 tCurrentT1# = Timer(.001)
01,284 tLASTrelease1# = tCurrentT1#
01,285 tLASTpress1# = tCurrentT1#
01,286 tCurrentT2# = Timer(.001)
01,287 tLASTrelease2# = tCurrentT2#
01,288 tLASTpress2# = tCurrentT2#
01,289
01,290 Do
01,291     ReturnCOMBO
01,292     If LALT_RALT Then
01,293         'PLAY "L4V2f" 'removed in order to get rid of audio dependency
01,294         _KeyClear
01,295         ReportTimeToLoad MostRightField
01,296         _Display
01,297     End If
01,298     If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(81) Or _KeyDown(113)) Then System
01,299     If (_KeyDown(LALTkey&) Or _KeyDown(RALTkey&)) And (_KeyDown(88) Or _KeyDown(120)) Then System
01,300
01,301     'IF _KEYDOWN(LCTRLkey&) AND (_KEYDOWN(83) OR _KEYDOWN(115)) THEN 'Ctrl+S
01,302     If (_KeyDown(RSHIFTkey&) And _KeyDown(LSHIFTkey&) = 0) Or (_KeyDown(RCTRLkey&) And _KeyDown(LCTRLkey&) = 0) And PageScrollBenchmark = 0 Then
01,303         InverseL = CsrLin
01,304
01,305         '[[[[[[[[[[[[[[[[[[[[[[[[
01,306
01,307         FiCbig$ = UCase$(Chr$(97))
01,308         SeCbig$ = UCase$(Chr$(122))
01,309
```

```
01,310          For i = 1 To MIN&(YdimROW, filecount)
01,311              'IF LEN(FileArray$(i + (File_Frame_y - 1))) >= XdimCOL THEN
01,312              '    FileArrayWINDOW$(i) = MID$(FileArray$(i + (File_Frame_y - 1)), 1, XdimCOL)
01,313              'ELSE
01,314              '    FileArrayWINDOW$(i) = FileArray$(i + (File_Frame_y - 1)) + SPACE$(XdimCOL - LEN(FileArray$(i + (File_Frame_y - 1))))
01,315              'END IF
01,316
01,317              'LOCATE I, 1: PRINT FileArrayWINDOW$(I);
01,318              Moron$ = FileArrayWINDOW$(i)
01,319              BigMoron$ = Space$(Len(Moron$))
01,320
01,321              SuperSlow$ = UCase$(Moron$)
01,322              Matrox% = Len(SuperSlow$)
01,323              j% = 0
01,324              wrd$ = ""
01,325              Do
01,326                  Again2:
01,327                  j% = j% + 1
01,328                  IamStupid$ = Mid$(SuperSlow$, j%, 1)
01,329                  If IamStupid$ >= FiCbig$ And IamStupid$ <= SeCbig$ Then
01,330                      wrd$ = wrd$ + IamStupid$
01,331                  End If
01,332                  If wrd$ = "" And j% < Matrox% Then GoTo Again2
01,333                  If wrd$ = "" And j% = Matrox% Then GoTo Nothing2
01,334                  If wrd$ <> "" And j% < Matrox% Then
01,335                      IamStupid2$ = Mid$(SuperSlow$, j% + 1, 1)
01,336                      If IamStupid2$ >= FiCbig$ And IamStupid2$ <= SeCbig$ Then GoTo Again2
01,337                  End If
01,338
01,339                  WRDpresent% = 0
01,340                  Open "masakari.wrd" For Binary As #6
01,341                  If LOF(6) Then
01,342                      Close #6
01,343                      Open "masakari.ind" For Random Access Read As #6 Len = 32
01,344                      LeftPoint& = 1
01,345                      RightPoint& = TotalWrd&
01,346                      Do
01,347                          If RightPoint& - LeftPoint& <= 1 Then 'Sxsedtcheta
01,348                              Get #6, LeftPoint&, lfixed$
01,349                              If wrd$ = UCase$(RTrim$(lfixed$)) Then WRDpresent% = 1
01,350                              Get #6, RightPoint&, lfixed$
01,351                              If wrd$ = UCase$(RTrim$(lfixed$)) Then WRDpresent% = 1
01,352                              Exit Do
01,353                          End If
01,354                          SplitPoint& = (LeftPoint& + RightPoint&) \ 2
01,355                          Get #6, SplitPoint&, lfixed$
01,356                          If wrd$ < UCase$(RTrim$(lfixed$)) Then
01,357                              RightPoint& = SplitPoint&
01,358                          ElseIf wrd$ > UCase$(RTrim$(lfixed$)) Then
01,359                              LeftPoint& = SplitPoint&
01,360                          Else
01,361                              WRDpresent% = 1: Exit Do
01,362                          End If
01,363                      Loop
01,364                      Close #6
01,365                  Else
01,366                      Close #6
01,367                      Kill "masakari.wrd"
01,368                      WRDpresent% = 2
01,369                  End If
01,370                  If WRDpresent% = 0 Then
01,371                      For Underdog% = j% - Len(wrd$) + 1 To j%
01,372                          Mid$(BigMoron$, Underdog%, 1) = "#"
01,373                      Next
01,374                  End If
01,375
01,376                  wrd$ = ""
01,377                  Nothing2:
01,378              Loop While j% < Matrox%
01,379
01,380              For j% = 1 To XdimCOL
01,381                  If Mid$(BigMoron$, j%, 1) <> " " Then
01,382                      If i = InverseL Then ' inverse line
01,383                          Locate i, j%: Color 15, InverseBCKGr: Print Mid$(Moron$, j%, 1);
01,384                      Else
01,385                          Locate i, j%: Color 15, 0: Print Mid$(Moron$, j%, 1);
01,386                      End If
01,387                  End If
01,388              Next
01,389              'CALL MovePage(VideoBuffer%, 0, 8)
01,390          Next
01,391
01,392          ']]]]]]]]]]]]]]]]]]]]]]
01,393          $If WINDOWS Then
01,394              PLAY "v20l20g"
01,395          $End If
01,396          Do Until _KeyDown(RCTRLkey&) = 0 And _KeyDown(RSHIFTkey&) = 0
01,397              _Display 'Caramba, when this line is commented the loop is not working as it should?!
01,398          Loop
01,399      End If
01,400
01,401      If _KeyDown(15616) And _KeyDown(LCTRLkey&) Then 'LCtrl+F3
01,402          'CALL DrawBox(3, 3, 7, 44, 8, 7)
01,403          'CALL DrawBoxShadow(3, 3, 7, 44)
01,404          'In case of YdimROW = 40 then shrunk panel
01,405          If YdimROW = 40 Then shrunkP = 14 Else shrunkP = 0
01,406          Call DrawBoxShadow3(3, 3, 48 - shrunkP, 107, "LCtrl+F3: Search Panel")
01,407          UpdateWindowFrame NormalFRGr, NormalBCKGr
01,408          _Display
01,409      End If
```

```
01,410
01,411      If _KeyDown(27) Then
01,412          LineScrollBenchmark = 0
01,413          PageScrollBenchmark = 0
01,414      End If
01,415
01,416      If LSHIFT_RSHIFT Or LineScrollBenchmark = 1 Then 'same as scroll page-by-page benchmark, but line-by-line
01,417          If LineScrollBenchmark = 0 Then TimeScrollA = Timer
01,418          LineScrollBenchmark = 1
01,419          'PgDn - just add the page height i.e. 'YdimROW' to 'File_Frame_y'
01,420          ' Don't execute PgDn (advancing the 'CurrentLine') if 'File_Frame_y' is not "eligible":
01,421          'IF File_Frame_y < filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
01,422          'IF File_Frame_y + 1 <= filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
01,423          'CAUTION [
01,424          ' Next three line beep NOT
01,425          'DEFLNG A-Z
01,426          'aaa = 61
01,427          'bbb = -48
01,428          'IF aaa <= bbb THEN BEEP: END
01,429          'Next three line beep
01,430          'aaa~& = 61
01,431          'bbb~& = -48
01,432          'IF aaa~& <= bbb~& THEN BEEP: END
01,433          'Next three line beep NOT
01,434          'aaa& = 61
01,435          'bbb& = -48
01,436          'IF aaa& <= bbb& THEN BEEP: END
01,437          'Next line works even when unsigned!
01,438          'IF File_Frame_y + YdimROW - (filecount - (YdimROW - 1)) <= 0 THEN
01,439          'CAUTION ]
01,440          'Next line doesn't work when unsigned!
01,441          If File_Frame_y + 1 <= filecount - (YdimROW - 1) Then
01,442              File_Frame_y = File_Frame_y + 1
01,443              'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,444              'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,445              If CurrentLine + 1 <= filecount Then CurrentLine = CurrentLine + 1
01,446              UpdateWindowFrame NormalFRGr, NormalBCKGr
01,447          Else
01,448              If LineScrollBenchmark = 1 Then
01,449                  TimeScrollB = Timer
01,450                  _KeyClear
01,451                  ReportTimeToScroll MostRightField
01,452                  LineScrollBenchmark = 0
01,453                  _Display
01,454              End If
01,455          End If
01,456      End If
01,457
01,458      ' Esc  F1  F2  F3  F4  F5  F6  F7  F8  F9  F10  F11  F12  Sys ScL Pause
01,459      '  27 +59 +60 +61 +62 +63 +64 +65 +66 +67 +68  +133 +134   -   -    -
01,460      ' `~  1!  2@  3#  4$  5%  6^  7&  8*  9(  0) -_ =+ BkSp   Ins Hme PUp   NumL  /   *    -
01,461      ' 126 33  64  35  36  37  94  38  42  40  41 95 43   8    +82 +71 +73    -    47  42   45
01,462      '  96 49  50  51  52  53  54  55  56  57  48 45 61
01,463      ' Tab Q   W   E   R   T   Y   U   I   O   P  [{  ]} \|   Del End PDn   7Hme 8/  9PU  +
01,464      '  9  81  87  69  82  84  89  85  73  79  80 123 125 124  +83 +79 +81   +71  +72  +73  43
01,465      '    113 119 101 114 116 121 117 105 111 112  91  93  92                55   56   57
01,466      ' CapL  A   S   D   F   G   H   J   K   L   ;:  '" Enter               4/-  5   6/-
01,467      '   -   65  83  68  70  71  72  74  75  76  58  34  13                +75  +76  +77  E
01,468      '       97 115 100 102 103 104 106 107 108  59  39                     52   53   54  n
01,469      '
01,470      ' Shift  Z   X   C   V   B   N   M   ,<  .>  /?    Shift             1End 2/  3PD  t
01,471      '   *    90  88  67  86  66  78  77  60  62  63      *       +72       +79  +80  +81  e
01,472      '       122 120  99 118  98 110 109  44  46  47                       49   50   51  r
01,473      ' Ctrl Win Alt       Spacebar         Alt Win Menu Ctrl   -     -   0Ins     .Del
01,474      '  *    -   *          32             *   -   -    *    +75 +80 +77   +82       +83  13
01,475      '                                                                    48        46
01,476      '
01,477      '     Italics = LCase/NumLock On  __________  + = 2 Byte: CHR$(0) + CHR$(code)
01,478      'NOTE: The above commented table can be copied and pasted directly into the QB64 IDE
01,479
01,480      key$ = InKey$
01,481      'DO: a$ = INKEY$: LOOP UNTIL a$ <> "" ' prevent ASC empty string read error
01,482      If key$ <> "" Then
01,483          code% = Asc(key$):
01,484          If code% Then ' ASC returns any value greater than 0
01,485              Select Case Asc(key$)
01,486                  Case 32:
01,487                      a$ = RTrim$(FileArray$(CurrentLine))
01,488                      If _FileExists(a$) Then
01,489                          Close #1
01,490                          _MemFree MhandleOFF
01,491                          _MemFree MhandleLEN
01,492                          If ToLoadOrNotFlag Then
01,493                              _MemFree Mwholefile
01,494                          End If
01,495                          GoTo GettingStarted 'Before going above lines must be executed, i.e. to initialize.
01,496                      End If
01,497                  Case 65 TO 97: 'PRINT key$;
01,498                  Case Asc("a") TO Asc("z"): 'PRINT key$;
01,499                      'CASE 27: COLOR 7, 0: SYSTEM 'END
01,500              End Select
01,501          Else
01,502              Select Case Asc(key$, 2)
01,503                  Case 59:
01,504                      Locate 1, 1, 0, 30, 31
01,505                      For i = 1 To YdimROW
01,506                          Locate i, 1: Print Space$(XdimCOL);
01,507                      Next
01,508                      Locate 1, 1
01,509                      ShowF1
```

