

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

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

- Relaces FC.EXE and Compares Files.

H A N D L I N G :

9/21/2024 - 6/25/2025 Norbert Suedland, 73431 Aalen, GERMANY

PREPARATION:

'OPTION EXPLICIT 'Useful with Visual Basic

CONST content1\$ = "content1.\$\$\$"

CONST content2\$ = "content2.\$\$\$"

DECLARE FUNCTION LASTINSTR% (text\$, pattern\$)

DECLARE SUB LINEINPUT (file%, line\$)

DECLARE SUB pause ()

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

DIM argument1\$, argument2\$

DIM datarecord1\$, datarecord2\$

DIM line\$, line1\$, line2\$

DIM order\$

DIM outputtext\$

DIM path\$, path1\$, path2\$

DIM c%

DIM datalines%

DIM file1%, file2%, file3%, file4%

DIM found%

DIM length%

DIM pages%

DIM p1&, p2&

MAIN PART:

'Preparation:

'-----'

ON ERROR GOTO errorhandling

pages% = 0

datalines% = 0

found% = 0

```
argument1$ = ENVIRON$("FC_1")
IF argument1$ = "" THEN GOTO endofprogram
path1$ = LEFT$(argument1$, LASTINSTR$(argument1$, "\"))
order$ = "DIR " + argument1$ + " > " + content1$
10  SHELL order$

argument2$ = ENVIRON$("FC_2")
IF argument2$ = "" THEN GOTO endofprogram
path2$ = LEFT$(argument2$, LASTINSTR$(argument2$, "\"))
order$ = "DIR " + argument2$ + " > " + content2$
20  SHELL order$

'Work out Data Lists:
'-----'
file1% = FREEFILE
30  OPEN content1$ FOR BINARY ACCESS READ AS #file1%
    DO
        CALL LINEINPUT(file1%, line1$)
        LOOP UNTIL INSTR(2, line1$, ":\")
        file2% = FREEFILE
40  OPEN content2$ FOR BINARY ACCESS READ AS #file2%
    DO
        CALL LINEINPUT(file2%, line2$)
        LOOP UNTIL INSTR(2, line2$, ":\")
    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$
62      CALL LINEINPUT(file2%, line2$)
        WEND
        IF Zeile1$ <> Zeile2$ GOTO endofloop
    END IF
    IF NOT EOF(file1%) AND NOT EOF(file2%) THEN
        outputtext$ = "Comparing the files " + path1$ + line1$ + " and "
        outputtext$ = outputtext$ + path2$ + line2$
        datalines% = datalines% + LEN(outputtext$) / 80 + 1
        IF INT(datalines% / 24) > pages% THEN
            pause
            pages% = pages% + 1
        END IF
        PRINT outputtext$
        file3% = FREEFILE
        path$ = path1$ + line1$
70      OPEN path$ FOR BINARY ACCESS READ SHARED AS #file3% LEN = 512
        file4% = FREEFILE
        path$ = path2$ + line2$
80      OPEN path$ FOR BINARY ACCESS READ SHARED AS #file4% LEN = 512
        datarecord1$ = SPACE$(512)
        datarecord2$ = SPACE$(512)
        p1& = 1
        p2& = 1
        WHILE NOT EOF(file3%) AND NOT EOF(file4%) AND found% = 0
90          GET #file3%, p1&, datarecord1$
100         GET #file4%, p2&, datarecord2$
            IF datarecord1$ = datarecord2$ THEN
                p1& = p1& + 512
                p2& = p2& + 512
                IF found% = 0 THEN
                    outputtext$ = "FC: Found no differences"
                ELSE
                    outputtext$ = ""
                END IF
            ELSE
                GOSUB difference1
                outputtext$ = ""
            END IF
        WEND
        datalines% = datalines% + INT((LEN(outputtext$) - 1) / 80) + 1
        IF INT(datalines% / 24) > pages% THEN
            pause
            pages% = pages% + 1
        END IF
        IF outputtext$ <> "" THEN
            PRINT outputtext$
            datalines% = datalines% + 1
            IF INT(datalines% / 24) > pages% THEN
                pause
                pages% = pages% + 1
            END IF
            PRINT
        END IF
        CLOSE file4%
        CLOSE file3%
    END IF
'-----'

