

COMPUTE.BAS	0.45	4/14/2025
-------------	------	-----------

# DESCRIPTION :

- Determining the Numerical Relations of Historical Data by 4 Methods

# REFERENCES :

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

# HANDLING :

11/12/1994 - 12/ 2/1994	Norbert Südland
2/12/2000 - 3/28/2003	Norbert Südland
4/ 5/2007 - 3/22/2017	Norbert Südland
10/21/2022 - 4/14/2025	Norbert Südland

# PREPARATION:

# First Commands:

```
'OPTION EXPLICIT          'This is useful with Visual Basic!
OPTION BASE 1              'Arrays start by index No.`1`
COMMON HistoricChoice%, WorkingPlace$, WorkingTime$, CountingMode%
'The command ``$DYNAMIC` does not always work reliably!
```

# Constants:

```
CONST RIGHT = -1           'pre-defined by BASIC
CONST WRONG = 0            'pre-defined by BASIC
CONST PreDatingEarliest$ = "I" 'incomplete pre-dating >=
CONST PreDatingComplete$ = "II" 'complete pre-dating
CONST PreDatingLatest$ = "III" 'incomplete pre-dating <=
CONST ConfigurationFile$ = "COMPUTE.CFG"
```

```
' |-----|
' |               Announce Subroutines:               |
' |-----|
```

```
' |-----|
' |               Basic Commands:                       |
' |-----|
```

```
DECLARE SUB Pause ()
```

```
DECLARE FUNCTION YesNoQuestion$ (Question$, Message$)
```

```
' |-----|
' |               Change Data:                           |
' |-----|
```

```
DECLARE SUB Change (Data$, Variable$, Contents$)
```

```
DECLARE SUB NewStatus (DateValue$)
```

```
DECLARE SUB ProtocolMessage (MessageText$, NewLine%)
```

```
DECLARE SUB SaveDate (Position&, Data$)
```

```
DECLARE SUB SortInto (Name$, Position&, NameType%)
```

```
DECLARE SUB WriteTo (File%, FL&, DateValue$, T%, Kind%, s%, Place&, Num&)
```

```
DECLARE FUNCTION ToFile$ (Symbol$)
```

```
DECLARE FUNCTION ToUser$ (Symbol$)
```

```
' |-----|
' |               Conversions:                           |
' |-----|
```

```
DECLARE SUB Add (Summand1$, Kind$, Direction$, Def$, ONum1%, ONum2%, Nxt$)
```

```
DECLARE SUB RestSystem (Sign%, Year%, Month%, Day%)
```

```
DECLARE SUB Reverse (s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, Direction$)
```

```
DECLARE SUB Shift (RelationFile%(), RL&(), Source%, Target%)
```

```
DECLARE SUB TermToNumber (Term$, Sign%, Year%, Month%, Day%)
```

```
DECLARE SUB TimeShift (s1%, y1%, m1%, d1%, Status$, s2%, y2%, m2%, d2%, d$)
```

```
DECLARE SUB TimeToNumber (Instant$, Year%, Month%, Day%)
```

```
DECLARE FUNCTION Intersection$ (DateValue1$, DateValue2$)
```

```
DECLARE FUNCTION NumberToTerm$ (Sign%, Year%, Month%, Day%)
```

```
DECLARE FUNCTION NumberToTime$ (Year%, Month%, Day%, Mode%)
```

```
DECLARE FUNCTION SequenceOrder& (Mode%, ListLength&, ListElement&)
```

```
' |-----|
' |               Generate Time Table:                   |
' |-----|
```

```
DECLARE SUB Check (Data$)
```

```
DECLARE SUB Connection (Num&)
```

```
DECLARE SUB CopyDate (Num&)
```

```
DECLARE SUB Correction (Array$, Rest%, RL&)
```

```
DECLARE SUB Evaluate (Rest%, RL&, Num&, What%, CountingMode%)
```

```
DECLARE SUB Optimize ()
```

```
DECLARE SUB RelationsBackwards (Num&)
```

```

DECLARE SUB RelationsForward (Num&)
DECLARE SUB Result (File%, FL&, Def$, Comparison$, Target%, Source%, Num&)

DECLARE FUNCTION StartCalculation$ (WorkingPlace$, New%, Rest%, RL&, CM%)
DECLARE FUNCTION Contradiction% (DateValue$)
DECLARE FUNCTION Total$ (WorkingTime$, Tol$, DateValue$, Direction$)

```

```

'-----'
'|                                     |
'|                               Represent Data:                               |
'|                                     |
'-----'