```
01,510                  Print: Print "Press ESC..."
01,511                  _Display
01,512                  $If WINDOWS Then
01,513                      PLAY "v20l20g"
01,514                  $End If
01,515                  Do While InKey$ <> Chr$(27)
01,516                      _Limit 30
01,517                  Loop
01,518                  UpdateWindowFrame NormalFRGr, NormalBCKGr
01,519              Case 72:
01,520                  If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
01,521                  If cry > 1 Then
01,522                      cry = cry - 1 'up
01,523                  Else 'scrolling is needed
01,524                      If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
01,525                      UpdateWindowFrame NormalFRGr, NormalBCKGr
01,526                  End If
01,527              Case 80:
01,528                  If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
01,529                  If cry < YdimROW Then
01,530                      If cry < filecount Then cry = cry + 1 'down
01,531                  Else 'scrolling is needed
01,532                      If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
01,533                      UpdateWindowFrame NormalFRGr, NormalBCKGr
01,534                  End If
01,535              Case 75: If crx > 1 Then crx = crx - 1 'left
01,536              Case 77: If crx < XdimCOL Then crx = crx + 1 'right
01,537              Case 73: 'PgUp
01,538                  If File_Frame_y - YdimROW >= 1 Then
01,539                      File_Frame_y = File_Frame_y - YdimROW
01,540                      If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
01,541                      UpdateWindowFrame NormalFRGr, NormalBCKGr
01,542                  End If
01,543              Case 81: 'PgDn
01,544                  If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
01,545                      File_Frame_y = File_Frame_y + YdimROW
01,546                      'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,547                      'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,548                      If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
01,549                      UpdateWindowFrame NormalFRGr, NormalBCKGr
01,550                  End If
01,551          End Select
01,552      End If
01,553  End If
01,554  If cryOLD <> cry Then
01,555      UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
01,556      cryOLD = cry
01,557  Else 'it 'cry' could be changed by Mouse Wheel too, check it
01,558      AsIfItIsINKEY% = _MouseInput '      Check the mouse status
01,559      buttondown1 = _MouseButton(1)
01,560      buttondown2 = _MouseButton(2)
01,561      buttondown3 = _MouseButton(3)
01,562      mwheel = _MouseWheel
01,563
01,564      If _KeyDown(LSHIFTkey&) And buttondown1 Then 'Fast Up
01,565          If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
01,566          If cry > 1 Then
01,567              cry = cry - 1 'up
01,568          Else 'scrolling is needed
01,569              If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
01,570              UpdateWindowFrame NormalFRGr, NormalBCKGr
01,571          End If
01,572      End If
01,573
01,574      If _KeyDown(LSHIFTkey&) And buttondown2 Then 'Fast Down
01,575          If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
01,576          If cry < YdimROW Then
01,577              If cry < filecount Then cry = cry + 1 'down
01,578          Else 'scrolling is needed
01,579              If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
01,580              UpdateWindowFrame NormalFRGr, NormalBCKGr
01,581          End If
01,582      End If
01,583
01,584      If _KeyDown(LALTkey&) And buttondown1 Then 'LCTRL_HOME
01,585          LCTRL_HOME = 1
01,586      End If
01,587      If _KeyDown(LALTkey&) And buttondown2 Then 'LCTRL_END
01,588          LCTRL_END = 1
01,589      End If
01,590
01,591      If _KeyDown(LCTRLkey&) And buttondown1 Then 'Fast PgUp
01,592          If File_Frame_y - YdimROW >= 1 Then
01,593              File_Frame_y = File_Frame_y - YdimROW
01,594              If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
01,595              UpdateWindowFrame NormalFRGr, NormalBCKGr
01,596          End If
01,597      End If
01,598
01,599      If _KeyDown(LCTRLkey&) And buttondown2 Then 'Fast PgDn, grmbl, for some reason (LShift+button3) and (LShift+button1) are not spitting?!
01,600          If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
01,601              File_Frame_y = File_Frame_y + YdimROW
01,602              'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,603              'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,604              If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
01,605              UpdateWindowFrame NormalFRGr, NormalBCKGr
01,606          End If
01,607      End If
01,608
01,609      If mwheel = 1 Then ' as if Down
```

```
01,610              If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
01,611              If cry < YdimROW Then
01,612                  If cry < filecount Then cry = cry + 1 'down
01,613              Else 'scrolling is needed
01,614                  If File_Frame_y < filecount - (YdimROW - 1) Then File_Frame_y = File_Frame_y + 1
01,615                  UpdateWindowFrame NormalFRGr, NormalBCKGr
01,616              End If
01,617          End If
01,618          If mwheel = -1 Then ' as if Up
01,619              If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
01,620              If cry > 1 Then
01,621                  cry = cry - 1 'up
01,622              Else 'scrolling is needed
01,623                  If File_Frame_y > 1 Then File_Frame_y = File_Frame_y - 1
01,624                  UpdateWindowFrame NormalFRGr, NormalBCKGr
01,625              End If
01,626          End If
01,627          If (_KeyDown(LSHIFTkey&) = 0) And buttondown1 Then
01,628              crxOLD = crx
01,629              cryOLD = cry
01,630              If _MouseY <= MIN&(YdimROW, filecount) Then
01,631                  cry = _MouseY
01,632                  crx = _MouseX
01,633                  UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
01,634                  Do While cryOLD > cry
01,635                      cryOLD = cryOLD - 1
01,636                      If CurrentLine > 1 Then CurrentLine = CurrentLine - 1
01,637                  Loop
01,638                  Do While cryOLD < cry
01,639                      cryOLD = cryOLD + 1
01,640                      If CurrentLine < filecount Then CurrentLine = CurrentLine + 1
01,641                  Loop
01,642              End If
01,643          End If
01,644
01,645          If (mwheel = -1) And buttondown2 Then ' same as LCTRL_HOME
01,646              LCTRL_HOME = 1
01,647          End If
01,648          If (mwheel = 1) And buttondown2 Then ' same as LCTRL_END
01,649              LCTRL_END = 1
01,650          End If
01,651
01,652          If buttondown2 Then ' 'drawing a line' gesture - 100 cells at least long, COMBO: Alt+X, Alt+Q
01,653              ReadOnceY = _MouseY
01,654              ReadOnceX = _MouseX
01,655              If Button2Down = 0 Then Mouse2Press! = Timer: Button2Down = 1
01,656              If WidenessINIT = 0 Then WidenessINIT = ReadOnceX
01,657              If HighnessINIT = 0 Then HighnessINIT = ReadOnceY
01,658              If ReadOnceY < YdimROW + 1 Then Locate ReadOnceY, ReadOnceX: Print Chr$(176); 'don't write trail in status line
01,659          Else
01,660              Button2Down = 0
01,661              Mouse2Release! = Timer
01,662              If WidenessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
01,663                  If Abs(_MouseX - WidenessINIT) >= 100 Then System 'PLAY "v10l8g" 'bidirectional i.e. can be drawn from left to right and vice versa
01,664                  WidenessINIT = 0
01,665              End If
01,666              'simulate PgDn (from bottom to top, as the hand cursor in PhotoShop), but not repetitive i.e. not-buffered
01,667              If HighnessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
01,668                  If (HighnessINIT - _MouseY) >= 5 Then 'PLAY "v10l8g" 'bidirectional i.e. can be drawn from left to right and vice versa
01,669
01,670                      If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
01,671                          File_Frame_y = File_Frame_y + YdimROW
01,672                          'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,673                          'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,674                          If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
01,675                          UpdateWindowFrame NormalFRGr, NormalBCKGr
01,676                      End If
01,677                      ' Here is the layout:
01,678                      ' The window-frame is 1..crx or 1..128 |  The file-frame is 1..LongestLine
01,679                      '                       .       .      |                  .
01,680                      '                       .       .      |                  .
01,681                      '                       cry     60     |                  filecount
01,682                      '             FileArrayWINDOW$(60)     |         FileArray(filecount)
01,683                      ' if File_Frame_x < 128 then PADDING to 128 else File_Frame_x = 1..LongestLine-(128-1)
01,684                      ' if File_Frame_y < 60 then PADDING to 60 else File_Frame_y = 1..filecount-(60-1)
01,685
01,686                  End If
01,687              End If
01,688              'simulate PgUp (from top to bottom, as the hand cursor in PhotoShop), but not repetitive i.e. not-buffered
01,689              If HighnessINIT And (Mouse2Release! - Mouse2Press! > 0 And Mouse2Release! - Mouse2Press! < 2) Then
01,690                  If (_MouseY - HighnessINIT) >= 5 Then 'PLAY "v10l8g" 'bidirectional i.e. can be drawn from left to right and vice versa
01,691
01,692                      If File_Frame_y - YdimROW >= 1 Then
01,693                          File_Frame_y = File_Frame_y - YdimROW
01,694                          If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
01,695                          UpdateWindowFrame NormalFRGr, NormalBCKGr
01,696                      End If
01,697
01,698                  End If
01,699              End If
01,700              HighnessINIT = 0
01,701          End If
01,702
01,703          If LCTRL_RCTRL Then PageScrollBenchmark = 1: TimeScrollA = Timer
01,704          If buttondown3 Or PageScrollBenchmark Then 'PgDn - just add the page height i.e. 'YdimROW' to 'File_Frame_y'
01,705              ' Don't execute PgDn (advancing the 'CurrentLine') if 'File_Frame_y' is not "eligible":
01,706              'IF File_Frame_y < filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
01,707              'IF File_Frame_y + 1 <= filecount - (YdimROW - 1) THEN File_Frame_y = File_Frame_y + 1
01,708              'CAUTION [
01,709              ' Next three line beep NOT
```

