

COMPARE.BAS	0.02	6/25/2025
-------------	------	-----------

D E S C R I P T I O N :

- Program for Generating Historical Time Tables

R E F E R E N C E S :

9. Remember the former things of old: for I [am] God, and [there is] none else; [I am] God, and [there is] none like me,
10. Declaring the end from the beginning, and from ancient times [the things] that are not [yet] done, saying, My counsel shall stand, and I will do all my pleasure:
(Isaiah 46)
18. For verily I say unto you, Till heaven and earth pass, one jot or one tittle shall in no wise pass from the law, till all be fulfilled.
(St. Matthew 5)

The Bible

"The Bible, Authorized Version" by King James 1769, and Webster Update 1833, Oxford University Press, 1994

H A N D L I N G :

10/14/2024 - 6/25/2025 Norbert Suedland, Aalen

P R E P A R A T I O N**F i r s t C o m m a n d s :**

```
OPTION BASE 1           'Arrays start by index No.`1`!  
'OPTION EXPLICIT       'This is useful with Visual Basic  
COMMON HistoricChoice%, WorkingPlace$, WorkingTime$, CountingMode%
```

C o n s t a n t s :**A n n o u n c e S u b r o u t i n e s :**

```
DECLARE SUB Present (Colors%, Quest$, Buffer$, Offs%, k$, Area%, Of%, ly%)  
DECLARE SUB CompareFiles (Argument1$, Argument2$)  
DECLARE SUB LINEINPUT (File%, Line$)  
DECLARE SUB Pause ()
```

```
DECLARE FUNCTION FoundFile% (Path$)  
DECLARE FUNCTION LASTINSTR% (Text$, Pattern$)
```

```

DECLARE FUNCTION SIZEOF% (Text$)
DECLARE FUNCTION STRLEN% (Text$, EndOfLine$)
DECLARE FUNCTION KeyInput$ ()

```

```

DEF FNMIN (a, b) = (a + b) / 2 - ABS(a - b) / 2

```

Global Variables:

```

'Set Stack Size:
'-----'
IF CountingMode% = 0 THEN
    CLEAR , , 4096
END IF

```

Local Variables:

```

'DIM AS STRING:
'-----'
DIM Order$
DIM Directory$

```

MAIN PART

```

ON ERROR GOTO ErrorHandler

```

```

'Compare Finally the Results:
'-----'
Directory$ = ENVIRON$("HISTORIKTEMP")
IF Directory$ <> "" THEN GOTO Case12
Directory$ = ENVIRON$("QBASICTEMP")
IF LEN(Directory$) > 0 AND RIGHT$(Directory$, 1) <> "\" THEN
    Directory$ = Directory$ + "\"
END IF
Directory$ = Directory$ + "HISTORIC.TMP"
IF Directory$ <> "HISTORIC.TMP" THEN GOTO Case12
Directory$ = ENVIRON$("HISTORIK.TMP")
IF Directory$ <> "" THEN GOTO Case12
GOTO EndOfProgram

```

```

'====='
Case12:
'====='

```

```

IF LEN(Directory$) > 0 AND RIGHT$(Directory$, 1) <> "\" THEN
    Directory$ = Directory$ + "\"
END IF
IF FoundFile%(Directory$ + "1\*.*) = 0 THEN GOTO Case23
IF FoundFile%(Directory$ + "2\*.*) = 0 THEN GOTO Case13
CALL CompareFiles(Directory$ + "1\*.HQL", Directory$ + "2\*.HQL")
CALL Pause