```

```

DECLARE FUNCTION Find& (Name$, Place&, NameType%)
DECLARE FUNCTION Load$ (File%, LengthOfDataSet%, FileLength&, Position&)
DECLARE FUNCTION Moment% (SymbolText$)
DECLARE FUNCTION NameRegister$ (Position&)
DECLARE FUNCTION OrdinaryNumber$ (Num%)
DECLARE FUNCTION Part$ (Data$, Variable$)
DECLARE FUNCTION ReadIn& (Num&, Data$)
DECLARE FUNCTION SimultaneousnessCheck$ (Text$)
DECLARE FUNCTION Symbol$ (MomentNumber%)

DECLARE FUNCTION QuickPosition% (List$(), Name$)
DECLARE FUNCTION SIZEOF% (StructureName$)
DECLARE FUNCTION STRLEN% (Text$, EndCharacter$)

```

```

'|-----'
'|                                     |
'|                               Global Variables:                               |
'|                                     |
'|-----'

```

```

DIM SHARED NameFile$(5)                'AS STRING
DIM SHARED RelationFileName$(6)
DIM SHARED ProtocolFile$
DIM SHARED ArrayLength%                'AS INTEGER
DIM SHARED DataLength%
DIM SHARED InputFile$
DIM SHARED GlobVarNumber%
DIM SHARED Names$(5)
DIM SHARED RelationFile$(6)
DIM SHARED GIL&                        'AS LONG
DIM SHARED NL&(5)
DIM SHARED RL&(6)

```

```

'|-----'
'|                                     |
'|                               Local Variables:                               |
'|                                     |
'|-----'

```

```

DIM Array$                'AS STRING
DIM Buffer$
DIM Data$
DIM InputFileName$
DIM List$
DIM ListFile$
DIM ReadyFile$
DIM Name$
DIM PositionFile$

```

```

DIM Program$
DIM Remainder$
DIM Simultaneous$
DIM Text$
DIM Variable$
DIM Interrupt$
DIM Question$
DIM Compared$
DIM FileCopy$
DIM c%           'AS INTEGER
DIM Configuration%
DIM Found%
DIM GoOn%
DIM L%
DIM New%
DIM Position%
DIM ReadingTrial%
DIM Rest%
DIM Ready%
DIM Protocol%
DIM CM%
DIM Copy%
DIM LB%           'List Begin AS LONG
DIM n%
DIM Num%
DIM Place%
DIM RL%           'Remainder List Length
DIM RLStart%

```

MAIN PART
-----------

```
'Preparation:
```

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

```
CLEAR , , 4096
```

```
ON ERROR GOTO ErrorHandler
```

```
GOSUB StartProgram
```

```
CM% = CountingMode%
```

```
Interrupt$ = StartCalculation$(WorkingPlace$, New%, Rest%, RL%, CM%)
```

```
IF Interrupt$ = "Y" THEN GOTO EndOfProgram
```

```
'Evaluate Relation Files:
```

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

```
L% = SIZEOF%("remainder")
```

```
IF LB% = 0 THEN
```

```
    LB% = 1
```

```
END IF
```

```
RLStart% = RL%
```

```
WHILE LB% <= RL%
```

```
    'Preparation:
```

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

```
    Remainder$ = SPACE$(L%)
```

```
    GET #Rest%, L% * LB% + 1, Remainder$
```

```
    PUT #Ready%, L% * LB% + 1, Remainder$
```

```

IF VAL(Part$(Remainder$, "remainder.position")) <> LB& THEN
    Text$ = "The calculation table has been mixed up!"
    ProtocolMessage Text$, 1
    Pause
    'Table has been Mixed Up!
    GOTO EndOfProgram
END IF
Num& = VAL(Part$(Remainder$, "remainder.number"))
c% = Moment%(Part$(Remainder$, "remainder.moment"))
ProtocolMessage "", 1
Text$ = LTRIM$(STR$(FRE(0))) + " Bytes yet free."
ProtocolMessage Text$, 1

'Interrupt Calculation?
'-----'
Question$ = "Interrupt calculation?"
Text$ = "The calculation was interrupted."
Interrupt$ = YesNoQuestion(Question$, Text$)
IF Interrupt$ = "Y" THEN GOTO EndOfProgram

'Begin of Next Data Record:
'-----'
Text$ = "(" + LTRIM$(STR$(LB&))
Text$ = Text$ + RIGHT$(OrdinaryNumber$(LB& MOD 100), 2)
Text$ = Text$ + "/" + LTRIM$(STR$(RL&)) + ") position: "
ProtocolMessage Text$, 0
n& = ReadIn&(Num&, Data$)
Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
Text$ = Part$(Data$, Variable$) + " " + ToUser$(Symbol$(c%))
Text$ = Text$ + " " + NameRegister$(Num&) + STR$(Num&)
ProtocolMessage Text$, 1

'Determine Connection:
'-----'
Connection Num&

'Consider Simultaneous Moments when Presenting:
'-----'
Simultaneous$ = ToUser$(Part$(Data$, "data.simultaneous"))
Found% = INSTR(Simultaneous$, Symbol$(c%))
IF Found% > 0 THEN
    IF Found% > 3 THEN
        c% = Moment%(MID$(Simultaneous$, 4, 1))
    ELSE
        c% = Moment%(MID$(Simultaneous$, 1, 1))
    END IF
END IF
Array$ = SPACE$(ArrayLength%)
FOR Place& = 0 TO RL&(c%)
    GET #RelationFile$(c%), Place& * ArrayLength% + 1, Array$
    Text$ = LEFT$(Array$, ArrayLength% - 2)
    ProtocolMessage Text$, 1
NEXT Place&
IF LB& <= 0 THEN 'Debugging Possibility
    Pause

'Enable Check of the Calculations:
'-----'
FOR c% = 1 TO 6