```
01,710            'DEFLNG A-Z
01,711            'aaa = 61
01,712            'bbb = -48
01,713            'IF aaa <= bbb THEN BEEP: END
01,714            'Next three line beep
01,715            'aaa~& = 61
01,716            'bbb~& = -48
01,717            'IF aaa~& <= bbb~& THEN BEEP: END
01,718            'Next three line beep NOT
01,719            'aaa& = 61
01,720            'bbb& = -48
01,721            'IF aaa& <= bbb& THEN BEEP: END
01,722            'Next line works even when unsigned!
01,723            'IF File_Frame_y + YdimROW - (filecount - (YdimROW - 1)) <= 0 THEN
01,724            'CAUTION ]
01,725            'Next line doesn't work when unsigned!
01,726            If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
01,727                File_Frame_y = File_Frame_y + YdimROW
01,728                'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,729                'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,730                If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
01,731                UpdateWindowFrame NormalFRGr, NormalBCKGr
01,732            Else
01,733                If PageScrollBenchmark = 1 Then
01,734                    TimeScrollB = Timer
01,735                    _KeyClear
01,736                    ReportTimeToScroll MostRightField
01,737                    PageScrollBenchmark = 0
01,738                    _Display
01,739                End If
01,740            End If
01,741        End If
01,742    End If
01,743    If cryOLD <> cry Then
01,744        UpdateCLine cryOLD, 1, NormalFRGr, NormalBCKGr, cryOLD
01,745        cryOLD = cry
01,746    End If
01,747    If LCTRL_HOME Then
01,748        Locate 1, 1, 1, 30, 31
01,749        crx = Pos(0)
01,750        cry = CsrLin
01,751        crxOLD = crx
01,752        cryOLD = cry
01,753        File_Frame_x = 1
01,754        File_Frame_y = 1
01,755        CurrentLine = 1
01,756        UpdateWindowFrame NormalFRGr, NormalBCKGr
01,757        'DO WHILE _MOUSEINPUT: LOOP
01,758    End If
01,759    If LCTRL_END Then
01,760        If filecount >= YdimROW Then
01,761            If filecount - (YdimROW - 1) Then
01,762                Locate YdimROW, 1, 1, 30, 31
01,763                crx = Pos(0)
01,764                cry = CsrLin
01,765                crxOLD = crx
01,766                cryOLD = cry
01,767                File_Frame_x = 1
01,768                File_Frame_y = filecount - (YdimROW - 1)
01,769                CurrentLine = filecount
01,770                UpdateWindowFrame NormalFRGr, NormalBCKGr
01,771            End If
01,772        Else
01,773            Locate filecount, 1, 1, 30, 31
01,774            crx = Pos(0)
01,775            cry = CsrLin
01,776            crxOLD = crx
01,777            cryOLD = cry
01,778            File_Frame_x = 1
01,779            File_Frame_y = 1
01,780            CurrentLine = filecount
01,781            UpdateWindowFrame NormalFRGr, NormalBCKGr
01,782        End If
01,783        'DO WHILE _MOUSEINPUT: LOOP
01,784    End If
01,785
01,786    If buttondown1 Then
01,787        tCurrentT1# = Timer(.001)
01,788        If tCurrentT1# - tLASTpress1# < .33 And tCurrentT1# - tLASTrelease1# < .33 Then Click1 = 2 Else Click1 = 0
01,789        If Click1 = 2 And SaveMouseY1 = _MouseY Then ' "Double click" same as SPACE key
01,790            a$ = RTrim$(FileArray$(CurrentLine))
01,791            If _FileExists(a$) Then
01,792                Close #1
01,793                _MemFree MhandleOFF
01,794                _MemFree MhandleLEN
01,795                If ToLoadOrNotFlag Then
01,796                    _MemFree Mwholefile
01,797                End If
01,798                GoTo GettingStarted 'Before going above lines must be executed, i.e. to initialize.
01,799            Else ' not responsive enough - it misses some double-clicks?!
01,800                If File_Frame_y - YdimROW >= 1 Then
01,801                    File_Frame_y = File_Frame_y - YdimROW
01,802                    If CurrentLine - YdimROW >= 1 Then CurrentLine = CurrentLine - YdimROW
01,803                    UpdateWindowFrame NormalFRGr, NormalBCKGr
01,804                End If
01,805            End If
01,806            SaveMouseY1 = 0 'execute once
01,807        End If
01,808        tLASTpress1# = Timer(.001)
01,809    Else
```

```
01,810          If Timer - tLASTpress1# < .01 Then tLASTrelease1# = Timer(.001): SaveMouseY1 = _MouseY
01,811      End If
01,812      If buttondown2 Then
01,813          tCurrentT2# = Timer(.001)
01,814          If tCurrentT2# - tLASTpress2# < .33 And tCurrentT2# - tLASTrelease2# < .33 Then Click2 = 2 Else Click2 = 0
01,815          If Click2 = 2 And SaveMouseY2 = _MouseY Then ' "Double click" same as SPACE key
01,816              If File_Frame_y + YdimROW <= filecount - (YdimROW - 1) Then
01,817                  File_Frame_y = File_Frame_y + YdimROW
01,818                  'IF CurrentLine < filecount THEN CurrentLine = CurrentLine + 1
01,819                  'IF CurrentLine + 1 <= filecount THEN CurrentLine = CurrentLine + 1
01,820                  If CurrentLine + YdimROW <= filecount Then CurrentLine = CurrentLine + YdimROW
01,821                  UpdateWindowFrame NormalFRGr, NormalBCKGr
01,822              End If
01,823              SaveMouseY2 = 0 'execute once
01,824          End If
01,825          tLASTpress2# = Timer(.001)
01,826      Else
01,827          If Timer - tLASTpress2# < .01 Then tLASTrelease2# = Timer(.001): SaveMouseY2 = _MouseY
01,828      End If
01,829
01,830      Locate cry, crx, 1, 30, 31
01,831      UpdateCLine cry, 1, InverseFRGr, InverseBCKGr, cry
01,832      UpdateCLL CurrentLine, Len(Dumbo$) + 1 + 1
01,833      _Display
01,834      'DO WHILE INKEY$ <> "": LOOP ' have to clear the keyboard buffer
01,835      _Limit 500 '_LIMIT 30 'commented because the wheel up/down was not working?!
01,836      '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".
01,837 Loop
01,838
01,839 End
01,840
01,841 Sub ReturnCOMBO
01,842      Shared LSHIFT_RSHIFT
01,843      Shared LCTRL_RCTRL
01,844      Shared LALT_RALT
01,845
01,846      Shared LALT_HOME
01,847      Shared LALT_END
01,848
01,849      Shared LALT_INS
01,850      Shared LALT_DEL
01,851
01,852      Shared LALT_PGUP
01,853      Shared LALT_PGDN
01,854
01,855      Shared LALT_Left
01,856      Shared LALT_Right
01,857      Shared LALT_Up
01,858      Shared LALT_Down
01,859
01,860      Shared RALT_HOME
01,861      Shared RALT_END
01,862
01,863      Shared RALT_INS
01,864      Shared RALT_DEL
01,865
01,866      Shared RALT_PGUP
01,867      Shared RALT_PGDN
01,868
01,869      Shared RALT_Left
01,870      Shared RALT_Right
01,871      Shared RALT_Up
01,872      Shared RALT_Down
01,873
01,874      Shared LSHIFT_HOME
01,875      Shared LSHIFT_END
01,876
01,877      Shared LSHIFT_INS
01,878      Shared LSHIFT_DEL
01,879
01,880      Shared LSHIFT_PGUP
01,881      Shared LSHIFT_PGDN
01,882
01,883      Shared LSHIFT_Left
01,884      Shared LSHIFT_Right
01,885      Shared LSHIFT_Up
01,886      Shared LSHIFT_Down
01,887
01,888      Shared RSHIFT_HOME
01,889      Shared RSHIFT_END
01,890
01,891      Shared RSHIFT_INS
01,892      Shared RSHIFT_DEL
01,893
01,894      Shared RSHIFT_PGUP
01,895      Shared RSHIFT_PGDN
01,896
01,897      Shared RSHIFT_Left
01,898      Shared RSHIFT_Right
01,899      Shared RSHIFT_Up
01,900      Shared RSHIFT_Down
01,901
01,902      Shared LCTRL_HOME
01,903      Shared LCTRL_END
01,904
01,905      Shared LCTRL_INS
01,906      Shared LCTRL_DEL
01,907
01,908      Shared LCTRL_PGUP
```

```
01,909      Shared LCTRL_PGDN
01,910
01,911      Shared LCTRL_Left
01,912      Shared LCTRL_Right
01,913      Shared LCTRL_Up
01,914      Shared LCTRL_Down
01,915
01,916      Shared RCTRL_HOME
01,917      Shared RCTRL_END
01,918
01,919      Shared RCTRL_INS
01,920      Shared RCTRL_DEL
01,921
01,922      Shared RCTRL_PGUP
01,923      Shared RCTRL_PGDN
01,924
01,925      Shared RCTRL_Left
01,926      Shared RCTRL_Right
01,927      Shared RCTRL_Up
01,928      Shared RCTRL_Down
01,929
01,930      Shared LSHIFT_LCTRL_HOME
01,931      Shared LSHIFT_LCTRL_END
01,932
01,933      Shared LSHIFT_LCTRL_INS
01,934      Shared LSHIFT_LCTRL_DEL
01,935
01,936      Shared LSHIFT_LCTRL_PGUP
01,937      Shared LSHIFT_LCTRL_PGDN
01,938
01,939      Shared LSHIFT_LCTRL_Left
01,940      Shared LSHIFT_LCTRL_Right
01,941      Shared LSHIFT_LCTRL_Up
01,942      Shared LSHIFT_LCTRL_Down
01,943
01,944      Shared LSHIFT_BackSpace
01,945      Shared LSHIFT_TAB
01,946      Shared LSHIFT_SPACE
01,947      Shared LSHIFT_ESC
01,948      Shared LSHIFT_ENTER
01,949
01,950      Shared RSHIFT_BackSpace
01,951      Shared RSHIFT_TAB
01,952      Shared RSHIFT_SPACE
01,953      Shared RSHIFT_ESC
01,954      Shared RSHIFT_ENTER
01,955
01,956      Shared LCTRL_SPACE
01,957      Shared LCTRL_ENTER
01,958
01,959      Shared RCTRL_SPACE
01,960      Shared RCTRL_ENTER
01,961
01,962      If _KeyDown(LSHIFTkey&) And _KeyDown(RSHIFTkey&) Then LSHIFT_RSHIFT = 1 Else LSHIFT_RSHIFT = 0
01,963      If _KeyDown(LCTRLkey&) And _KeyDown(RCTRLkey&) Then LCTRL_RCTRL = 1 Else LCTRL_RCTRL = 0
01,964      If _KeyDown(LALTkey&) And _KeyDown(RALTkey&) Then LALT_RALT = 1 Else LALT_RALT = 0
01,965
01,966      'LALT:
01,967      If _KeyDown(LALTkey&) And _KeyDown(HOMEkey&) Then LALT_HOME = 1 Else LALT_HOME = 0
01,968      If _KeyDown(LALTkey&) And _KeyDown(ENDkey&) Then LALT_END = 1 Else LALT_END = 0
01,969
01,970      If _KeyDown(LALTkey&) And _KeyDown(INSkey&) Then LALT_INS = 1 Else LALT_INS = 0
01,971      If _KeyDown(LALTkey&) And _KeyDown(DELkey&) Then LALT_DEL = 1 Else LALT_DEL = 0
01,972
01,973      If _KeyDown(LALTkey&) And _KeyDown(PGUPkey&) Then LALT_PGUP = 1 Else LALT_PGUP = 0
01,974      If _KeyDown(LALTkey&) And _KeyDown(PGDNkey&) Then LALT_PGDN = 1 Else LALT_PGDN = 0
01,975
01,976      If _KeyDown(LALTkey&) And _KeyDown(LEFTkey&) Then LALT_Left = 1 Else LALT_Left = 0
01,977      If _KeyDown(LALTkey&) And _KeyDown(RIGHTkey&) Then LALT_Right = 1 Else LALT_Right = 0
01,978      If _KeyDown(LALTkey&) And _KeyDown(UPkey&) Then LALT_Up = 1 Else LALT_Up = 0
01,979      If _KeyDown(LALTkey&) And _KeyDown(DOWNkey&) Then LALT_Down = 1 Else LALT_Down = 0
01,980
01,981      'RALT:
01,982      If _KeyDown(RALTkey&) And _KeyDown(HOMEkey&) Then RALT_HOME = 1 Else RALT_HOME = 0
01,983      If _KeyDown(RALTkey&) And _KeyDown(ENDkey&) Then RALT_END = 1 Else RALT_END = 0
01,984
01,985      If _KeyDown(RALTkey&) And _KeyDown(INSkey&) Then RALT_INS = 1 Else RALT_INS = 0
01,986      If _KeyDown(RALTkey&) And _KeyDown(DELkey&) Then RALT_DEL = 1 Else RALT_DEL = 0
01,987
01,988      If _KeyDown(RALTkey&) And _KeyDown(PGUPkey&) Then RALT_PGUP = 1 Else RALT_PGUP = 0
01,989      If _KeyDown(RALTkey&) And _KeyDown(PGDNkey&) Then RALT_PGDN = 1 Else RALT_PGDN = 0
01,990
01,991      If _KeyDown(RALTkey&) And _KeyDown(LEFTkey&) Then Print RALT_Left = 1 Else RALT_Left = 0
01,992      If _KeyDown(RALTkey&) And _KeyDown(RIGHTkey&) Then RALT_Right = 1 Else RALT_Right = 0
01,993      If _KeyDown(RALTkey&) And _KeyDown(UPkey&) Then RALT_Up = 1 Else RALT_Up = 0
01,994      If _KeyDown(RALTkey&) And _KeyDown(DOWNkey&) Then RALT_Down = 1 Else RALT_Down = 0
01,995
01,996      'LSHIFT:
01,997      If _KeyDown(LSHIFTkey&) And _KeyDown(HOMEkey&) Then LSHIFT_HOME = 1 Else LSHIFT_HOME = 0
01,998      If _KeyDown(LSHIFTkey&) And _KeyDown(ENDkey&) Then LSHIFT_END = 1 Else LSHIFT_END = 0
01,999
02,000      If _KeyDown(LSHIFTkey&) And _KeyDown(INSkey&) Then LSHIFT_INS = 1 Else LSHIFT_INS = 0
02,001      If _KeyDown(LSHIFTkey&) And _KeyDown(DELkey&) Then LSHIFT_DEL = 1 Else LSHIFT_DEL = 0
02,002
02,003      If _KeyDown(LSHIFTkey&) And _KeyDown(PGUPkey&) Then LSHIFT_PGUP = 1 Else LSHIFT_PGUP = 0
02,004      If _KeyDown(LSHIFTkey&) And _KeyDown(PGDNkey&) Then LSHIFT_PGDN = 1 Else LSHIFT_PGDN = 0
02,005
02,006      If _KeyDown(LSHIFTkey&) And _KeyDown(LEFTkey&) Then LSHIFT_Left = 1 Else LSHIFT_Left = 0
02,007      If _KeyDown(LSHIFTkey&) And _KeyDown(RIGHTkey&) Then LSHIFT_Right = 1 Else LSHIFT_Right = 0
02,008      If _KeyDown(LSHIFTkey&) And _KeyDown(UPkey&) Then LSHIFT_Up = 1 Else LSHIFT_Up = 0
```