```

```
Order$ = "DIR /P " + Directory$ + "1\*.LST"
SHELL Order$
CALL Pause
Order$ = "DIR /P " + Directory$ + "2\*.LST"
SHELL Order$
CALL Pause

'====='
Case13:
'====='
    IF FoundFile%(Directory$ + "3\*.*)" = 0 THEN GOTO Case14
    CALL CompareFiles(Directory$ + "1\*.HQL", Directory$ + "3\*.HQL")
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "1\*.LST"
    SHELL Order$
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "3\*.LST"
    SHELL Order$
    CALL Pause

'====='
Case14:
'====='
    IF FoundFile%(Directory$ + "4\*.*)" = 0 THEN GOTO Case23
    CALL CompareFiles(Directory$ + "1\*.HQL", Directory$ + "4\*.HQL")
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "1\*.LST"
    SHELL Order$
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "4\*.LST"
    SHELL Order$
    CALL Pause

'====='
Case23:
'====='
    IF FoundFile%(Directory$ + "2\*.*)" = 0 THEN GOTO Case34
    IF FoundFile%(Directory$ + "3\*.*)" = 0 THEN GOTO Case24
    CALL CompareFiles(Directory$ + "2\*.HQL", Directory$ + "3\*.HQL")
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "2\*.LST"
    SHELL Order$
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "3\*.LST"
    SHELL Order$
    CALL Pause

'====='
Case24:
'====='
    IF FoundFile%(Directory$ + "4\*.*)" = 0 THEN GOTO EndOfProgram
    CALL CompareFiles(Directory$ + "2\*.HQL", Directory$ + "4\*.HQL")
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "2\*.LST"
    SHELL Order$
    CALL Pause
    Order$ = "DIR /P " + Directory$ + "4\*.LST"
    SHELL Order$
```

CALL Pause

'====='

Case34:

'====='

 IF FoundFile%(Directory\$ + "4*.*)" = 0 THEN GOTO EndOfProgram

 CALL CompareFiles(Directory\$ + "3*.HQL", Directory\$ + "4*.HQL")

 CALL Pause

 Order\$ = "DIR /P " + Directory\$ + "3*.LST"

 SHELL Order\$

 CALL Pause

 Order\$ = "DIR /P " + Directory\$ + "4*.LST"

 SHELL Order\$

 CALL Pause

 GOTO EndOfProgram

'===== '

EndOfProgram:

'===== '

 SYSTEM

' _____ END OF THE MAIN PART _____ '

' _____

| _____ ERROR HANDLING _____ |

' _____ '

'===== '

ErrorHandling:

'===== '

 PRINT "ERR = "; ERR; " , ERL = "; ERL

 STOP

 ON ERROR GOTO 0

 GOTO EndOfProgram

' _____ END OF THE ERROR HANDLING _____ '

' _____

| _____ SUBROUTINES AND FUKCTIONS _____ |

' _____ '

'===== '

 SUB CompareFiles (Argument1\$, Argument2\$)

'===== '

 ' Replaces FC.EXE of DOS, because this is not yet always available.

 '

 ' Handling: 9/21/2024 - 6/25/2025 Norbert Suedland, Aalen

 '-----

 CONST Content1\$ = "Content1.\$\$\$"

 CONST Content2\$ = "Content2.\$\$\$"

 DIM Line\$, Line1\$, Line2\$

 DIM Order\$

 DIM OutputText\$

 DIM Path\$, Path1\$, Path2\$

 DIM Record1\$, Record2\$

```

DIM c%
DIM Lines%
DIM File1%, File2%, File3%, File4%
DIM Found%
DIM Length%
DIM Pages%

DIM P1&, P2&

'List the File to be Compared:
'-----'
Pages% = 0
Lines% = 0
Found% = 0
Path1$ = LEFT$(Argument1$, LASTINSTR%(Argument1$, "\"))
Order$ = "DIR " + Argument1$ + " > " + Content1$
200 SHELL Order$
Path2$ = LEFT$(Argument2$, LASTINSTR%(Argument2$, "\"))
Order$ = "DIR " + Argument2$ + " > " + Content2$
201 SHELL Order$

'Work out the File Lists:
'-----'
File1% = FREEFILE
210 OPEN Content1$ FOR BINARY ACCESS READ AS #File1%
DO
    CALL LINEINPUT(File1%, Line1$)
    LOOP UNTIL INSTR(2, Line1$, ":\") > 0
    File2% = FREEFILE
211 OPEN Content2$ FOR BINARY ACCESS READ AS #File2%
DO
    CALL LINEINPUT(File2%, Line2$)
    LOOP UNTIL INSTR(2, Line2$, ":\") > 0
DO
    CALL LINEINPUT(File1%, Line1$)
    IF Line1$ = "" THEN
        CALL LINEINPUT(File1%, Line1$)
    END IF
    IF MID$(Line1$, 9, 1) = SPACE$(1) THEN
        Line$ = RTRIM$(LEFT$(Line1$, 8)) + "."
        Line$ = Line$ + RTRIM$(MID$(Line1$, 10, 3))
        Line1$ = Line$
    ELSE
        DO
            CALL LINEINPUT(File1%, Line1$)
            LOOP UNTIL EOF(File1%)
        END IF
        CALL LINEINPUT(File2%, Line2$)
        IF Line2$ = "" THEN
            CALL LINEINPUT(File2%, Line2$)
        END IF
        IF MID$(Line2$, 9, 1) = SPACE$(1) THEN
            Line$ = RTRIM$(LEFT$(Line2$, 8)) + "."
            Line$ = Line$ + RTRIM$(MID$(Line2$, 10, 3))
            Line2$ = Line$
        ELSE
            DO