```

```

        CLOSE #RelationFile%(c%)
    NEXT c%
    '-----'
    'Here Other Editors may get Access to the Results:
    '-----'
    'SHELL "C:\BIBEL\UED.EXE " + WorkingPlace$ + "*.RL*"
    'FOR c% = 1 TO 6
    '    Buffer$ = RelationFileName$(c%)
    '    OPEN Buffer$ FOR BINARY ACCESS READ WRITE AS #RelationFile%(c%)
    'NEXT c%
    'Pause
    'c% = Moment%(Part$(Remainder$, "remainder.moment"))
END IF

'Feedback Check to the Yielded Dates:
'-----'
Evaluate Rest%, RL%, Num%, c%, CountingMode%
IF LB% <= 0 THEN
    Pause
END IF

'Reading Pauses in Second's Measure:
'-----'
'GoOn% = 0
'TIMER ON
'ON TIMER(1) GOSUB GoOnInterrupt
'WHILE GoOn% = 0
'WEND
'TIMER OFF
'Pause

LB% = LB% + 1
WEND

'Denounce Eventually a Defective Dating:
'-----'
IF New% = 1 AND RL% = RLStart% THEN
    Text$ = "All pre-datings are without connection."
    ProtocolMessage Text$, 1
END IF

'=====
EndOfProgram:
'=====
'Present Eventually the Error Number:
'-----'
IF ERR > 0 THEN
    Text$ = "Error No." + STR$(ERR) + " occurred in line" + STR$(ERL) + "."
1    ProtocolMessage Text$, 1
    Pause
END IF

'Eventually Finish Calculation:
'-----'
IF Interrupt$ <> "Y" AND Rest% <> 0 AND Ready% <> 0 THEN
    L% = SIZEOF("remainder")
    Remainder$ = STRING$(L%, "-")
    Change Remainder$, "remainder.end", "|" + CHR$(13) + CHR$(10)

```

```

    PUT #Rest%, L% * (RL& + 1) + 1, Remainder$
    PUT #Ready%, L% * (RL& + 1) + 1, Remainder$
    CLOSE #Ready%
    CLOSE #Rest%
    ProtocolMessage "", 1
    ProtocolMessage "The calculation has been finished successfully.", 1

    CM% = CountingMode%
    CountingMode% = (CountingMode% MOD 4) + 1
    Text$ = "New calculation method: counting mode number"
    Text$ = Text$ + STR$(CountingMode%) + "."
    ProtocolMessage Text$, 1
END IF

'Present Calculation Time:
'-----'
ProtocolMessage "", 1
WorkingTime$ = "Calculation begin: " + WorkingTime$ + CHR$(13) + CHR$(10)
Text$ = LEFT$(WorkingTime$, LEN(WorkingTime$) - 2)
ProtocolMessage Text$, 1
c% = LEN(WorkingTime$)
WorkingTime$ = "Calculation end:   " + DATE$ + SPACE$(2) + TIME$
ProtocolMessage WorkingTime$, 1
BEEP: BEEP: BEEP

'Close all Files:
'-----'
CLOSE
FOR c% = 1 TO 6
    IF LEN(RelationFileName$(c%)) > 0 THEN
        KILL RelationFileName$(c%)
    END IF
NEXT c%
FOR c% = 1 TO 5
    IF LEN(RelationFileName$(c%)) > 0 THEN
        KILL NameFile$(c%)
    END IF
NEXT c%

'Free Data Space:
'-----'
REDIM SHARED GVBEGIN%(1)
REDIM SHARED GVLENGTH%(1)
REDIM SHARED GVNAME$(1)
REDIM SHARED GVTYPE$(1)
PositionFile$ = ""
List$ = ""
Remainder$ = ""
Data$ = ""
Variable$ = ""
Array$ = ""
Simultaneous$ = ""
Question$ = ""
Compared$ = ""

'State of the Handling:
'-----'