```
02,009     If _KeyDown(LSHIFTkey&) And _KeyDown(DOWNkey&) Then LSHIFT_Down = 1 Else LSHIFT_Down = 0
02,010
02,011     'RSHIFT:
02,012     If _KeyDown(RSHIFTkey&) And _KeyDown(HOMEkey&) Then RSHIFT_HOME = 1 Else RSHIFT_HOME = 0
02,013     If _KeyDown(RSHIFTkey&) And _KeyDown(ENDkey&) Then RSHIFT_END = 1 Else RSHIFT_END = 0
02,014
02,015     If _KeyDown(RSHIFTkey&) And _KeyDown(INSkey&) Then RSHIFT_INS = 1 Else RSHIFT_INS = 0
02,016     If _KeyDown(RSHIFTkey&) And _KeyDown(DELkey&) Then RSHIFT_DEL = 1 Else RSHIFT_DEL = 0
02,017
02,018     If _KeyDown(RSHIFTkey&) And _KeyDown(PGUPkey&) Then RSHIFT_PGUP = 1 Else RSHIFT_PGUP = 0
02,019     If _KeyDown(RSHIFTkey&) And _KeyDown(PGDNkey&) Then RSHIFT_PGDN = 1 Else RSHIFT_PGDN = 0
02,020
02,021     If _KeyDown(RSHIFTkey&) And _KeyDown(LEFTkey&) Then RSHIFT_Left = 1 Else RSHIFT_Left = 0
02,022     If _KeyDown(RSHIFTkey&) And _KeyDown(RIGHTkey&) Then RSHIFT_Right = 1 Else RSHIFT_Right = 0
02,023     If _KeyDown(RSHIFTkey&) And _KeyDown(UPkey&) Then RSHIFT_Up = 1 Else RSHIFT_Up = 0
02,024     If _KeyDown(RSHIFTkey&) And _KeyDown(DOWNkey&) Then RSHIFT_Down = 1 Else RSHIFT_Down = 0
02,025
02,026     'LCTRL:
02,027     If _KeyDown(LCTRLkey&) And _KeyDown(HOMEkey&) Then LCTRL_HOME = 1 Else LCTRL_HOME = 0
02,028     If _KeyDown(LCTRLkey&) And _KeyDown(ENDkey&) Then LCTRL_END = 1 Else LCTRL_END = 0
02,029
02,030     If _KeyDown(LCTRLkey&) And _KeyDown(INSkey&) Then LCTRL_INS = 1 Else LCTRL_INS = 0
02,031     If _KeyDown(LCTRLkey&) And _KeyDown(DELkey&) Then LCTRL_DEL = 1 Else LCTRL_DEL = 0
02,032
02,033     If _KeyDown(LCTRLkey&) And _KeyDown(PGUPkey&) Then LCTRL_PGUP = 1 Else LCTRL_PGUP = 0
02,034     If _KeyDown(LCTRLkey&) And _KeyDown(PGDNkey&) Then LCTRL_PGDN = 1 Else LCTRL_PGDN = 0
02,035
02,036     If _KeyDown(LCTRLkey&) And _KeyDown(LEFTkey&) Then LCTRL_Left = 1 Else LCTRL_Left = 0
02,037     If _KeyDown(LCTRLkey&) And _KeyDown(RIGHTkey&) Then LCTRL_Right = 1 Else LCTRL_Right = 0
02,038     If _KeyDown(LCTRLkey&) And _KeyDown(UPkey&) Then LCTRL_Up = 1 Else LCTRL_Up = 0
02,039     If _KeyDown(LCTRLkey&) And _KeyDown(DOWNkey&) Then LCTRL_Down = 1 Else LCTRL_Down = 0
02,040
02,041     'RCTRL:
02,042     If _KeyDown(RCTRLkey&) And _KeyDown(HOMEkey&) Then RCTRL_HOME = 1 Else RCTRL_HOME = 0
02,043     If _KeyDown(RCTRLkey&) And _KeyDown(ENDkey&) Then RCTRL_END = 1 Else RCTRL_END = 0
02,044
02,045     If _KeyDown(RCTRLkey&) And _KeyDown(INSkey&) Then RCTRL_INS = 1 Else RCTRL_INS = 0
02,046     If _KeyDown(RCTRLkey&) And _KeyDown(DELkey&) Then RCTRL_DEL = 1 Else RCTRL_DEL = 0
02,047
02,048     If _KeyDown(RCTRLkey&) And _KeyDown(PGUPkey&) Then RCTRL_PGUP = 1 Else RCTRL_PGUP = 0
02,049     If _KeyDown(RCTRLkey&) And _KeyDown(PGDNkey&) Then RCTRL_PGDN = 1 Else RCTRL_PGDN = 0
02,050
02,051     If _KeyDown(RCTRLkey&) And _KeyDown(LEFTkey&) Then RCTRL_Left = 1 Else RCTRL_Left = 0
02,052     If _KeyDown(RCTRLkey&) And _KeyDown(RIGHTkey&) Then RCTRL_Right = 1 Else RCTRL_Right = 0
02,053     If _KeyDown(RCTRLkey&) And _KeyDown(UPkey&) Then RCTRL_Up = 1 Else RCTRL_Up = 0
02,054     If _KeyDown(RCTRLkey&) And _KeyDown(DOWNkey&) Then RCTRL_Down = 1 Else RCTRL_Down = 0
02,055
02,056     'LSHIFT+LCTRL: NOTICE: LSHIFT+LCTRL+Left triggers 3 variables on - 1] LSHIFT_LCTRL_Left 2] LSHIFT_Left 3] LCTRL_Left
02,057     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(HOMEkey&) Then LSHIFT_LCTRL_HOME = 1 Else LSHIFT_LCTRL_HOME = 0
02,058     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(ENDkey&) Then LSHIFT_LCTRL_END = 1 Else LSHIFT_LCTRL_END = 0
02,059
02,060     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(INSkey&) Then LSHIFT_LCTRL_INS = 1 Else LSHIFT_LCTRL_INS = 0
02,061     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(DELkey&) Then LSHIFT_LCTRL_DEL = 1 Else LSHIFT_LCTRL_DEL = 0
02,062
02,063     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(PGUPkey&) Then LSHIFT_LCTRL_PGUP = 1 Else LSHIFT_LCTRL_PGUP = 0
02,064     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(PGDNkey&) Then LSHIFT_LCTRL_PGDN = 1 Else LSHIFT_LCTRL_PGDN = 0
02,065
02,066     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(LEFTkey&) Then LSHIFT_LCTRL_Left = 1 Else LSHIFT_LCTRL_Left = 0
02,067     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(RIGHTkey&) Then LSHIFT_LCTRL_Right = 1 Else LSHIFT_LCTRL_Right = 0
02,068     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(UPkey&) Then LSHIFT_LCTRL_Up = 1 Else LSHIFT_LCTRL_Up = 0
02,069     If _KeyDown(LSHIFTkey&) And _KeyDown(LCTRLkey&) And _KeyDown(DOWNkey&) Then LSHIFT_LCTRL_Down = 1 Else LSHIFT_LCTRL_Down = 0
02,070
02,071     'LSHIFT:
02,072     If _KeyDown(LSHIFTkey&) And _KeyDown(BACKSPCkey&) Then LSHIFT_BackSpace = 1 Else LSHIFT_BackSpace = 0
02,073     If _KeyDown(LSHIFTkey&) And _KeyDown(TABkey&) Then LSHIFT_TAB = 1 Else LSHIFT_TAB = 0
02,074     If _KeyDown(LSHIFTkey&) And _KeyDown(SPACEkey&) Then LSHIFT_SPACE = 1 Else LSHIFT_SPACE = 0
02,075     If _KeyDown(LSHIFTkey&) And _KeyDown(ESCkey&) Then LSHIFT_ESC = 1 Else LSHIFT_ESC = 0
02,076     If _KeyDown(LSHIFTkey&) And _KeyDown(ENTERkey&) Then LSHIFT_ENTER = 1 Else LSHIFT_ENTER = 0
02,077     'RSHIFT:
02,078     If _KeyDown(RSHIFTkey&) And _KeyDown(BACKSPCkey&) Then RSHIFT_BackSpace = 1 Else RSHIFT_BackSpace = 0
02,079     If _KeyDown(RSHIFTkey&) And _KeyDown(TABkey&) Then RSHIFT_TAB = 1 Else RSHIFT_TAB = 0
02,080     If _KeyDown(RSHIFTkey&) And _KeyDown(SPACEkey&) Then RSHIFT_SPACE = 1 Else RSHIFT_SPACE = 0
02,081     If _KeyDown(RSHIFTkey&) And _KeyDown(ESCkey&) Then RSHIFT_ESC = 1 Else RSHIFT_ESC = 0
02,082     If _KeyDown(RSHIFTkey&) And _KeyDown(ENTERkey&) Then RSHIFT_ENTER = 1 Else RSHIFT_ENTER = 0
02,083
02,084     'LCTRL:
02,085     If _KeyDown(LCTRLkey&) And _KeyDown(SPACEkey&) Then LCTRL_SPACE = 1 Else LCTRL_SPACE = 0
02,086     If _KeyDown(LCTRLkey&) And _KeyDown(ENTERkey&) Then LCTRL_ENTER = 1 Else LCTRL_ENTER = 0
02,087     'RCTRL:
02,088     If _KeyDown(RCTRLkey&) And _KeyDown(SPACEkey&) Then RCTRL_SPACE = 1 Else RCTRL_SPACE = 0
02,089     If _KeyDown(RCTRLkey&) And _KeyDown(ENTERkey&) Then RCTRL_ENTER = 1 Else RCTRL_ENTER = 0
02,090
02,091     '_KEYCLEAR
02,092 End Sub
02,093
02,094 Sub ExpandTabs (l$)
02,095     Shared XdimCOL
02,096     If InStr(l$, Chr$(9)) Then
02,097         TabV% = 8
02,098         b$ = "": f& = 0
02,099         PossibleRecalculation = Len(l$)
02,100         For i& = 1 To MIN&(PossibleRecalculation, XdimCOL) 'LEN(l$) ' For some reason going all the length is too slow...?! As it is, the lateral scroll is crippled!
02,101             RemoveSpecial$ = Mid$(l$, i&, 1)
02,102             If RemoveSpecial$ = Chr$(9) Then
02,103                 b$ = b$ + String$((f& \ TabV%) * TabV% + TabV% - f&, " ")
02,104                 '  |
02,105                 '  |
02,106                 ' \|/
02,107                 'TabV% - (f% - (f% \ TabV%) * TabV%) =
02,108                 'TabV% - (f% MOD TabV%)
```

