

VERGLEIX.BAS	0.02	25. 6.2025
--------------	------	------------

B E S C H R E I B U N G :

- Programm zum Vergleich historischer Zeittafeln

Q U E L L E N - A N G A B E N :

9. Gedenket des Vorigen von Alters her; denn ich bin Gott, und keiner mehr, ein Gott, desgleichen nirgend ist.

10. Der ich verkündige zuvor, was hernach kommen soll, und vorhin, ehe denn es geschieht, und sage: Mein Anschlag besteht, und ich thue Alles, was mir gefällt.

(Jesaja 46)

18. Denn ich sage euch, wahrlich, bis daß Himmel und Erde zergehe, wird nicht zergehen der kleinste Buchstabe, noch ein Titel vom Gesetz, bis daß es alles geschehe.

(Matthäus 5)

Bibel

"Die Bibel, oder die ganze Heilige Schrift Alten und Neuen Testaments" nach der deutschen Uebersetzung Dr. Martin Luthers.", Revision durch Dr. J. Ph. Fresenius, (1751); Druck und Verlag von Heinrich Ludwig Brönner, Frankfurt am Main, 40.Auflage, (1841)

B E A R B E I T U N G :

14.10.2024 - 25. 6.2025

Norbert Südland, Aalen

VORBEREITUNG**Erste Befehle:**

```
OPTION BASE 1           'Felder beginnen beim Eintrag Nr.`1`!  
'OPTION EXPLICIT       'Dies ist nützlich mit Visual Basic  
COMMON HistAuswahl%, Arbeitsplatz$, Zeit$, Zaehlweise%
```

Konstanten:**Unterprogramme anmelden:**

```
DECLARE SUB Ausgabe (Farbe%, Frage$, Puffer$, Offs%, a$, b%, Bereiche%, y%)  
DECLARE SUB Dateivergleich (Argument1$, Argument2$)  
DECLARE SUB LINEINPUT (Datei%, Zeile$)
```

```
DECLARE SUB Pause ()
```

```
DECLARE FUNCTION DateiGefunden% (Pfad$)
```

```
DECLARE FUNCTION LASTINSTR% (Text$, Suchausdruck$)
```

```
DECLARE FUNCTION SIZEOF% (Text$)
```

```
DECLARE FUNCTION STRLEN% (Text$, Ende$)
```

```
DECLARE FUNCTION Taste$ ()
```

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

Globale Variablen:

```
'Stack-Größe festlegen:
```

```
'-----'
```

```
IF Zaehlweise% = 0 THEN
```

```
    CLEAR , , 4096
```

```
END IF
```

Lokale Variablen:

```
'DIM AS STRING:
```

```
'-----'
```

```
DIM Befehl$
```

```
DIM Verzeichnis$
```