```

```
        CALL LINEINPUT(File2%, Line2$)
    LOOP UNTIL EOF(File2%)
END IF
IF Path1$ <> Path2$ THEN
    WHILE NOT EOF(File1%) AND NOT EOF(File2%) AND Line1$ < Line2$
        CALL LINEINPUT(File1%, Line1$)
    WEND
    WHILE NOT EOF(File1%) AND NOT EOF(File2%) AND Line1$ > Line2$
220        CALL LINEINPUT(File2%, Line2$)
    WEND
    IF Line1$ <> Line2$ GOTO EndOfLoop
END IF
IF NOT EOF(File1%) AND NOT EOF(File2%) THEN
    OutputText$ = "Comparing the Files " + Path1$ + Line1$ + " and "
    OutputText$ = OutputText$ + Path2$ + Line2$
    Lines% = Lines% + INT((LEN(OutputText$) - 1) / 80) + 1
    IF INT((Lines% - 1) / 24) > Pages% THEN
        CALL Pause
        Pages% = Pages% + 1
    END IF
    PRINT OutputText$
    File3% = FREEFILE
    Path$ = Path1$ + Line1$
230    OPEN Path$ FOR BINARY ACCESS READ SHARED AS #File3% LEN = 512
        File4% = FREEFILE
        Path$ = Path2$ + Line2$
240    OPEN Path$ FOR BINARY ACCESS READ SHARED AS #File4% LEN = 512
        Record1$ = SPACE$(512)
        Record2$ = SPACE$(512)
        P1% = 1
        P2% = 1
        WHILE NOT EOF(File3%) AND NOT EOF(File4%) AND Found% = 0
250            GET #File3%, P1%, Record1$
260            GET #File4%, P2%, Record2$
            IF Record1$ = Record2$ THEN
                P1% = P1% + 512
                P2% = P2% + 512
            ELSE
                PRINT "FC: The Files are different:"
                Lines% = Lines% + 1
                IF INT((Lines% - 1) / 24) > Pages% THEN
                    CALL Pause
                    Pages% = Pages% + 1
                END IF
                Length% = FNMIN(LEN(Record1$), LEN(Record2$))
                FOR c% = 1 TO Length%
                    IF MID$(Record1$, c%, 1) <> MID$(Record2$, c%, 1) THEN
                        Found% = c%
                        c% = Length%
                    END IF
                NEXT
                Length% = FNMIN(Length% - Found% + 1, 79)
                COLOR 0, 7
                PRINT MID$(Record1$, Found%, Length%);
                COLOR 7, 0
                PRINT
                Lines% = Lines% + 1
                IF INT((Lines% - 1) / 24) > Pages% THEN
```

```

        CALL Pause
        Pages% = Pages% + 1
    END IF
    PRINT " and"
    Lines% = Lines% + 1
    IF INT((Lines% - 1) / 24) > Pages% THEN
        CALL Pause
        Pages% = Pages% + 1
    END IF
    COLOR 0, 7
    PRINT MID$(Record2$, Found%, Length%);
    COLOR 7, 0
    PRINT
    Lines% = Lines% + 1
    IF INT((Lines% - 1) / 24) > Pages% THEN
        CALL Pause
        Pages% = Pages% + 1
    END IF
END IF
WEND
IF Found% = 0 THEN
    OutputText$ = "FC: Found no differences"
    Lines% = Lines% + INT((LEN(OutputText$) - 1) / 80) + 1
    IF INT((Lines% - 1) / 24) > Pages% THEN
        CALL Pause
        Pages% = Pages% + 1
    END IF
    PRINT OutputText$
END IF
CLOSE File4%
CLOSE File3%
END IF
'-----'
EndOfLoop:
'-----'
    LOOP WHILE NOT EOF(File2%) AND NOT EOF(File1%)
    CLOSE File2%
    CLOSE File1%

    'Clear away:
    '-----'
300 KILL Content1$
310 KILL Content2$

END SUB 'CompareFiles _____'

'=====
FUNCTION FoundFile% (Path$)
'=====
    ' Checks, whether 'Path$' can be found and then returns 1, else 0.
    '
    ' Handling:
    ' 10/17/2024 - 5/31/2025 Norbert Suedland, 73431 Aalen, GERMANY
    '-----'
    DIM Order$
    DIM Line$

    DIM File%