```
02,109                f& = (f& \ TabV%) * TabV% + TabV%
02,110            Else
02,111                b$ = b$ + RemoveSpecial$ 'MID$(l$, i&, 1)
02,112                f& = f& + 1
02,113            End If
02,114        Next
02,115        l$ = b$
02,116    End If
02,117 End Sub
02,118
02,119 Function MIN& (YdimROW, filecount)
02,120    If YdimROW < filecount Then MIN& = YdimROW Else MIN& = filecount
02,121 End Function
02,122
02,123 Function AddCommas$ (numeral)
02,124    s$ = LTrim$(Str$(numeral))
02,125    If Len(s$) > 3 Then
02,126        If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
02,127        s$ = ""
02,128        For i = 1 To Len(x$) Step 3
02,129            s$ = s$ + Mid$(x$, i, 3) + ","
02,130        Next
02,131        s$ = Left$(s$, Len(s$) - 1)
02,132    End If
02,133    AddCommas$ = LTrim$(s$)
02,134 End Function
02,135
02,136 Function AddCommasPaddedNUM$ (numeral)
02,137    Shared FieldLineNum
02,138    s$ = LTrim$(Str$(numeral))
02,139    If Len(s$) > 3 Then
02,140        If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
02,141        s$ = ""
02,142        For i = 1 To Len(x$) Step 3
02,143            s$ = s$ + Mid$(x$, i, 3) + ","
02,144        Next
02,145        s$ = Left$(s$, Len(s$) - 1)
02,146    End If
02,147    s$ = LTrim$(s$)
02,148    Padded$ = Right$("000,000,000,000", FieldLineNum)
02,149    Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
02,150    AddCommasPaddedNUM$ = Padded$
02,151 End Function
02,152
02,153 Function AddCommasPaddedLEN$ (numeral)
02,154    Shared FieldLineLen
02,155    s$ = LTrim$(Str$(numeral))
02,156    If Len(s$) > 3 Then
02,157        If (Len(s$) Mod 3) Then x$ = String$(3 - (Len(s$) Mod 3), " ") + s$ Else x$ = s$
02,158        s$ = ""
02,159        For i = 1 To Len(x$) Step 3
02,160            s$ = s$ + Mid$(x$, i, 3) + ","
02,161        Next
02,162        s$ = Left$(s$, Len(s$) - 1)
02,163    End If
02,164    s$ = LTrim$(s$)
02,165    Padded$ = Right$("000,000,000,000", FieldLineLen)
02,166    Mid$(Padded$, Len(Padded$) - Len(s$) + 1, Len(s$)) = s$
02,167    AddCommasPaddedLEN$ = Padded$
02,168 End Function
02,169
02,170 Sub UpdateCLL (l, posit)
02,171    Shared YdimROW ', FileArray$()
02,172    crx = Pos(0)
02,173    cry = CsrLin
02,174    Locate YdimROW + 1, posit: Color 9, 0
02,175    Print "; Line Number: " + AddCommasPaddedNUM$(l);
02,176    Print "; Line Length: " + AddCommasPaddedLEN$(Len(FileArray$(l)));
02,177    Locate cry, crx, 1, 30, 31
02,178 End Sub
02,179
02,180 Sub UpdateNextToCCL_BUSY (posit)
02,181    Shared YdimROW
02,182    crx = Pos(0)
02,183    cry = CsrLin
02,184    Locate YdimROW + 1, posit: Color 9, 0: Print "; Status: BUSY";
02,185    Locate cry, crx, 1, 30, 31
02,186 End Sub
02,187
02,188 Sub UpdateNextToCCL_DONE (posit)
02,189    Shared YdimROW
02,190    crx = Pos(0)
02,191    cry = CsrLin
02,192    Locate YdimROW + 1, posit: Color 9, 0: Print "; Status: DONE";
02,193    Locate cry, crx, 1, 30, 31
02,194 End Sub
02,195
02,196 Sub UpdateNextToCCL_UNWRAPPABLEcount (posit)
02,197    Shared YdimROW
02,198    Shared UnwrappableLines
02,199    crx = Pos(0)
02,200    cry = CsrLin
02,201    Locate YdimROW + 1, posit: Color 9, 0: Print "; Unwrappable Lines: " + AddCommas$(UnwrappableLines);
02,202    Locate cry, crx, 1, 30, 31
02,203 End Sub
02,204
02,205 Sub UpdateCLine (lineToWrite, columnToWrite, FRGR, BACKGR, ln)
02,206    Shared FileArrayWINDOW$()
02,207    Shared XdimCOL
02,208    Shared CurrentWord$
```

```
02,209     crx = Pos(0)
02,210     cry = CsrLin
02,211     Color FRGR, BACKGR
02,212     Locate lineToWrite, columnToWrite, 1, 30, 31
02,213     Print FileArrayWINDOW$(ln);
02,214     'get the word under the cursor [
02,215     LeftM = crx: RightM = crx 'both can go within 1..XdimCOL
02,216     CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM, 1)
02,217     If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
02,218         Do While LeftM > 1
02,219             CurrentChar$ = Mid$(FileArrayWINDOW$(ln), LeftM - 1, 1)
02,220             If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
02,221                 LeftM = LeftM - 1
02,222             Else
02,223                 Exit Do
02,224             End If
02,225         Loop
02,226         Do While RightM < XdimCOL
02,227             CurrentChar$ = Mid$(FileArrayWINDOW$(ln), RightM + 1, 1)
02,228             If (CurrentChar$ >= "a" And CurrentChar$ <= "z") Or (CurrentChar$ >= "A" And CurrentChar$ <= "Z") Then
02,229                 RightM = RightM + 1
02,230             Else
02,231                 Exit Do
02,232             End If
02,233         Loop
02,234     End If
02,235     If RightM - LeftM > 0 Then CurrentWord$ = Mid$(FileArrayWINDOW$(ln), LeftM, RightM - LeftM + 1) Else CurrentWord$ = ""
02,236     'get the word under the cursor ]
02,237     Locate cry, crx, 1, 30, 31
02,238 End Sub
02,239
02,240 Sub UpdateWindowFrame (FRGR, BACKGR)
02,241     Shared YdimROW, filecount, File_Frame_y, File_Frame_x, XdimCOL, FileArrayWINDOW$() ', FileArray$()
02,242     Color FRGR, BACKGR
02,243     If YdimROW > filecount Then
02,244         For i = 1 To YdimROW
02,245             Locate i, 1: Print Space$(XdimCOL);
02,246         Next
02,247     End If
02,248     For i = 1 To MIN&(YdimROW, filecount)
02,249         DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
02,250         If Len(DumboReadOnceNotThrice$) >= XdimCOL Then
02,251             FileArrayWINDOW$(i) = Mid$(DumboReadOnceNotThrice$, File_Frame_x, XdimCOL)
02,252         Else
02,253             FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + Space$(XdimCOL - Len(DumboReadOnceNotThrice$))
02,254         End If
02,255         'since r.8 the goal is to browse properly binary/.tar files
02,256         For j = 1 To Len(FileArrayWINDOW$(i))
02,257             If Mid$(FileArrayWINDOW$(i), j, 1) < " " Then Mid$(FileArrayWINDOW$(i), j, 1) = Chr$(32)
02,258         Next
02,259         Locate i, 1: Print FileArrayWINDOW$(i);
02,260     Next
02,261 End Sub
02,262
02,263 Sub ReportTimeToLoad (posit)
02,264     Shared YdimROW, XdimCOL
02,265     Shared TimeA, TimeB
02,266     crx = Pos(0)
02,267     cry = CsrLin
02,268     ElapsedTime% = TimeB - TimeA
02,269     If ElapsedTime% < 0 Then ElapsedTime% = (86400 - TimeA) + TimeB
02,270     Paddedstr$ = "; Loaded in " + LTrim$(Str$(ElapsedTime%)) + " seconds."
02,271     '     p
02,272     '[123456] (8-5)-1=2
02,273     Locate YdimROW + 1, posit: Color 4, 0: Print Space$((XdimCOL - posit) - 1);
02,274     Locate YdimROW + 1, posit: Color 4, 0: Print Paddedstr$;
02,275     Locate cry, crx, 1, 30, 31
02,276 End Sub
02,277
02,278 Sub ReportTimeToScroll (posit)
02,279     Shared YdimROW, XdimCOL
02,280     Shared TimeScrollA, TimeScrollB
02,281     Shared LineScrollBenchmark
02,282     Shared PageScrollBenchmark
02,283     Shared filecount
02,284     crx = Pos(0)
02,285     cry = CsrLin
02,286     ElapsedTime% = TimeScrollB - TimeScrollA
02,287     If ElapsedTime% < 0 Then ElapsedTime% = (86400 - TimeScrollA) + TimeScrollB
02,288     If ElapsedTime% = 0 Then ElapsedTime% = 1
02,289     'Line-By-Line or Line granularity Scroll Rate; LPS stands for Lines-Per-Second
02,290     'Page-By-Page or Page granularity Scroll Rate; PPS stands for Pages-Per-Second
02,291     Locate YdimROW + 1, posit: Color 4, 0: Print Space$((XdimCOL - posit) - 1);
02,292     If XdimCOL = 198 Then
02,293         If LineScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; DOWNed in"; ElapsedTime%; "seconds, or "; AddCommasPaddedLEN$(filecount / ElapsedTime%); " LPS";
02,294         If PageScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; PGDNed in"; ElapsedTime%; "seconds, or "; AddCommasPaddedLEN$((filecount / YdimROW) /
ElapsedTime%); " PPS";
02,295     Else
02,296         If LineScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; DOWNed in"; ElapsedTime%; "seconds.";
02,297         If PageScrollBenchmark = 1 Then Locate YdimROW + 1, posit: Color 4, 0: Print "; PGDNed in"; ElapsedTime%; "seconds.";
02,298     End If
02,299     Locate cry, crx, 1, 30, 31
02,300 End Sub
02,301
02,302 Function FileArray$ (i)
02,303     Shared QWORD, LineLen13
02,304     'SHARED TheWholeFile$
02,305     _MemGet MhandleOFF, MhandleOFF.OFFSET + 8&& * (i - 1), QWORD
02,306     _MemGet MhandleLEN, MhandleLEN.OFFSET + 8&& * (i - 1), LineLen13
02,307     If ToLoadOrNotFlag Then
```