```

```
Text$ = LTRIM$(STR$(RL&)) + " records in list file "
ProtocolMessage Text$, 1
ProtocolMessage ListFile$, 1
Text$ = LTRIM$(STR$(FRE(0))) + " Bytes yet free."
ProtocolMessage Text$, 1
```

```
'Eventually Copy the Result:
'-----'
```

```
IF Interrupt$ <> "Y" AND New% = 1 THEN
  ProtocolMessage "", 1
```

```
2  MKDIR WorkingPlace$ + LTRIM$(STR$(CM%))
3  ProtocolMessage "The result file are copied to:", 1
  ProtocolMessage WorkingPlace$ + LTRIM$(STR$(CM%)) + "\*.*", 1
```

```
ProtocolMessage InputFileName$, 1
InputFile% = FREEFILE
OPEN InputFileName$ FOR INPUT AS #InputFile%
  FileCopy$ = WorkingPlace$ + LTRIM$(STR$(CM%)) + "\" + Name$ + "HQL"
  GOSUB CopyFile
CLOSE #InputFile%
```

```
ProtocolMessage ListFile$, 1
InputFile% = FREEFILE
OPEN ListFile$ FOR INPUT AS #InputFile%
  FileCopy$ = WorkingPlace$ + LTRIM$(STR$(CM%)) + "\" + Name$ + "LST"
  GOSUB CopyFile
CLOSE #InputFile%
```

```
ProtocolMessage ReadyFile$, 1
InputFile% = FREEFILE
OPEN ReadyFile$ FOR INPUT AS #InputFile%
  FileCopy$ = WorkingPlace$ + LTRIM$(STR$(CM%)) + "\" + Name$ + "RDY"
  GOSUB CopyFile
CLOSE #InputFile%
```

```
ProtocolMessage ProtocolFile$, 1
InputFile% = FREEFILE
OPEN ProtocolFile$ FOR INPUT AS #InputFile%
  FileCopy$ = WorkingPlace$ + LTRIM$(STR$(CM%)) + "\" + Name$ + "LOG"
  GOSUB CopyFile
CLOSE #InputFile%
```

```
END IF
```

```
'Delete Remaining Arrays:
'-----'
```

```
Interrupt$ = ""
InputFileName$ = ""
ListFile$ = ""
ReadyFile$ = ""
ProtocolFile$ = ""
Text$ = ""
Name$ = ""
FileCopy$ = ""
Buffer$ = ""
```

```
'Pause before Program Chaining:
```

```

' -----'
Pause
IF Program$ <> "" THEN CHAIN Program$
SYSTEM
' _____ END OF THE MAIN PART _____ '

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

'====='
ErrorHandling:
'====='
SELECT CASE ERR
CASE 5 'Illegal Function Call:
Pause 'VisualBasic is not compatible to memory request!
RESUME NEXT
CASE 52 'Illegal File Name:
SELECT CASE ERL
CASE 501
PRINT "Protocol file name not available."
GOTO EndOfProgram
CASE ELSE
Pause
END SELECT
CASE 53 'File Not Found:
SELECT CASE ERL
CASE 1
RESUME NEXT
CASE 10
PRINT "Configuration file "; CHR$(34); ConfigurationFile$; CHR$(34);
PRINT " not found."
GOTO EndOfProgram
CASE 30
PRINT "Position file "; CHR$(34); "HISTORIC.POS"; CHR$(34); " not found."
ReadingTrial% = ReadingTrial% + 1
RESUME 21
CASE 60, 65, 66 'Undeletable File:
RESUME NEXT
CASE 70
RESUME 80
CASE 71
RESUME 73
CASE 80
RESUME NEXT
CASE ELSE
RESUME NEXT
END SELECT
CASE 62 'Input Past End of File:
SELECT CASE ERL
CASE 20
PRINT "Configuration file "; CHR$(34); ConfigurationFile$; CHR$(34);
PRINT " is empty."
GOTO EndOfProgram
CASE 40
PRINT "Position file "; CHR$(34); "HISTORIC.POS"; CHR$(34); " is empty."
GOTO EndOfProgram