```

```

    DIM Result%

    Result% = 0
    Order$ = "DIR " + Path$ + " > Found.Yes"
101 SHELL Order$
    File% = FREEFILE
102 OPEN "Found.Yes" FOR BINARY ACCESS READ AS #File%
    WHILE Result% = 0 AND EOF(File%) = 0
        CALL LINEINPUT(File%, Line$)
        IF MID$(Line$, 9, 4) = " HQL" THEN
            Result% = 1
        END IF
    WEND
    CLOSE #File%
103 KILL "Found.Yes"

104 FoundFile% = Result%
END FUNCTION 'FoundFile% _____'

'=====
FUNCTION KeyInput$
'=====
' Waits for a keyboard input and gives back the corresponding ASCII letter.
'
' Handling:
' 8/18/2001: Norbert Suedland und Eckhard Walter, Adelshofen
' Check:
' 8/18/2001: Norbert Suedland und Eckhard Walter, Adelshofen
' 9/ 6/2002: Norbert Suedland, Aalen
'-----
DIM Answer$          'AS STRING

'Empty the Keyboard Buffer:
'-----
WHILE INKEY$ <> ""
WEND

'Question Keyboard again, until a Key has been Pressed:
'-----
Answer$ = ""
WHILE LEN(Answer$) = 0
    Answer$ = INKEY$
WEND

'Result:
'-----
KeyInput$ = Answer$
END FUNCTION 'KeyInput$ _____'

'=====
FUNCTION LASTINSTR% (Text$, Pattern$)
'=====
' Determines the last position of 'Pattern$' in 'Text$'.
'
' Handling:
' 9/21/2004 Norbert Suedland, 73431 Aalen, GERMANY

```

```

'-----'
DIM Begin%
DIM LastBegin%

Begin% = 0
DO
    LastBegin% = Begin%
    Begin% = INSTR(Begin% + 1, Text$, Pattern$)
LOOP WHILE Begin% > 0
LASTINSTR = LastBegin%

END FUNCTION 'LASTINSTR% _____'

'=====
SUB LINEINPUT (File%, Line$)
'=====
' Replaces the command LINE INPUT to run also under the DOS Box 0.73.
' This box generates by DIR an output like under Unix and can be evaluated
' by LINEINPUT instead of LINE INPUT.
' Therefore the 'File%' must be opened FOR ACCESS BINARY READ (WRITE).
' At the end, a line of 512 characters is returned, thus each line not
' ending by CHR$(10) is taken. This compromise enables speed and is
' sufficient here to evaluate the DIR command.
'
' Handling:
' 8/11/2009: Norbert Suedland, 73431 Aalen, GERMANY
' Check:
' 8/13/2009: Norbert Suedland, 73431 Aalen, GERMANY
'-----'

DIM OldPosition%
DIM Length%
DIM NewPosition%

Line$ = SPACE$(512)
OldPosition% = SEEK(File%)
GET #File%, OldPosition%, Line$
Length% = INSTR(Line$, CHR$(10))
IF Length% <> 0 THEN
    Line$ = LEFT$(Line$, Length% - 1)
    IF INSTR(RIGHT$(Line$, 1), CHR$(13)) <> 0 THEN
        Line$ = LEFT$(Line$, LEN(Line$) - 1)
    END IF
END IF

IF EOF(File%) = 0 OR Length% <> 0 THEN
    NewPosition% = OldPosition% + Length%
    SEEK #File%, NewPosition%
END IF
'PRINT LEN(Line$), Line$
'Pause
END SUB 'LINEINPUT _____'

'=====
SUB Pause
'=====
' Will present a statement in line 25 and wait for a pressed key.
'
' Handling:

```

```

' 8/18/2001 Norbert Suedland und Eckhard Walter, Adelshofen
' Check:
' 8/18/2001 Norbert Suedland und Eckhard Walter, Adelshofen
'-----'
DIM x%      'AS INTEGER
DIM y%      'AS INTEGER
DIM Answer$ 'AS STRING

'Find the Current Cursor Position:
'-----'
x% = CSRLIN
y% = POS(0)

'Delete Line 25 and Present the Statement:
'-----'
LOCATE 25, 1, 0
PRINT SPACE$(80);
Present 1, "", "Press any key to go on", 0, "C", 1, 1, 25

'Wait for a Presed Key:
'-----'
Answer$ = KeyInput$

'Delete Line 25 Again:
'-----'
LOCATE 25, 1, 1
PRINT SPACE$(80);

'Restore Cursor Position:
'-----'
LOCATE x%, y%
END SUB 'Pause _____'

'=====
SUB Present (Colors%, Quest$, Buffer$, Offset%, Kind$, Area%, Areas%, ly%)
'=====
' Presents `Quest$`, followed by `Buffer$` in linee `ly%`.
'
' Meaning of the further parameters:
' `Colors%`   Choice mode 1 (normal) or 2 (emphasized)
' `Offset%`   Shifting possibility within a column
' `Area%`     Wanted column
' `Areas%`    Total number of columns
' `Kind$`     "L" (left hand sided), "C" (centered), "R" (right hand sided)
'
' Handling:
' 8/ 7/2001 - 9/ 4/2001: Norbert Suedland und Eckhard Walter, Adelshofen
'-----'
DIM Length% 'AS INTEGER
DIM x%

SELECT CASE Colors%
CASE 1
COLOR 7, 0
CASE 2
COLOR 15, 0

```

```

END SELECT
Length% = LEN(Quest$) + LEN(Buffer$) + 2
IF LEN(Buffer$) = 0 THEN
    Length% = Length% - 2
END IF
IF Length% > 80 / Areas% THEN
    Length% = 80 / Areas%
    IF LEN(Quest$) >= Length% THEN      'Programming Discrepancy!
        Quest$ = LEFT$(Quest$, Length%)
        Buffer$ = ""
    ELSE
        Pause
        Buffer$ = LEFT$(Buffer$, LEN(Quest$) - Length%)
    END IF
END IF

SELECT CASE Kind$
CASE "L"
    x% = INT(Offset% + (Area% - 1) * 80 / Areas%)
CASE "C"
    x% = INT(Offset% + (Area% - .5) * 80 / Areas% - Length% / 2) + 1
CASE "R"
    x% = INT(Offset% + Area% * 80 / Areas% - Length%)
END SELECT
LOCATE ly%, x%
PRINT Quest$;

IF Buffer$ <> "" THEN
    COLOR 0, 7
    SELECT CASE Colors%
    CASE 1
        PRINT " "; Buffer$; " ";
    CASE 2
        IF Buffer$ <> "" THEN
            x% = POS(0)
            LOCATE ly%, x% + 1
            PRINT Buffer$;
        END IF
    END SELECT
END IF

COLOR 7, 0
END SUB 'Present _____'

'=====
FUNCTION STRLEN% (Text$, EndOfLine$)
'=====
' Will find the string length of `Text$` until to the first occurrence of
' all `EndOfLine$`, like being usual within the C programming language.
' If `EndOfLine$` does not occur, the string length `LEN(Text$)` is
' returned.
'
' Handling:
' 8/ 4/2001:  Norbert Suedland, Munich
' Check:
' 8/ 4/2001:  Norbert Suedland, Munich
'-----
DIM IntermediateResult%      'AS INTEGER

```

```
IntermediateResult% = INSTR(Text$, EndOfLine$)
IF IntermediateResult% = 0 THEN
    IntermediateResult% = LEN(Text$)
ELSE
    IntermediateResult% = IntermediateResult% - 1
END IF

STRLEN% = IntermediateResult%
END FUNCTION 'STRLEN% _____'
```