```

```
endofloop:
'-----'
        LOOP WHILE NOT EOF(file2%) AND NOT EOF(file1%)
        'STOP
        CLOSE file2%
        CLOSE file1%

900 KILL content2$
910 KILL content1$

'=====
endofprogram:
'=====
        SYSTEM
'----- END OF MAIN PART -----'

'=====
' ERROR HANDLING:
'=====

errorhandling:
'=====
        PRINT "ERR = "; ERR, "ERL = "; ERL
        STOP
        ON ERROR GOTO 0
GOTO endofprogram
'----- END OF ERROR HANDLING -----'

'=====
' SUBPROGRAM AND FUNCTION:
'=====

'-----
difference1:
'-----
200 PRINT "FC: Files are different:"
    datalines% = datalines% + 1
    IF INT(datalines% / 24) > pages% THEN
        pause
        pages% = pages% + 1
    END IF
    length% = FNMIN(LEN(datarecord1$), LEN(datarecord2$))
    FOR c% = 1 TO length%
        IF MID$(datarecord1$, c%, 1) <> MID$(datarecord2$, c%, 1) THEN
            found% = c%
            c% = length%
        END IF
    NEXT
    length% = FNMIN(length% - found%, 79)
    outputtext$ = MID$(datarecord1$, found%, length%)
    COLOR 0, 7
    PRINT outputtext$;
    COLOR 7, 0
    PRINT
    datalines% = datalines% + 1
```

```

    IF INT(datalines% / 24) > pages% THEN
        pause
        pages% = pages% + 1
    END IF
    PRINT " and"
    datalines% = datalines% + 1
    IF INT(datalines% / 24) > pages% THEN
        pause
        pages% = pages% + 1
    END IF
    outputtext$ = MID$(datarecord2$, found%, length%)
    COLOR 0, 7
    PRINT outputtext$;
    COLOR 7, 0
    PRINT
    datalines% = datalines% + 1
    IF INT(datalines% / 24) > pages% THEN
        pause
        pages% = pages% + 1
    END IF
RETURN 'difference1 _____'

'=====
FUNCTION LASTINSTR% (text$, pattern$)
'=====
' Determines the last position of 'pattern$' in 'text$'.
'
' Handling:
' 09/21/2024 - 04/12/2025 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$)
'=====
' Replaces the command LINE INPUT to run also under the DOS-Box 0.73.
' This DOS-Box generated with DIR an output like under Unix and can be
' evaluated with LINEINPUT instead of LINE INPUT.
' For this, the 'file%' must have been opened as BINARY ACCESS READ (WRITE).
' At the end, a data line is returned with 512 characters, thus each data
' line, ending with CHR$(10), is caught, yet. This compromise reaches
' speed and is enough here to evaluate DIR.
'
' Handling:
' 08/11/2009: Norbert Suedland, 73431 Aalen, GERMANY
' Check:
' 08/13/2009: Norbert Suedland, 73431 Aalen, GERMANY
'-----
DIM oldposition&

```

```

DIM length&
DIM newposition&

line$ = SPACE$(512)
oldposition& = SEEK(file%)
IF EOF(file%) THEN
    line$ = ""
    GOTO end2
END IF

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$), line$
'STOP
'____'
end2:
'____'
END SUB 'LINEINPUT _____'

'=====
SUB pause
'=====
' Generates a pause, after the screen has been filled with output.
'
' Handling:
' 10/18/2024 - 06/25/2025 Norbert Suedland, 73431 Aalen, GERMANY
'-----'

DIM x%      'AS INTEGER
DIM y%      'AS INTEGER

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

'Delete and Inscribe Data Line 25:
'-----'
LOCATE 25, 1, 0
PRINT SPACE$(80);
LOCATE 25, 1, 1
PRINT " -- continue -- ";

'Wait for Pressed Key:
'-----'
WHILE LEN(INKEY$) = 0
WEND

```

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

'Restore CursorPosition:
'-----'
LOCATE x%, y%
END SUB 'pause _____'
```