```

```
CASE 50
  PRINT "Position file "; CHR$(34); "HISTORIC.POS"; CHR$(34);
  PRINT " is incomplete."
  GOTO EndOfProgram
CASE ELSE
  Pause
  RESUME NEXT
END SELECT
CASE 63      'Illegal File Record Number:
  SELECT CASE ERL
  CASE ELSE
    Pause
    RESUME NEXT
  END SELECT
CASE 70      'Access Denied:
  SELECT CASE ERL
  CASE 1      'Writing Tests onto `WorkingPlace$`:
    PRINT "The working place directory " + WorkingPlace$
    PRINT "is write protected. It can be changed via " + CHR$(34);
    PRINT "SET HISTORICTEMP=.." + CHR$(34) + " at the DOS level."
    Pause
    SYSTEM      'Hard Program Exit
  CASE ELSE
    Pause
    RESUME NEXT
  END SELECT
CASE 75      'Path /File Access Error
  SELECT CASE ERL
  CASE 2      'Make Existing Subdirectory
    RESUME NEXT
  CASE 5      'Writing Tests onto `WorkingPlace$`
    RESUME NEXT
  CASE ELSE
    Pause
    RESUME NEXT
  END SELECT
CASE 100
  PRINT "Unexpected data structure in file "; CHR$(34); "HISTORIC.POS;"
  PRINT CHR$(34); " ."
  GOTO EndOfProgram
CASE 101
  Text$ = "Missing pre-dating in data file " + CHR$(34) + InputFileName$
  Text$ = Text$ + CHR$(34) + " ."
  ProtocolMessage Text$, 1
  GOTO EndOfProgram
CASE 102
  Text$ = "Name of file record is missing."
  ProtocolMessage Text$, 1
  GOTO EndOfProgram
CASE 103
  Text$ = "Statements on predecessor are wrong."
  ProtocolMessage Text$, 1
  GOTO EndOfProgram
CASE 104
  Text$ = "Statements on relation name are wrong."
  ProtocolMessage Text$, 1
  GOTO EndOfProgram
CASE 105
```

```
Text$ = "Statements on duration are wrong."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 106
Text$ = "Too large year number. Four digits should be enough."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 107
Text$ = "Source statement on pre-dating is missing."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 108
Text$ = " in the file " + CHR$(34) + InputFileName$ + CHR$(34) + "."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 109
Text$ = " in the file " + CHR$(34) + InputFileName$ + CHR$(34) + "."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 110
Text$ = "Senseless dating with mit chaos of ordinary numbers."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 111
Text$ = "Unknown structure cannot be changed."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 112
GOTO EndOfProgram
CASE 113
Text$ = "Contradiction within a dating."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 114
Text$ = "Contradiction between the pre-datings."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 115
Text$ = "Dating cannot be attached to nor be saved."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 120
Text$ = "Error in function `SortInto`:"
ProtocolMessage Text$, 1
Text$ = "Double record within relieve list."
ProtocolMessage Text$, 1
GOTO EndOfProgram
CASE 190
'Non-positive Length of Structure was Saved:
Text$ = "Error in function `SIZEOF%`:"
ProtocolMessage Text$, 1
Text$ = "Non-positive length of structure was saved."
ProtocolMessage Text$, 1
GOTO EndOfProgram
END SELECT

PRINT "Error No."; ERR; " in line"; ERL
Pause
ON ERROR GOTO 0
```

GOTO EndOfProgram

' \_\_\_\_\_ END OF THE ERROR HANDLING \_\_\_\_\_ '

SUBROUTINES VIA GOSUB
-----------------------

'====='

StartProgram:

'====='

WorkingTime\$ = DATE\$ + SPACE\$(2) + TIME\$  
CLS

' As `WorkingPlace\$` the working directory is used, which is given by  
' %HISTORICTEMP%, %TEMP%, or %TMP%.  
' If (at old DOS versions) not any `WorkingPlace\$` is given, the program  
' tries to write to the data medium that also contains the program.  
' Eventually the program stops, if no `WorkingPlace\$` is writable:  
'-----'

WorkingPlace\$ = ENVIRON\$("HISTORICTEMP")  
IF WorkingPlace\$ = "" THEN  
    WorkingPlace\$ = ENVIRON\$("QBASICTEMP")  
    IF WorkingPlace\$ <> "" THEN WorkingPlace\$ = WorkingPlace\$ + "\"  
5 MKDIR WorkingPlace\$ + "HISTORIC.TMP"  
    WorkingPlace\$ = WorkingPlace\$ + "HISTORIC.TMP\"  
ELSE  
    IF WorkingPlace\$ <> "" THEN WorkingPlace\$ = WorkingPlace\$ + "\"  
END IF

'Check Writability of `WorkingPlace\$`:  
'-----'

BSAVE WorkingPlace\$ + "HISTORIC.CHK", 0, 0  
KILL WorkingPlace\$ + "HISTORIC.CHK"

'Read in Configuration File:  
'-----'

Configuration% = FREEFILE  
10 OPEN WorkingPlace\$ + ConfigurationFile\$ FOR INPUT AS #Configuration%  
20 LINE INPUT #Configuration%, InputFileName\$  
    LINE INPUT #Configuration%, Buffer\$  
    GIL& = VAL(Buffer\$)  
    LINE INPUT #Configuration%, Buffer\$  
    New% = VAL(Buffer\$)  
    LINE INPUT #Configuration%, Buffer\$  
    IF CoutingMode% < 1 OR CountingMode% > 4 THEN  
        CountingMode% = VAL(Buffer\$)  
        IF CountingMode% < 1 OR CountingMode% > 4 THEN  
            CountingMode% = 1  
        END IF  
    END IF  
    LINE INPUT #Configuration%, Program\$  
CLOSE #Configuration%