```
02,308         'BufferForLine$ = MID$(TheWholeFile$, QWORD, LineLen13)
02,309         BufferForLine$ = Space$(LineLen13)
02,310         _MemGet Mwholefile, Mwholefile.OFFSET + (QWORD - 1), BufferForLine$
02,311     Else
02,312         BufferForLine$ = Space$(LineLen13)
02,313         Seek #1, QWORD
02,314         Get #1, , BufferForLine$
02,315     End If
02,316     ExpandTabs (BufferForLine$)
02,317     FileArray$ = BufferForLine$
02,318 End Function
02,319
02,320 Sub ShowIndigoWindow
02,321     'COLOR 8, 0: LOCATE i, 1 + LEN(FileArrayWINDOW$(i)): PRINT CHR$(179);: COLOR FRGR, BACKGR
02,322 End Sub
02,323
02,324 'http://rosettacode.org/wiki/Levenshtein_distance#FutureBasic
02,325 'FutureBasic
02,326 'Based on Wikipedia algorithm. Suitable for Pascal strings.
02,327
02,328 'include "ConsoleWindow"
02,329
02,330 'local fn LevenshteinDistance( aStr as Str255, bStr as Str255 ) as long
02,331 'dim as long m, n, i, j, min, k, l
02,332 'dim as long distance( 255, 255 )
02,333
02,334 'm = len(aStr)
02,335 'n = len(bStr)
02,336
02,337 'for i = 0 to m
02,338 '   distance( i, 0 ) = i
02,339 'next
02,340
02,341 'for j = 0 to n
02,342 '   distance( 0, j ) = j
02,343 'next
02,344
02,345 'for j = 1 to n
02,346 '   for i = 1 to m
02,347 '      if mid$( aStr, i, 1 ) == mid$( bStr, j, 1 )
02,348 '         distance( i, j ) = distance( i-1, j-1 )
02,349 '      else
02,350 '         min = distance( i-1, j   ) + 1
02,351 '         k   = distance( i, j - 1 ) + 1
02,352 '         l   = distance( i-1, j-1 ) + 1
02,353 '         if k < min then min = k
02,354 '         if l < min then min = l
02,355 '         distance( i, j ) = min
02,356 '      end if
02,357 '   next
02,358 'next
02,359 'end fn = distance( m, n )
02,360
02,361 'dim as long i
02,362 'dim as Str255 testStr( 5, 2 )
02,363
02,364 'testStr( 0, 0 ) = "kitten"      : testStr( 0, 1 ) = "sitting"
02,365 'testStr( 1, 0 ) = "rosettacode" : testStr( 1, 1 ) = "raisethysword"
02,366 'testStr( 2, 0 ) = "Saturday"    : testStr( 2, 1 ) = "Sunday"
02,367 'testStr( 3, 0 ) = "FutureBasic" : testStr( 3, 1 ) = "FutureBasic"
02,368 'testStr( 4, 0 ) = "here's a bunch of words"
02,369 'testStr( 4, 1 ) = "to wring out this code"
02,370
02,371 'for i = 0 to 4
02,372 '   print "1st string = "; testStr( i, 0 )
02,373 '   print "2nd string = "; testStr( i, 1 )
02,374 '   print "Levenshtein distance ="; fn LevenshteinDistance( testStr( i, 0 ), testStr( i, 1 ) )
02,375 '   print
02,376 'next
02,377
02,378 'Output:
02,379
02,380 '1st string = kitten
02,381 '2nd string = sitting
02,382 'Levenshtein distance = 3
02,383
02,384 '1st string = rosettacode
02,385 '2nd string = raisethysword
02,386 'Levenshtein distance = 8
02,387
02,388 '1st string = Saturday
02,389 '2nd string = Sunday
02,390 'Levenshtein distance = 3
02,391
02,392 '1st string = FutureBasic
02,393 '2nd string = FutureBasic
02,394 'Levenshtein distance = 0
02,395
02,396 '1st string = here's a bunch of words
02,397 '2nd string = to wring out this code
02,398 'Levenshtein distance = 18
02,399
02,400 '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.
02,401
02,402 Sub DrawBoxShadow3 (TopLrow, TopLcol, BottomRrow, BottomRcol, Captnn$)
02,403     Shared FileSize
02,404     Shared LoadedFile$
02,405     Shared PSPlike$
```

```
Shared YdimROW, filecount, File_Frame_y, XdimCOL, FileArrayWINDOW$() ', FileArray$()
'Shadow
BckGRcolor = 4
Color 8, BACKGR
For i = 1 To MIN&(YdimROW, filecount)
    'DumboReadOnceNotThrice$ = FileArray$(i + (File_Frame_y - 1))
    'IF LEN(DumboReadOnceNotThrice$) >= XdimCOL THEN
    '    FileArrayWINDOW$(i) = MID$(DumboReadOnceNotThrice$, 1, XdimCOL)
    'ELSE
    '    FileArrayWINDOW$(i) = DumboReadOnceNotThrice$ + SPACE$(XdimCOL - LEN(DumboReadOnceNotThrice$))
    'END IF
    If i >= TopLrow + 1 And i <= BottomRrow + 1 Then
        Locate i, TopLcol + 1: Print Mid$(FileArrayWINDOW$(i), TopLcol + 1, BottomRcol - TopLcol + 1);
    End If
Next
'Outer frame
Locate TopLrow, TopLcol
Color 7, BckGRcolor
Print Chr$(218); String$(BottomRcol - TopLcol - 1, Chr$(196));
Color 0, BckGRcolor: Print Chr$(191);
Color 7, BckGRcolor: Locate TopLrow, TopLcol + 2: Print "[ ";: Color 15, BckGRcolor: Print Captnn$;: Color 7, BckGRcolor: Print " ]";
For i = TopLrow + 1 To BottomRrow - 1
    Color 7, BckGRcolor: Locate i, TopLcol: Print Chr$(179); String$(BottomRcol - TopLcol - 1, Chr$(32));: Color 0, BckGRcolor: Print Chr$(179);
Next
Locate BottomRrow, TopLcol
Color 7, BckGRcolor: Print Chr$(192);
Color 0, BckGRcolor: Print String$(BottomRcol - TopLcol - 1, Chr$(196)); Chr$(217)
'Inner frame
'In case of YdimROW = 40 then shrunk panel
If YdimROW = 40 Then shrunkP = 14 Else shrunkP = 0
Locate TopLrow + 38 - shrunkP, TopLcol + 1
Color 0, BckGRcolor
Print Chr$(218); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196));
Color 7, BckGRcolor: Print Chr$(191);
For i = TopLrow + 39 - shrunkP To TopLrow + 40 - shrunkP
    Color 0, BckGRcolor: Locate i, TopLcol + 1: Print Chr$(179); String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(32));: Color 7, BckGRcolor: Print Chr$(179);
Next
Locate TopLrow + 40 - shrunkP, TopLcol + 1
Color 0, BckGRcolor: Print Chr$(192);
Color 7, BckGRcolor: Print String$(BottomRcol - 1 - (TopLcol + 1) - 1, Chr$(196)); Chr$(217)

Locate TopLrow + 2, TopLcol + 2
'PRINT " _  __              _                          "
'PRINT "| |/ /__ _ ______ _| |__   __ _ _ __   __ _ "
'PRINT "| ' // _` |_  / _` | '_ \ / _` | '_ \ / _` |"
'PRINT "| . \ (_| |/ / (_| | | | | (_| | | | | (_| |"
'PRINT "|_|\_\__,_/___\__,_|_| |_|\__,_|_| |_|\__,_|"
Color 7, BckGRcolor
Print " ____  __.                   .__"
Locate , TopLcol + 2
Print "|    |/ _|____  _____________ |  |__ _____    ____ _____"
Locate , TopLcol + 2
Print "|      < \__  \ \___   /\__  \ |  |  \\__  \  /    \\__  \"
Locate , TopLcol + 2
Print "|    |  \ / __ \_/    /  / __ \|   Y  \/ __ \|   |  \/ __ \_"
Locate , TopLcol + 2
Print "|____|__ \(____  /_____ \(____  /___|  (____  /___|  (____  /"
Locate , TopLcol + 2
Print "        \/     \/      \/     \/     \/     \/     \/     \/"
Locate , TopLcol + 2
Print ""

Locate , TopLcol + 2
Print "A typhoon-class exact & wildcards & Levenshtein Distance (Wagner-Fischer) searcheress"
Print ""
Locate , TopLcol + 2
Print "Note1: The dump (resultant hits) goes to Kazahana.txt file."
Locate , TopLcol + 2
Print "Note2: Nine SLOW wildcards are available:"
Locate , TopLcol + 2
Print "      wildcard '*' any character(s) or empty,"
Locate , TopLcol + 2
Print "      wildcard '.' any ALPHA character(s) or empty,"
Locate , TopLcol + 2
Print "      wildcard '`' any NON-ALPHA character(s) or empty,"
Locate , TopLcol + 2
Print "      wildcard '@'/'#' any character {or empty}/{and not empty},"
Locate , TopLcol + 2
Print "      wildcard '^'/'$' any ALPHA character {or empty}/{and not empty},"
Locate , TopLcol + 2
Print "      wildcard '|'/'~' any NON-ALPHA character {or empty}/{and not empty}."
Locate , TopLcol + 2
Print "Note3: Two FAST wildcards are available:"
Locate , TopLcol + 2
Print "      wildcard '&' any character(s) or empty,"
Locate , TopLcol + 2
Print "      wildcard '+' any character and not empty."
Locate , TopLcol + 2
Print "Note4: Don't mix SLOW and FAST, the SLOW overrides the FAST."
If shrunkP = 0 Then
    Locate , TopLcol + 2
    Print "Exact Case-Sensitive 16-threaded search for all lines up to 26208 chars long:"
    Locate , TopLcol + 2
    Print "Example1: Arnold"
    Locate , TopLcol + 2
    Print "Example2: "; Chr$(34); "metal fatigue"; Chr$(34)
    Locate , TopLcol + 2
    Print "Wildcard Case-Insensitive 16-threaded search for all lines up to 26208 chars long:"
    Locate , TopLcol + 2
    Print "Example3: "; Chr$(34); "out^^^^^^^^^^^^^ize*"; Chr$(34)
```

```
02,506        Locate , TopLcol + 2
02,507        Print "         Possible hits within a line starting with O|o: outhyperbolize, OUTSIZE, outsized"
02,508        Locate , TopLcol + 2
02,509        Print "NORMAL/EXHAUSTIVE Fuzzy Case-Insensitive 16-threaded search for all lines up to 156/26208 chars long:"
02,510        Locate , TopLcol + 2
02,511        Print "Example4: 3 psychedlicize"
02,512        Locate , TopLcol + 2
02,513        Print "         This line '1234psychedlicize' won't match."
02,514        Locate , TopLcol + 2
02,515        Print "Example5: 2e edelvais"
02,516        Locate , TopLcol + 2
02,517        Print "         Possible hits: edelweiss, edelweisses, psychedelicism"
02,518        Locate , TopLcol + 2
02,519        Print "Note5: Levenshtein search can be NORMAL, as in Example4, matching the whole line."
02,520        Locate , TopLcol + 2
02,521        Print "Note6: Levenshtein search can be EXHAUSTIVE, as in Example5, if LD is postfixed with 'e',"
02,522        Locate , TopLcol + 2
02,523        Print "       matching each position in the line."
02,524        Locate , TopLcol + 2
02,525        Print "Note7: The Needle/Pattern below is set by default as the Current Word or the CLIPBOARD."
02,526    End If
02,527    Color 7, BckGRcolor
02,528    HelpLine$ = "Edit keys, allowed: Home, Left, Right, End, Backspace, Del"
02,529    Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print HelpLine$
02,530    _Display
02,531    AddToKazahana$ = InputLine$(TopLrow + 39 - shrunkP, TopLcol + 2, BottomRcol - TopLcol - 1 - 2)
02,532    StartSt$ = Date$ + " " + Time$
02,533    Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print String$(Len(HelpLine$), " ")
02,534    Locate TopLrow + 41 - shrunkP, TopLcol + 2: Print "Start: "; StartSt$
02,535    Locate 1, 1, 0, 30, 31
02,536    _Display
02,537
02,538    TimeK1# = Timer(.01)
02,539    $If WINDOWS Then
02,540        PRINT #13, CHR$(34) + PSPlike$ + "Kazahana.exe" + CHR$(34) + " " + AddToKazahana$ + " " + CHR$(34) + LoadedFile$ + CHR$(34) + " 30539"
02,541    $End If
02,542    $If WINDOWS Then
02,543        'SHELL _DONTWAIT _HIDE CHR$(34) + PSPlike$ + "Kazahana_Hexadecad_GCC_472_SSE41_32bit.exe" + CHR$(34) + " " + AddToKazahana$ + " " + CHR$(34) + LoadedFile$ + CHR$(34) + " 1539" '9
is odd therefore no dump of pattern field, 9 is bigger than 4 therefore case insensitive wildcard search.
02,544        SHELL _HIDE CHR$(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.
02,545    $Else
02,546        Shell _Hide Chr$(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.