HAUPTTEIL

```
ON ERROR GOTO FehlerBehandlung
```

```
'Zum Schluss Ergebnisse vergleichen:
```

```
'-----'
```

```
Verzeichnis$ = ENVIRON$("HISTORIKTEMP")
```

```
IF Verzeichnis$ <> "" THEN GOTO Fall12
```

```
Verzeichnis$ = ENVIRON$("QBASICTEMP")
```

```
IF LEN(Verzeichnis$) > 0 AND RIGHT$(Directory$, 1) <> "\" THEN
```

```
    Verzeichnis$ = Verzeichnis$ + "\"
```

```
END IF
```

```
Verzeichnis$ = Verzeichnis$ + "HISTORIK.TMP"
```

```
IF Verzeichnis$ <> "HISTORIK.TMP" THEN GOTO Fall12
```

```
Verzeichnis$ = ENVIRON$("HISTORIK.TMP")
```

```
IF Verzeichnis$ <> "" THEN GOTO Fall12
```

```
GOTO Programmende
```

```
'====='
```

```
Fall12:
```

```
'====='
```

```
IF LEN(Verzeichnis$) > 0 AND RIGHT$(Directory$, 1) <> "\" THEN
```

```
    Verzeichnis$ = Verzeichnis$ + "\"
```

```
END IF
```

```
IF DateiGefunden%(Verzeichnis$ + "1\*.*)" = 0 THEN GOTO Fall123
IF DateiGefunden%(Verzeichnis$ + "2\*.*)" = 0 THEN GOTO Fall113
CALL Dateivergleich(Verzeichnis$ + "1\*.HQL", Verzeichnis$ + "2\*.HQL")
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "1\*.LST"
SHELL Befehl$
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "2\*.LST"
SHELL Befehl$
CALL Pause

'====='
Fall113:
'====='
IF DateiGefunden%(Verzeichnis$ + "3\*.*)" = 0 THEN GOTO Fall114
CALL Dateivergleich(Verzeichnis$ + "1\*.HQL", Verzeichnis$ + "3\*.HQL")
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "1\*.LST"
SHELL Befehl$
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "3\*.LST"
SHELL Befehl$
CALL Pause

'====='
Fall114:
'====='
IF DateiGefunden%(Verzeichnis$ + "4\*.*)" = 0 THEN GOTO Fall123
CALL Dateivergleich(Verzeichnis$ + "1\*.HQL", Verzeichnis$ + "4\*.HQL")
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "1\*.LST"
SHELL Befehl$
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "4\*.LST"
SHELL Befehl$
CALL Pause

'====='
Fall123:
'====='
IF DateiGefunden%(Verzeichnis$ + "2\*.*)" = 0 THEN GOTO Fall134
IF DateiGefunden%(Verzeichnis$ + "3\*.*)" = 0 THEN GOTO Fall124
CALL Dateivergleich(Verzeichnis$ + "2\*.HQL", Verzeichnis$ + "3\*.HQL")
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "2\*.LST"
SHELL Befehl$
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "3\*.LST"
SHELL Befehl$
CALL Pause

'====='
Fall124:
'====='
IF DateiGefunden%(Verzeichnis$ + "4\*.*)" = 0 THEN GOTO Programmende
CALL Dateivergleich(Verzeichnis$ + "2\*.HQL", Verzeichnis$ + "4\*.HQL")
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "2\*.LST"
```

```
SHELL Befehl$
CALL Pause
Befehl$ = "DIR /P " + Verzeichnis$ + "4\*.LST"
SHELL Befehl$
CALL Pause

'====='
Fall134:
'====='
    IF DateiGefunden%(Verzeichnis$ + "4\*.") = 0 THEN GOTO Programmende
    CALL Dateivergleich(Verzeichnis$ + "3\*.HQL", Verzeichnis$ + "4\*.HQL")
    CALL Pause
    Befehl$ = "DIR /P " + Verzeichnis$ + "3\*.LST"
    SHELL Befehl$
    CALL Pause
    Befehl$ = "DIR /P " + Verzeichnis$ + "4\*.LST"
    SHELL Befehl$
    CALL Pause
    GOTO Programmende

'=====
Programmende:
'=====
SYSTEM
_____ ENDE DES HAUPTTEILS _____

FEHLERBEHANDLUNG

=====
FehlerBehandlung:
=====
PRINT "ERR = "; ERR; " , ERL = "; ERL
STOP
ON ERROR GOTO 0
GOTO Programmende
_____ ENDE DER FEHLERBEHANDLUNG _____

SUBROUTINEN UND FUNKTIONEN

=====
SUB Ausgabe (Farbe%, Frage$, Puffer$, Offset%, Art$, Bereich%, Bereiche%, y%)
=====
    Gibt `Frage$` gefolgt von `Puffer$` in Zeile `y` aus.
    ,
    Bedeutung der weiteren übergebenen Parameter:
    `Farbe%`      Wahlmodus 1 (normal) oder 2 (hervorgehoben)
    `Offset%`     Verschiebungsmöglichkeit innerhalb einer Spalte
    `Bereich%`    Gewünschte-Spalte
    `Bereiche%`   Spaltenzahl insgesamt
    `Art$`        "L" (linksbündig), "M" (mittenzentriert), "R" (rechtsbündig)
```

```

' Bearbeitung:
' 7. 8.2001 - 4. 9.2001: Norbert Südland und Eckhard Walter, Adelshofen
'-----'
DIM Laenge% 'AS INTEGER
DIM x%

SELECT CASE Farbe%
CASE 1
COLOR 7, 0
CASE 2
COLOR 15, 0
END SELECT
Laenge% = LEN(Frage$) + LEN(Puffer$) + 2
IF LEN(Puffer$) = 0 THEN
Laenge% = Laenge% - 2
END IF
IF Laenge% > 80 / Bereiche% THEN
Laenge% = 80 / Bereiche%
IF LEN(Frage$) >= Laenge% THEN 'Programmier-Ungereimtheit!
Frage$ = LEFT$(Frage$, Laenge%)
Puffer$ = ""
ELSE
Pause
Puffer$ = LEFT$(Puffer$, LEN(Frage$) - Laenge%)
END IF
END IF

SELECT CASE Art$
CASE "L"
x% = INT(Offset% + (Bereich% - 1) * 80 / Bereiche%)
CASE "M"
x% = INT(Offset% + (Bereich% - .5) * 80 / Bereiche% - Laenge% / 2) + 1
CASE "R"
x% = INT(Offset% + Bereich% * 80 / Bereiche% - Laenge%)
END SELECT
LOCATE y%, x%
PRINT Frage$;

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

COLOR 7, 0
END SUB 'Ausgabe _____'

'=====
FUNCTION DateiGefunden% (Pfad$)