'Start Calculation Protocol:  
'-----'

L% = STRLEN\$(InputFileName\$, ".HQL") + 1  
Name\$ = LEFT\$(InputFileName\$, L%)

```

ProtocolFile$ = WorkingPlace$ + Name$ + "LOG"
Text$ = LTRIM$(STR$(FRE(0))) + " Bytes free."
Protocol% = FREEFILE
OPEN ProtocolFile$ FOR OUTPUT AS #Protocol%
    PRINT #Protocol%, ProtocolFile$
    PRINT #Protocol%, "Calculation begin: "; WorkingTime$
    PRINT #Protocol%, Text$
    PRINT #Protocol%,
CLOSE #Protocol%

'Read In the Structure Positions:
'-----'
Position% = FREEFILE
PRINT "Read in the structur positions from "; CHR$(34); "HISTORIC.POS";
PRINT CHR$(34); "... "
ReadingTrial% = 1
21 SELECT CASE ReadingTrial%
CASE 1
    PositionFile$ = "HISTORIC.POS"
CASE 2
    PositionFile$ = WorkingPlace$ + PositionFile$
CASE 3
    Configuration% = FREEFILE
    OPEN "HISTORIC.STR" FOR INPUT AS #Configuration%
    CLOSE #Configuration%
22 OPEN WorkingPlace$ + "STRUCT.CFG" FOR OUTPUT AS #Configuration%
    PRINT #Configuration%, "HISTORIC.STR"      'Structure File
    PRINT #Configuration%, PositionFile$      'The Very Position File
    PRINT #Configuration%, "COMPUTE.BAS"      'Rejump Program
    CLOSE #Configuration%
    CHAIN "STRUCT.BAS"
CASE ELSE
    Pause                                'Programming Mistake!
END SELECT
30 OPEN PositionFile$ FOR INPUT AS #Position%
40 LINE INPUT #Position%, Text$
    L% = STRLEN$(Text$, " ")
    GlobVarNumber% = VAL(LEFT$(Text$, L%))
    PRINT LTRIM$(STR$(GlobVarNumber%)); " structure elements"
    REDIM SHARED GVBEGIN$(GlobVarNumber%)      'AS INTEGER
    REDIM SHARED GVLENGTH$(GlobVarNumber%)      'AS INTEGER
    REDIM SHARED GVNAME$(GlobVarNumber%)        'AS STRING
    REDIM SHARED GVTYPE$(GlobVarNumber%)        'AS STRING
    FOR c% = 1 TO GlobVarNumber%
50 INPUT #Position%, GVBEGIN$(c%)
    INPUT #Position%, GVLENGTH$(c%)
    INPUT #Position%, GVNAME$(c%)
    INPUT #Position%, GVTYPE$(c%)
    NEXT c%
CLOSE #Position%
DataLength% = SIZEOF$("data")
IF DataLength% = 0 THEN ERROR 100
ArrayLength% = SIZEOF$("array")

'Open Data File:
'-----'
InputFile% = FREEFILE
InputFileName$ = WorkingPlace$ + Name$ + "HQL"

```

```

OPEN InputFileName$ FOR BINARY ACCESS READ WRITE AS #InputFile%

'Generate Name Lists:
'-----'
L% = SIZEOF%("list")
List$ = SPACE$(L%)
Change List$, "list.end", "␣" + CHR$(10) + CHR$(13)
FOR c% = 1 TO 5      'Name Element No. #
    NameFile$(c%) = WorkingPlace$ + Name$ + "NE" + LTRIM$(STR$(c%))
60    KILL NameFile$(c%)
    Names%(c%) = FREEFILE
61    OPEN NameFile$(c%) FOR BINARY ACCESS READ WRITE AS #Names%(c%)
    SELECT CASE c%
        CASE 1
            Buffer$ = "Name /Event:"
        CASE 2, 3
            Buffer$ = OrdinaryNumber$(c% - 1) + " Predecessor:"
        CASE 4, 5
            Buffer$ = OrdinaryNumber$(c% - 3) + " Relation Name:"
        CASE ELSE
            Pause           'Programming Mistake!
    END SELECT
    Change List$, "list.name", Buffer$ + SPACE$(L%)
    Change List$, "list.number", "Number: "
    PUT #Names%(c%), 1, List$
    NL&(c%) = 0
NEXT c%

'Set Up Relation Files:
'-----'
L% = SIZEOF%("array")
List$ = SPACE$(L%)
Change List$, "array.destination", " D"
Change List$, "array.name", "Knd"
Change List$, "array.source", " S"
Change List$, "array.from", "from:"
Change List$, "array.to", "to:"
Change List$, "array.end", "␣" + CHR$(13) + CHR$(10)
FOR c% = 1 TO 6
    RelationFileName$(c%) = WorkingPlace$ + Name$ + "RL" + LTRIM$(STR$(c%))
    Buffer$ = RelationFileName$(c%)
65    KILL RelationFileName$(c%)
    RelationFile%(c%) = FREEFILE
66    OPEN Buffer$ FOR BINARY ACCESS READ WRITE AS #RelationFile%(c%)
    PUT #RelationFile%(c%), 1, List$
    RL&(c%) = 0
NEXT c%

'Set Up Calculation Plans:
'-----'
ListFile$ = WorkingPlace$ + Name$ + "LST"
ReadyFile$ = WorkingPlace$ + Name$ + "RDY"
L% = SIZEOF%("remainder")
Rest% = FREEFILE
70    OPEN ListFile$ FOR BINARY ACCESS READ AS #Rest%
    Ready% = FREEFILE
71    OPEN ReadyFile$ FOR BINARY ACCESS READ AS #Ready%
    LB& = 0