02,547    $End If
02,548    TimeK2# = Timer(.01)
02,549    SearchTime! = TimeK2# - TimeK1#
02,550    If SearchTime! < 0 Then SearchTime! = (86400 - TimeK1#) + TimeK2#
02,551    Locate TopLrow + 42 - shrunkP, TopLcol + 2: Print "Ready: "; Date$ + " " + Time$
02,552    f = FreeFile
02,553    Open "Kazahana.txt" For Binary As #f
02,554    KazLen = LOF(f)
02,555    Close #f
02,556    SearchRate = FileSize / (SearchTime! + 0.0001)
02,557    Locate TopLrow + 43 - shrunkP, TopLcol + 2: Print "Size of Kazahana.txt: "; AddCommas$(KazLen); " bytes"; "; Search-and-Dump Speed: "; AddCommas$(SearchRate); " bytes/s";
02,558    Locate TopLrow + 44 - shrunkP, TopLcol + 2: Print "Press Esc...";
02,559    _Display
02,560    $If WINDOWS Then
02,561        PLAY "v20l20g"
02,562    $End If
02,563    Do While InKey$ <> Chr$(27)
02,564        _Limit 30
02,565    Loop
02,566    Locate 1, 1, 1, 30, 31
02,567
02,568    'E:\_KAZE_Smxrt_Benchmarks\QB64_kit_v1.4_2.48 GB\qb64>Kazahana_Hexadecad_GCC_472_SSE41_32bit.exe
02,569    '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.
02,570    'Usage: Kazahana [AtMostLevenshteinDistance][e] string textualfile MasterBufferSize
02,571    'Note0: MasterBufferSize is in KB, consider 1024, 3072, 7168 or bigger (up to 2GB). Three additional flags were mapped on this value: all dump
02,572    '       lines (except fuzzy's) will have/lack pattern-source info when the number is even/odd respectively, see Examples #5 and #6.
02,573    '       When MasterBufferSize ends in 0, then No-Dump i.e. hits are only counted.
02,574    '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.
02,575    'Note1: There are three regimes: exact, wildcards and fuzzy searches. First two kick in when 3 parameters are given, fuzzy when 4.
02,576    'Note2: What decides whether exact or wildcards? Of course presence of at least one wildcard. To see exact search see Example #4.
02,577    'Note3: Exact search hits with 'Railgun_Trolldom', not 'Railgun_Sekireigan_Wolfram'.
02,578    'Note4a: Incoming string is automatically lowercased for fuzzy searches i.e. they are case insensitive.
02,579    'Note4b: Incoming string is NOT automatically lowercased for wildcards searches when MasterBufferSize ends in 0..4 i.e. they are case sensitive.
02,580    'Note4c: Incoming string is automatically lowercased for wildcards searches when MasterBufferSize ends in 5..9 i.e. they are case insensitive.
02,581    'Note5: Incoming string could be up to 26208/156 chars for Exact&Wildcard&ExhaustiveFuzzy/Fuzzy respectively.
02,582    'Note5a: Since 2013-Nov-21 Levenshtein search exits not when the incoming line is bigger than 156 chars, now it just skips longer lines.
02,583    'Note5b: Since 2013-Dec-05 Levenshtein search can be EXHAUSTIVE if LD is postfixed with 'e'.
02,584    'Note6: Incoming textualfile could be bigger than 4GB.
02,585    'Note7: Each line should end with [CR]LF, that is Windows or/and UNIX style.
02,586    'Note8: The dump goes to Kazahana.txt file.
02,587    'Note9a: Nine SLOW wildcards are available:
02,588    '        wildcard '*' any character(s) or empty,
02,589    '        wildcard '.' any ALPHA character(s) or empty,
02,590    '        wildcard '`' any NON-ALPHA character(s) or empty,
02,591    '        wildcard '@'/'#' any character {or empty}/{and not empty},
02,592    '        wildcard '^'/'$' any ALPHA character {or empty}/{and not empty},
02,593    '        wildcard '|'/'~' any NON-ALPHA character {or empty}/{and not empty}.
02,594    'Note9b: Two FAST wildcards are available:
02,595    '        wildcard '&' any character(s) or empty,
02,596    '        wildcard '+' any character and not empty.
02,597    '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.
02,598    'Example1: E:\>Kazahana 0 ramjet MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
02,599    'Example2: E:\>Kazahana 3 psychedlicize MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
02,600    'Example3: E:\>Kazahana "psyched^^^^^ize^" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
```

```
02,601     'Example4: E:\>Kazahana "metal fatigue" enwiki-20121201-pages-articles.xml 7168
02,602     'Example5: E:\>Kazahana "out^^^^^^^^^^^^^ize*" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1536
02,603     '          E:\>type Kazahana.txt
02,604     '          [out^^^^^^^^^^^^^ize*] outhyperbolize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
02,605     '          [out^^^^^^^^^^^^^ize*] outsize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
02,606     '          [out^^^^^^^^^^^^^ize*] outsized /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
02,607     '          [out^^^^^^^^^^^^^ize*] outstrategize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
02,608     '          [out^^^^^^^^^^^^^ize*] outtyrannize /MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd/
02,609     'Example6: E:\>Kazahana "out^^^^^^^^^^^^^ize*" MASAKARI_General-Purpose_Grade_English_Wordlist_r3_316423_words.wrd 1537
02,610     '          E:\>type Kazahana.txt
02,611     '          outhyperbolize
02,612     '          outsize
02,613     '          outsized
02,614     '          outstrategize
02,615     '          outtyrannize
02,616     'Example7: E:\>Kazahana 2e edelvais MASAKARI_General-Purpose_Grade_English_Wordlist.wrd 1024
02,617     '          E:\>type Kazahana.txt
02,618     '          bordelais
02,619     '          bordelaise
02,620     '          edelweiss
02,621     '          edelweisses
02,622     '          foredevised
02,623     '          predellas
02,624     '          psychedelicism
02,625     'Info1: One second seems to have 1,000 clocks.
02,626
02,627     'E:\_KAZE_Smxrt_Benchmarks\QB64_kit_v1.4_2.48 GB\qb64>
02,628 End Sub
02,629
02,630 Function InputLine$ (Irow, Icol, LineLen)
02,631     Shared CurrentWord$
02,632     Locate Irow, Icol, 1, 30, 31
02,633     MaxCol = Icol + LineLen - 1
02,634     CurCol = Icol 'Icol..MaxCol
02,635     If CurrentWord$ = "" Then
02,636         If InStr(_Clipboard$, Chr$(10)) Then '_CLIPBOARD$ could be multi-line, so get the first
02,637             a2$ = Left$(_Clipboard$, InStr(_Clipboard$, Chr$(10)) - 1)
02,638             If Right$(a2$, 1) = Chr$(13) Then a2$ = Left$(a2$, Len(a2$) - 1)
02,639             a$ = Left$(a2$, LineLen - 1) '""
02,640         Else
02,641             a$ = Left$(_Clipboard$, LineLen - 1) '""
02,642         End If
02,643         CurCol = Icol + Len(a$)
02,644     Else
02,645         a$ = CurrentWord$
02,646     End If
02,647     Do
02,648         key$ = InKey$
02,649         If key$ <> "" Then
02,650             code% = Asc(key$):
02,651             If code% Then ' ASC returns any value greater than 0
02,652                 Select Case Asc(key$)
02,653                     Case 27:
02,654                         a$ = "": CurCol = Icol
02,655                     Case 13:
02,656                         If a$ <> "" Then Exit Do
02,657                     Case 8: 'Backspace
02,658                         If CurCol <> Icol Then
02,659                             If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
02,660                                 a$ = Left$(a$, Len(a$) - 1)
02,661                             Else 'inhere we are not at the end i.e. 1..len(a$)-1
02,662                                 a$ = Left$(a$, (CurCol - Icol) - 1) + Right$(a$, Len(a$) - (CurCol - Icol))
02,663                             End If
02,664                             CurCol = CurCol - 1
02,665                         End If
02,666                     Case 32 TO 127 - 1:
02,667                         If Len(a$) < LineLen - 1 Then 'a$ = a$ + key$: CurCol = CurCol + 1 '-1 because cursor goes next to the field otherwise
02,668                             If CurCol = Icol + Len(a$) Then ' this condition includes a$ = ""
02,669                                 a$ = a$ + key$
02,670                             Else 'inhere we are not at the end i.e. 1..len(a$)-1
02,671                                 ' Insert at (CurCol - Icol) + 1:
02,672                                 a$ = Left$(a$, (CurCol - Icol)) + key$ + Right$(a$, Len(a$) - (CurCol - Icol))
02,673                             End If
02,674                             CurCol = CurCol + 1
02,675                         End If
02,676                 End Select
02,677             Else
02,678                 Select Case Asc(key$, 2)
02,679                     Case 75: If CurCol > Icol Then CurCol = CurCol - 1
02,680                     Case 77: If CurCol < MaxCol And CurCol < Icol + Len(a$) Then CurCol = CurCol + 1
02,681                     Case 71: CurCol = Icol 'Home
02,682                     Case 79: CurCol = Icol + Len(a$) 'End
02,683                     Case 83: 'Del
02,684                         If a$ <> "" Then
02,685                             a$ = Left$(a$, (CurCol - Icol)) + Right$(a$, Len(a$) - (CurCol - Icol) - 1)
02,686                         End If
02,687                     Case 82: 'Ins
02,688                 End Select
02,689             End If
02,690         End If
02,691         Locate Irow, Icol, 1, 30, 31
02,692         Print String$(LineLen, " ");
02,693         Locate Irow, Icol, 1, 30, 31
02,694         Print a$;
02,695         Locate Irow, CurCol, 1, 30, 31
02,696         _Display
02,697         _Limit 30
02,698     Loop
02,699     InputLine$ = a$
02,700 End Function
```