```

```
'=====
' Prüft nach, ob 'Pfad$' zu finden ist und gibt dann 1 zurück, sonst 0.
'
' Bearbeitung:
' 17.10.2024 - 31. 5.2025 Norbert Südland, Aalen
'-----
DIM Befehl$
DIM Zeile$
DIM Datei%
DIM Ergebnis%

Ergebnis% = 0
Befehl$ = "DIR " + Pfad$ + " > Gefunden.Ja"
101 SHELL Befehl$
Datei% = FREEFILE
102 OPEN "Gefunden.Ja" FOR BINARY ACCESS READ AS #Datei%
    WHILE Ergebnis% = 0 AND EOF(Datei%) = 0
        CALL LINEINPUT(Datei%, Zeile$)
        IF MID$(Zeile$, 9, 4) = " HQL" THEN
            Ergebnis% = 1
        END IF
    WEND
CLOSE #Datei%
103 KILL "Gefunden.Ja"

104 DateiGefunden% = Ergebnis%
END FUNCTION 'DateiGefunden% _____

'=====
SUB Dateivergleich (Argument1$, Argument2$)
'=====
' Ersetzt das FC von DOS, weil dies nicht mehr überall verfügbar ist.
'
' Bearbeitung: 21. 9.2024 - 25. 6.2025 Norbert Südland, Aalen
'-----
CONST Inhalt1$ = "Inhalt_1.$$$"
CONST Inhalt2$ = "Inhalt_2.$$$"

DIM Ausgabetext$
DIM Befehl$
DIM Datensatz1$, Datensatz2$
DIM Pfad$, Pfad1$, Pfad2$
DIM Zeile$, Zeile1$, Zeile2$

DIM Datei1%, Datei2%, Datei3%, Datei4%
DIM Gefunden%
DIM Laenge%
DIM Seiten%
DIM z%
DIM Zeilen%

DIM P1&, P2&

'Zu vergleichende Dateien auflisten:
'-----
Seiten% = 0
```

```
Zeilen% = 0
Gefunden% = 0
Pfad1$ = LEFT$(Argument1$, LASTINSTR$(Argument1$, "\"))
Befehl$ = "DIR " + Argument1$ + " > " + Inhalt1$
200 SHELL Befehl$
Pfad2$ = LEFT$(Argument2$, LASTINSTR$(Argument2$, "\"))
Befehl$ = "DIR " + Argument2$ + " > " + Inhalt2$
201 SHELL Befehl$

'Dateilisten abarbeiten:
'-----'
Datei1% = FREEFILE
210 OPEN Inhalt1$ FOR BINARY ACCESS READ AS #Datei1%
DO
    CALL LINEINPUT(Datei1%, Zeile1$)
    LOOP UNTIL INSTR(2, Zeile1$, ":\") > 0
    Datei2% = FREEFILE
211 OPEN Inhalt2$ FOR BINARY ACCESS READ AS #Datei2%
DO
    CALL LINEINPUT(Datei2%, Zeile2$)
    LOOP UNTIL INSTR(2, Zeile2$, ":\") > 0
DO
    CALL LINEINPUT(Datei1%, Zeile1$)
    IF Zeile1$ = "" THEN
        CALL LINEINPUT(Datei1%, Zeile1$)
    END IF
    IF MID$(Zeile1$, 9, 1) = SPACE$(1) THEN
        Zeile$ = RTRIM$(LEFT$(Zeile1$, 8)) + "."
        Zeile$ = Zeile$ + RTRIM$(MID$(Zeile1$, 10, 3))
        Zeile1$ = Zeile$
    ELSE
        DO
            CALL LINEINPUT(Datei1%, Zeile1$)
            LOOP UNTIL EOF(Datei1%)
        END IF
    CALL LINEINPUT(Datei2%, Zeile2$)
    IF Zeile2$ = "" THEN
        CALL LINEINPUT(Datei2%, Zeile2$)
    END IF
    IF MID$(Zeile2$, 9, 1) = SPACE$(1) THEN
        Zeile$ = RTRIM$(LEFT$(Zeile2$, 8)) + "."
        Zeile$ = Zeile$ + RTRIM$(MID$(Zeile2$, 10, 3))
        Zeile2$ = Zeile$
    ELSE
        DO
            CALL LINEINPUT(Datei2%, Zeile2$)
            LOOP UNTIL EOF(Datei2%)
        END IF
    IF Pfad1$ <> Pfad2$ THEN
        WHILE NOT EOF(Datei1%) AND NOT EOF(Datei2%) AND Zeile1$ < Zeile2$
            CALL LINEINPUT(Datei1%, Zeile1$)
        WEND
        WHILE NOT EOF(Datei1%) AND NOT EOF(Datei2%) AND Zeile1$ > Zeile2$
220     CALL LINEINPUT(Datei2%, Zeile2$)
        WEND
        IF Zeile1$ <> Zeile2$ GOTO Schleifenende
    END IF
```

```
IF NOT EOF(Datei1%) AND NOT EOF(Datei2%) THEN