```

```
    RL& = 0
    Remainder$ = SPACE$(L%)
    Compared$ = SPACE$(L%)
    DO
        GET #Rest%, L% * RL& + 1, Remainder$
        IF LB& = RL& THEN
            GET #Ready%, L% * LB& + 1, Compared$
        END IF
        IF Remainder$ <> STRING$(L%, 0) THEN
            IF Remainder$ = Compared$ THEN
                IF LEFT$(Remainder$, L% - 3) = STRING$(L% - 3, "--") THEN
                    Remainder$ = STRING$(L%, 0)
                    LB& = 0
                    RL& = 0
                ELSE
                    LB& = LB& + 1
                    RL& = RL& + 1
                END IF
            ELSE
                RL& = RL& + 1
            END IF
        END IF
        LOOP UNTIL Remainder$ = STRING$(L%, 0)
72     CLOSE #Ready%
73     CLOSE #Rest%
    IF LB& > 0 THEN
        LB& = LB& - 1
    END IF
    IF RL& > 0 THEN
        RL& = RL& - 1
    END IF
    IF RL& > 0 THEN
        Question$ = "Continue unfinished calculation?"
        Text$ = "Calculation is continued."
        IF YesNoQuestion$(Question$, Text$) = "N" THEN
            LB& = 0
            RL& = 0
        END IF
    END IF

80  IF RL& = 0 THEN
        KILL ListFile$
        KILL ReadyFile$
    END IF
    Rest% = FREEFILE
    OPEN ListFile$ FOR BINARY ACCESS READ WRITE AS #Rest%
    IF RL& = 0 THEN
        Remainder$ = SPACE$(L%)
        Change Remainder$, "remainder.position", "Position:" + SPACE$(L%)
        Change Remainder$, "remainder.number", "Number:" + SPACE$(L%)
        Change Remainder$, "remainder.moment", "M"
        Change Remainder$, "remainder.name", "Name /Event:" + SPACE$(L%)
        Change Remainder$, "remainder.date", "Date:" + SPACE$(L%)
        Change Remainder$, "remainder.end", "|" + CHR$(13) + CHR$(10)
        PUT #Rest%, 1, Remainder$
    END IF

    Ready% = FREEFILE
```

```

    OPEN ReadyFile$ FOR BINARY ACCESS READ WRITE AS #Ready%
    IF RL& = 0 THEN
        PUT #Ready%, 1, Remainder$
    END IF

RETURN 'StartProgram _____'

'=====
GoOnInterrupt:
'=====
' Enables a Reading Pause of 1 Second.
'
' Handling:
'   1/31/2003: Norbert Suedland
' Translation:
'   2/ 5/2008: Norbert Suedland
'-----
GoOn% = 1
RETURN 'GoOnInterrupt _____'

'=====
CopyFile:
'=====
' Copies a file, being opened as #InputFile%, to FileCopy$.
'
' Handling:
'   3/18/2017 Norbert Suedland
' Translation:
'   3/22/2017 Norbert Suedland
'-----

99 Copy% = FREEFILE
OPEN FileCopy$ FOR OUTPUT AS #Copy%
WHILE EOF(InputFile%) = 0
    LINE INPUT #InputFile%, Buffer$
    PRINT #Copy%, Buffer$
WEND
CLOSE #Copy%

RETURN 'CopyFile _____'

```

SUBROUTINES AND FUNCTIONS
---------------------------

```

'=====
SUB Add (Summand1$, Kind$, Direction$, Default$, ONum1%, ONum2%, NextSt$)
'=====
' Will add `Summand1$` and `Default$` together.
'
' `Summand1$`: First time record
' `Kind$`:      "U": Union set
'              ">>": Take all properties of `Default$`
' `Direction$`: "+" : Addition   "-" : Subtraction
' `Default$`:   Second time record, works as default of properties
' `ONum1$`:     <> 0: `Summand1$` is an ordinary number
' `ONum2$`:     <> 0: `Default$` is an ordinary number