```
02,701
02,702 Sub NextFrame (TopRow)
02,703     Shared ASCIIFrame
02,704     Shared PSPlike$
02,705     Locate TopRow, 73
02,706     ASCIIFrame = ASCIIFrame + 1
02,707     If ASCIIFrame > 20 Then ASCIIFrame = 1 '1..20
02,708     f = FreeFile
02,709     fr$ = LTrim$(Str$(ASCIIFrame))
02,710     If Len(fr$) = 1 Then fr$ = "0" + fr$
02,711     If _FileExists(PSPlike$ + "glass" + fr$ + ".txt") Then
02,712         Open PSPlike$ + "glass" + fr$ + ".txt" For Input As #f
02,713         Do While Not EOF(f)
02,714             Line Input #f, a$
02,715             Locate , 73
02,716             Print a$
02,717         Loop
02,718         Close #f
02,719     End If
02,720 End Sub
02,721
02,722 Sub NextFramePEN (TopRow)
02,723     Shared ASCIIFramePEN
02,724     Shared PSPlike$
02,725     Shared XdimCOL
02,726     If XdimCOL = 198 Then HorizPos = 128 Else HorizPos = 70
02,727     Locate TopRow, HorizPos
02,728     ASCIIFramePEN = ASCIIFramePEN + 1
02,729     If ASCIIFramePEN > 24 Then ASCIIFramePEN = 1 '1..20
02,730     f = FreeFile
02,731     fr$ = LTrim$(Str$(ASCIIFramePEN))
02,732     If Len(fr$) = 1 Then fr$ = "0" + fr$
02,733     If _FileExists(PSPlike$ + "pen" + fr$ + ".txt") Then
02,734         Open PSPlike$ + "pen" + fr$ + ".txt" For Input As #f
02,735         Do While Not EOF(f)
02,736             Line Input #f, a$
02,737             Locate , HorizPos
02,738             Print Right$(a$, 80 - 23)
02,739         Loop
02,740         Close #f
02,741     End If
02,742 End Sub
02,743
02,744 Sub ShowF1
02,745     PostFix$ = ""
02,746     'IF ToLoadOrNotFlag = 0 THEN PostFix$ = PostFix$ + "_External" ELSE PostFix$ = PostFix$ + "_Fast"
02,747     If WrapFlag = 1 Then PostFix$ = PostFix$ + "_Wrapper" Else PostFix$ = PostFix$ + "_Vanilla"
02,748     Print "Masakari, revision 8+" + PostFix$ + ", written in QB64 by Kaze, source code downloadable at https://www.qb64.org/forum"
02,749
02,750     $If WINDOWS Then
02,751         PRINT "Usage: Masakari [filename]|[/help]"
02,752     $Else
02,753         Print "Usage: Masakari [filename]|[-h]"
02,754     $End If
02,755     Print "Note: The 'filename' could be a filelist, i.e. a list of filenames (see Space and Double-Left-Click)."
02,756     Print
02,757     Print "Mouse:"
02,758     Print "       Button 1 - sets the cursor and the inverse line to the chosen position"
02,759     Print "       Button 3 - Burst PgDn"
02,760     Print "       Wheel Up/Dn - Up/Dn"
02,761     Print "       Button 2 + Wheel Up - going to the top left position"
02,762     Print "       Button 2 + Wheel Dn - going to the bottom left position"
02,763     Print "       Button 2 + dragging (sideways) for at least 100 columns/cells (within 2 seconds) - same as Alt+X, Alt+Q"
02,764     Print "       Button 2 + dragging (from top to bottom) for at least 5 lines/cells (within 2 seconds) - same as PgUp"
02,765     Print "       Button 2 + dragging (from bottom to top) for at least 5 lines/cells (within 2 seconds) - same as PgDn"
02,766     Print "       Button 1 + Button 1 (within 0.33 seconds) - Double-Left-Click - PgUp or loads the inverse line (if it is a valid file)"
02,767     Print "       Button 2 + Button 2 (within 0.33 seconds) - Double-Right-Click - PgDn"
02,768     Print "       LShift + Button 1/2 - Burst Up/Dn"
02,769     Print "       LCtrl + Button 1/2 - Burst PgUp/PgDn"
02,770     Print "       LAlt + Button 1 = LCtrl+Home - going to the top left position"
02,771     Print "       LAlt + Button 2 = LCtrl+End - going to the bottom left position"
02,772     Print "Keyboard:"
02,773     Print "       F1 - this help panel"
02,774     Print "       Up/Dn/PgUp/PgDn"
02,775     Print "       Left/Right - still no sideways scroll"
02,776     Print "       LCtrl+Home - going to the top left position"
02,777     Print "       LCtrl+End  - going to the bottom left position"
02,778     Print "       Alt+X or Alt+Q - quit to the system, without demanding keypress"
02,779     Print "       Space - loads the highlighted line (if it is an actual filename)"
02,780     Print "       RCtrl|RShift - (can hold it) highlighting all unfamiliar words to masakari.wrd file/wordlist"
02,781     Print "       LCtrl+F3 - Search Panel, Kazahana (with 16 threads) searches the file, currently displayed"
02,782     Print "Benchmarking:"
02,783     Print "       LAlt+RAlt - Reporting (in the status line in red color) the time for load"
02,784     Print "       LCtrl+RCtrl - Reporting (in the status line in red color) the time for PgDn-ing (page-by-page) the entire file"
02,785     Print "       LShift+RShift - Reporting (in the status line in red color) the time for Down-ing (line-by-line) the entire file"
02,786     Print "       Esc - Stops the auto-scrolling started with LCtrl+RCtrl or LShift+RShift"
02,787     Print "Note1: During the LShift+RShift auto-scroll, you can still press/hold RCtrl, during the LCtrl+RCtrl, you cannot."
02,788     Print "Note2: During the auto-scroll, you can still press PgUp, PgDn, LCtrl+Home, LCtrl+End, Esc."
02,789     Print "Note3: The 'Vanilla' sub-variant loads textual files without wrapping the lines."
02,790     Print "Note4: The 'Wrapper' sub-variant makes the text file viewable without side/lateral scroll."
02,791     Print "       If unwrappable lines exist then those lines are dumped to filename+"; Chr$(34); ".unwrappable"; Chr$(34); ","
02,792     Print "       otherwise, the wrapped lines are dumped to filename+"; Chr$(34); ".wrapped"; Chr$(34); ", and auto-loaded."
02,793     Print "       If wrapped file exists during start then it is used, not re-created."
02,794 End Sub
02,795
```

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 () {
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
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 }
```

1251toGesch.bas:

```
REM Converting windows_cp1251 to Gesch codpage

$SCREENHIDE
$CONSOLE
_CONSOLE ON
_CONSOLETITLE "Gesch codepage convertor"
_DEST _CONSOLE

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.
REM      128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143
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
REM      144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159
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
REM      160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175
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
REM      176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191
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
REM Cyrillic:
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
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
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
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 --\
REM                                                                                       |
REM Microsoft_pc_cpGESCH: 'Gesch is Sanmayce's layout, combining the MIK and 437, in this way:                    |
REM                                                                                       |
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   |
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   |
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   |
REM                                                                                       |
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   |
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   |
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   |
REM                                                                                       |
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 <-/
REM      240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255
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

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

DIM byt AS STRING * 1
IF COMMAND$ = "" THEN PRINT "Usage: 1251toGesch.exe filename": SYSTEM
OPEN COMMAND$ FOR BINARY AS #1
OPEN COMMAND$ + ".Gesch" FOR BINARY AS #2
PRINT "Converting"; LOF(1); "bytes..."
FOR i& = 1 TO LOF(1)
    GET #1, i&, byt$
    SELECT CASE ASC(byt$)
        CASE 130:
            bytNEW$ = CHR$(242)
            PUT #2, , bytNEW$

        CASE 132:
            bytNEW$ = CHR$(246)
            PUT #2, , bytNEW$

        CASE 133:
            bytNEW$ = CHR$(255)
            PUT #2, , bytNEW$

        CASE 145:
            bytNEW$ = CHR$(240)
            PUT #2, , bytNEW$

        CASE 146:
            bytNEW$ = CHR$(241)
            PUT #2, , bytNEW$

        CASE 147:
            bytNEW$ = CHR$(244)
            PUT #2, , bytNEW$

        CASE 148:
            bytNEW$ = CHR$(245)
            PUT #2, , bytNEW$

        CASE 152:
            bytNEW$ = CHR$(0)
            PUT #2, , bytNEW$

        CASE 160:
            bytNEW$ = CHR$(0)
            PUT #2, , bytNEW$

        CASE 162:
            bytNEW$ = CHR$(249)
            PUT #2, , bytNEW$

        CASE 168:
            bytNEW$ = CHR$(250)
            PUT #2, , bytNEW$

        CASE 184:
            bytNEW$ = CHR$(251)
            PUT #2, , bytNEW$
```

```
            'Not in the tables above, but should be changed [
        CASE 171:
            bytNEW$ = CHR$(252)
            PUT #2, , bytNEW$
        CASE 187:
            bytNEW$ = CHR$(253)
            PUT #2, , bytNEW$
        CASE 150:
            bytNEW$ = CHR$(196)
            PUT #2, , bytNEW$
        CASE 151:
            bytNEW$ = CHR$(196)
            PUT #2, , bytNEW$
        CASE 173:
            bytNEW$ = CHR$(196)
            PUT #2, , bytNEW$
            'Not in the tables above, but should be changed ]

        CASE 128 + 16 * 4 TO 128 + 16 * 5 - 1:
            bytNEW$ = CHR$(ASC(byt$) - (4 - 0) * 16)
            PUT #2, , bytNEW$
        CASE 128 + 16 * 5 TO 128 + 16 * 6 - 1:
            bytNEW$ = CHR$(ASC(byt$) - (5 - 1) * 16)
            PUT #2, , bytNEW$
        CASE 128 + 16 * 6 TO 128 + 16 * 7 - 1:
            bytNEW$ = CHR$(ASC(byt$) - (6 - 2) * 16)
            PUT #2, , bytNEW$
        CASE 128 + 16 * 7 TO 128 + 16 * 8 - 1:
            bytNEW$ = CHR$(ASC(byt$) - (7 - 6) * 16)
            PUT #2, , bytNEW$
        CASE ELSE:
            PUT #2, , byt$
    END SELECT
NEXT
CLOSE #1, #2
SYSTEM
```

鉞 MASAKARI

The masakari is mainly used to smooth down and finish timber. There are two types of masakari. One has a longer handle and is used mainly for lumbering work, while the other type with a shorter handle, also called the carpenters' broadax (daiku-masakari), is used for rough carpentry work or for making wedges from scraps of wood. A carpenter's hewing ax. This ax was an essential tool for rough hewing work and for smoothing logs to be used as pillars and beams. It is still used by carpenter's who work on shrines and temples.

Ono and Masakari as religious symbols

The animistic tradition from ancient times state that deities descend to and reside in the mountains. For lumbermen, the mountain was therefore a sacred territory which required strict ritual abstentions to be entered. The ax has been closely related with this religious revering of the mountain and its trees. For example, the first act amongst the myriad of Shinto rituals carried out before the lumbering for the rebuilding of the Ise Shrine every 20 years, is the cutting into a tree with a ritually purified ax (imi-ono). Moreover in the festival of the pillar (Onbashira-matsuri) at the Suwa shrine, a vermillion-lacquered ax is used to cut down a tree which is to become the sacred pillar.

In Buddhist symbolism the ax also acquires the power of cutting off evil, and there are numerous existing statues of bodhisattva holding axes. Shugen-do, a traditional Japanese religion born out of an amalgam of different religions including Shintoism and Buddhism which has a particular connection with mountains, regards the ax as one of the symbolic objects to be carried by practitioners when going into mountains for ascetic training.

Ax is also an important (heavenly/carpentry) instrument in Laoism, a few excerpts from pseudo-chapter 74 of 'Dao De Jing':

Translation: **Lin Yutang**
And to take the place of the executioner
Is like handling the hatchet for the master carpenter.
He who handles the hatchet for the master carpenter
Seldom escapes injury to his hands.

Translation: **Gu Zhengkun**
If one wants to kill on behalf of the executioner,
It is like chopping wood on behalf of the master carpenter.
There are few who can escape cutting their own hands
When they chop wood on behalf of the master carpenter.

Translation: **Ch'u Ta-Kao**
Only the Supreme Executioner kills.
To kill in place of the Supreme Executioner is to hack instead of a greater carpenter.
Now if one hacks in place of a great carpenter one can scarcely avoid cutting one's own hand.

Translation: **Hua-Ching Ni**
To become the executioner of artificial righteousness is like the inexperienced lad who would brandish a sharp axe of a master carpenter.
He can seldom escape cutting himself.

Translation: **Witter Bynner**
Nature is executioner.
When man usurps the place,
A carpenter's apprentice takes the place of the master:
And 'an apprentice hacking with the master's axe
May slice his own hand.'

Translation: **Hu Xuezhi**
Therefore, there always exists the executioner in charge of inflicting death.
To inflict death in place of the executioner,
Is equal to finishing an excellent woodwork in place of a master carpenter.
Of those who finish an excellent woodwork in place of a master carpenter,
Few will not hurt their hands.

Translation: J.J.L. Duyvendak
There is always a Chief Executioner who kills.
To kill in place of the Chief Executioner may be called: to chop in place of the Master Carpenter.
Now in chopping in place of the Master Carpenter, few will escape maiming their hands!

Translation: **Spurgeon Medhurst**
There is one who inflicts sentence of death.
To usurp his functions and to kill would be to assume the role of the Master-Carpenter.
There are few who can act as Master-Carpenter without cutting their hands.

Translation: **Richard Wilhelm**

There is always a power of death that kills.
To kill instead of leaving killing to this power of death
is as if one wanted to use the axe oneself
instead of leaving it to the carpenter.
Whosoever would use the axe
instead of leaving it to the carpenter
shall rarely get away
without injuring his hand.

KINTARO - From the folklore of Japan - Legendary symbol of virtue and strength

Kintaro is a beloved legendary and symbolic figure from Japan. Like many legendary figures he appears in both history and mythology. According to classic Japanese literature he was fathered by a great Red Dragon (the thunder god - see below) who visited his mountain sorceress mother in a dream. She awoke amidst powerful claps of thunder and knew at once she was with child. Kintaro means "Golden Boy" and his jealous uncle sought to kill him. His mother took him and fled into the Hakone mountains to the deepest forests of Mount Kintoki. Growing up deep in the forest his beautiful spirit caused him to become a special friend to all the wild animals, most especially the rabbits and the bears. He loved to play with his animal friends about the rocks of the Yuhi no Taki Falls. So strong was he as a boy and so gifted at Sumo wrestling that he could throw down a bear. He was a very good boy, rosy-cheeked and chubby and always carried a hatchet, the Japanese symbol of the thunder god and is usually depicted riding his beloved bear.

This documentlet is available thanks to sources:
1] TAKENAKA CARPENTRY TOOLS MUSEUM
   4-18-25, Nakayamate-dori, Chuo-ku,
   Kobe 650-0004, Japan
2] The Asian Myths & Legends Art page of Howard David Johnson
3] Japanese Architecture and Art Net Users System
4] My web-page: www.sanmayce.com

Kaze,
2013-Jan-07