  Ausgabertext$ = "Vergleichen der Dateien " + Pfad1$ + Zeile1$
  Ausgabertext$ = Ausgabertext$ + " und " + Pfad2$ + Zeile2$
  Zeilen% = Zeilen% + INT(LEN(Ausgabertext$) / 80) + 1
  IF INT((Zeilen% - 1) / 24) > Seiten% THEN
    CALL Pause
    Seiten% = Seiten% + 1
  END IF
  PRINT Ausgabertext$
  Datei3% = FREEFILE
  Pfad$ = Pfad1$ + Zeile1$
230  OPEN Pfad$ FOR BINARY ACCESS READ SHARED AS #Datei3% LEN = 512
    Datei4% = FREEFILE
    Pfad$ = Pfad2$ + Zeile2$
240  OPEN Pfad$ FOR BINARY ACCESS READ SHARED AS #Datei4% LEN = 512
    Datensatz1$ = SPACE$(512)
    Datensatz2$ = SPACE$(512)
    P1% = 1
    P2% = 1
    WHILE NOT EOF(Datei3%) AND NOT EOF(Datei4%) AND Gefunden% = 0
250     GET #Datei3%, P1%, Datensatz1$
260     GET #Datei4%, P1%, Datensatz2$
    IF Datensatz1$ = Datensatz2$ THEN
      P1% = P1% + 512
      P2% = P2% + 512
    ELSE
      PRINT "FC: Die Dateien sind verschieden:"
      Zeilen% = Zeilen% + 1
      IF INT((Zeilen% - 1) / 24) > Seiten% THEN
        CALL Pause
        Seiten% = Seiten% + 1
      END IF
      Laenge% = FNMIN(LEN(Datensatz1$), LEN(Datensatz2$))
      FOR z% = 1 TO Laenge%
        IF MID$(Datensatz1$, z%, 1) <> MID$(Datensatz2$, z%, 1) THEN
          Gefunden% = z%
          z% = Laenge%
        END IF
      NEXT
      Laenge% = FNMIN(Laenge% - Gefunden% + 1, 79)
      COLOR 0, 7
      PRINT MID$(Datensatz1$, Gefunden%, Laenge%);
      COLOR 7, 0
      PRINT
      Zeilen% = Zeilen% + 1
      IF INT((Zeilen% - 1) / 24) > Seiten% THEN
        CALL Pause
        Seiten% = Seiten% + 1
      END IF
      PRINT " und"
      Zeilen% = Zeilen% + 1
      IF INT((Zeilen% - 1) / 24) > Seiten% THEN
        CALL Pause
        Seiten% = Seiten% + 1
      END IF
      COLOR 0, 7
      PRINT MID$(Datensatz2$, Gefunden%, Laenge%);
      COLOR 7, 0
```



```

        PRINT
        Zeilen% = Zeilen% + 1
        IF INT((Zeilen% - 1) / 24) > Seiten% THEN
            CALL Pause
            Seiten% = Seiten% + 1
        END IF
    END IF
WEND
IF Gefunden% = 0 THEN
    Ausgabetext$ = "FC: Keine Unterschiede gefunden"
    Zeilen% = Zeilen% + 1
    IF INT((Zeilen% - 1) / 24) > Seiten% THEN
        CALL Pause
        Seiten% = Seiten% + 1
    END IF
    PRINT Ausgabetext$
END IF
CLOSE Datei4%
CLOSE Datei3%
END IF
'-----'
Schleifenende:
'-----'
        LOOP WHILE NOT EOF(Datei2%) AND NOT EOF(Datei1%)
        CLOSE Datei2%
        CLOSE Datei1%

        'Aufräumen:
        '-----'
300 KILL Inhalt1$
310 KILL Inhalt2$

END SUB 'Dateivergleich _____'

'=====
FUNCTION LASTINSTR% (Text$, Suchausdruck$)
'=====
' Ermittelt das letzte Auftreten von 'Suchausdruck$' in 'Text$'.
'
' Bearbeitung:
' 21. 9.2004 Norbert Südland, D-73431 Aalen
'-----
DIM Beginn%
DIM LetzterBeginn%

Beginn% = 0
DO
    LetzterBeginn% = Beginn%
    Beginn% = INSTR(Beginn% + 1, Text$, Suchausdruck$)
LOOP WHILE Beginn% > 0
LASTINSTR = LetzterBeginn%

END FUNCTION 'LASTINSTR% _____'

'=====
SUB LINEINPUT (Datei%, Zeile$)
'=====
' Ersetzt den Befehl LINE INPUT, um auch unter der DOS-Box 0.73 zu laufen.

```

```

' Diese DOS-Box erzeugt mit DIR eine Ausgabe wie unter Unix und kann mit
' LINEINPUT statt mit LINE INPUT ausgewertet werden.
' Die 'Datei%' muss dazu als BINARY ACCESS READ (WRITE) geöffnet sein.
' Am Ende wird eine Zeile mit 512 Zeichen zurück gegeben, damit jede Zeile,
' die nicht mit CHR$(10) endet, noch erfasst wird. Dieser Kompromiss sorgt
' für Geschwindigkeit und reicht hier für das Auswerten von DIR aus.
'
' Bearbeitung:
' 11. 8.2009: Norbert Südland, D-73431 Aalen
' Überprüfung:
' 13. 8.2009: Norbert Südland, D-73431 Aalen
' -----
DIM AltePosition&
DIM Laenge&
DIM NeuePosition&

Zeile$ = SPACE$(512)
AltePosition& = SEEK(Datei%)
GET #Datei%, AltePosition&, Zeile$
Laenge& = INSTR(Zeile$, CHR$(10))
IF Laenge& <> 0 THEN
    Zeile$ = LEFT$(Zeile$, Laenge& - 1)
    IF INSTR(RIGHT$(Zeile$, 1), CHR$(13)) <> 0 THEN
        Zeile$ = LEFT$(Zeile$, LEN(Zeile$) - 1)
    END IF
END IF

IF EOF(Datei%) = 0 OR Laenge& <> 0 THEN
    NeuePosition& = AltePosition& + Laenge&
    SEEK #Datei%, NeuePosition&
END IF
'PRINT LEN(Zeile$), Zeile$
'Pause
END SUB 'LINEINPUT _____

'=====
SUB Pause
'=====
' Blendet in Zeile 25 eine Ansage ein und wartet auf einen Tastendruck.
'
' Bearbeitung:
' 18. 8.2001 Norbert Südland und Eckhard Walter, Adelshofen
'
' Überprüfung:
' 18. 8.2001 Norbert Südland und Eckhard Walter, Adelshofen
' -----
DIM x%      'AS INTEGER
DIM y%      'AS INTEGER
DIM Antwort$ 'AS STRING

'Aktuelle Cursor-Position ermitteln:
'-----
x% = CSRLIN
y% = POS(0)

'Zeile 25 löschen und beschriften:
'-----

```

```

LOCATE 25, 1, 0
PRINT SPACE$(80);
Ausgabe 1, "", "Weiter mit Tastendruck", 0, "M", 1, 1, 25

'Tastendruck abwarten:
'-----'
Antwort$ = Taste$

'Zeile 25 erneut löschen:
'-----'
LOCATE 25, 1, 1
PRINT SPACE$(80);

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

'=====
FUNCTION STRLEN% (Text$, EndeZeichen$)
'=====
' Ermittelt die Stringlänge von `Text$` bis zum ersten Auftreten aller
' `EndeZeichen$`, so wie es in der Programmiersprache C üblich ist.
' Kommt `Ende` nicht vor, wird die Stringlänge `LEN(Text$)` zurückgegeben.
'
' Bearbeitung:
' 4. 8.2001: Norbert Südland, München
' Überprüfung:
' 4. 8.2001: Norbert Südland, München
'-----
DIM Zwischenergebnis% 'AS INTEGER

Zwischenergebnis% = INSTR(Text$, EndeZeichen$)
IF Zwischenergebnis% = 0 THEN
    Zwischenergebnis% = LEN(Text$)
ELSE
    Zwischenergebnis% = Zwischenergebnis% - 1
END IF

STRLEN% = Zwischenergebnis%
END FUNCTION 'STRLEN% _____

'=====
FUNCTION Taste$
'=====
' Wartet auf einen Tastendruck und liefert das entsprechende ASCII-Zeichen.
'
' Bearbeitung:
' 18. 8.2001: Norbert Südland und Eckhard Walter, Adelshofen
' Überprüfung:
' 18. 8.2001: Norbert Südland und Eckhard Walter, Adelshofen
' 6. 9.2002: Norbert Südland, Aalen
'-----
DIM Antwort$ 'AS STRING

'Tastaturpuffer löschen:

```

```
'-----'
WHILE INKEY$ <> ""
WEND

'Tastatur erneut abfragen, bis ein Zeichen geschrieben wurde:
'-----'
Antwort$ = ""
WHILE LEN(Antwort$) = 0
  Antwort$ = INKEY$
WEND

'Ergebnis:
'-----'
Taste$ = Antwort$
END FUNCTION 'Taste$ _____'
```