```

```

' `NextSt$`:
'
' Handling:
' 6/27/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 2/ 5/2008:                Norbert Südland, Aalen
' -----

' Preparation: '
'=====
DIM Summand2$      'AS STRING
DIM Status1$
DIM Status2$
DIM Status$
DIM Term1$
DIM Term2$
DIM Length%        'AS INTEGER
DIM s0%, s1%, s2%, s3%, s4%, s5%, s6%
DIM y0%, y1%, y2%, y3%, y4%, y5%, y6%
DIM m0%, m1%, m2%, m3%, m4%, m5%, m6%
DIM d0%, d1%, d2%, d3%, d4%, d5%, d6%
DIM Start%
DIM Finish%
DIM c%

Summand2$ = Default$

'Determine Each Current Status:
'-----
NewStatus Summand1$
Status1$ = Part$(Summand1$, "date.status")

NewStatus Summand2$
Status2$ = Part$(Summand2$, "date.status")

'Eventually Not Anything is to be Added:
'-----
Length% = SIZEOF$("date")
IF Summand1$ = SPACE$(Length%) AND Summand2$ = SPACE$(Length%) THEN
    GOTO EndAdding
END IF

'Care for Kind of Addition:
'=====
SELECT CASE Kind$
CASE "U"      'Union Set:
    Status$ = Status1$
    IF Status$ <> Status2$ THEN
        Status$ = " "
    END IF
CASE ">>"    'Summand2$ is Dominant:
    Status$ = Status2$
    IF Summand2$ = SPACE$(Length%) THEN
        Summand1$ = Summand2$
        GOTO EndAdding
    END IF
CASE ELSE    'Not Expected:

```

```

        GOTO EndAdding
END SELECT

'The Very Adding:'
'=====
TermToNumber Part$(Summand1$, "date.minimum"), s3%, y3%, m3%, d3%
TermToNumber Part$(Summand2$, "date.minimum"), s4%, y4%, m4%, d4%
TermToNumber Part$(Summand1$, "date.maximum"), s5%, y5%, m5%, d5%
TermToNumber Part$(Summand2$, "date.maximum"), s6%, y6%, m6%, d6%
IF ONum1% <> 0 THEN
    TimeShift s3%, y3%, m3%, d3%, Status1$, s5%, y5%, m5%, d5%, "-"
END IF
IF ONum2% <> 0 THEN
    TimeShift s4%, y4%, m4%, d4%, Status2$, s6%, y6%, m6%, d6%, "-"
END IF

Reverse s4%, y4%, m4%, d4%, s6%, y6%, m6%, d6%, Direction$

SELECT CASE Status$
CASE ">="
    Start% = 1
    Finish% = 1
CASE "<="
    Start% = 2
    Finish% = 2
CASE SPACE$(1)
    Start% = 1
    Finish% = 2
END SELECT
FOR c% = Start% TO Finish%
    IF c% = 1 THEN
        s1% = s3%: y1% = y3%: m1% = m3%: d1% = d3%
        s2% = s4%: y2% = y4%: m2% = m4%: d2% = d4%
    ELSE
        s1% = s5%: y1% = y5%: m1% = m5%: d1% = d5%
        s2% = s6%: y2% = y6%: m2% = m6%: d2% = d6%
    END IF
    d0% = d1% + d2%
    m0% = m1% + m2%
    y0% = s1% * y1% + s2% * y2%
    s0% = 1
    RestSystem s0%, y0%, m0%, d0%
    IF ABS(y0%) >= 10000 THEN
        c% = Finish%
    ELSE
        IF c% = 1 THEN
            s3% = s0%: y3% = y0%: m3% = m0%: d3% = d0%
        ELSE
            s5% = s0%: y5% = y0%: m5% = m0%: d5% = d0%
        END IF
    END IF
NEXT c%
IF ABS(y0%) >= 10000 THEN GOTO EndAdding
IF (ONum1% <> 0 OR ONum2% <> 0) AND NextSt$ <> "-" THEN
    TimeShift s3%, y3%, m3%, d3%, Status$, s5%, y5%, m5%, d5%, "+"
END IF
SELECT CASE Status$
CASE ">="

```

```

    Term1$ = NumberToTerm$(s3%, y3%, m3%, d3%)
    Term2$ = SPACE$(SIZEOF%("term"))
CASE "<="
    Term1$ = SPACE$(SIZEOF%("term"))
    Term2$ = NumberToTerm$(s5%, y5%, m5%, d5%)
CASE " "
    Term1$ = NumberToTerm$(s3%, y3%, m3%, d3%)
    Term2$ = NumberToTerm$(s5%, y5%, m5%, d5%)
END SELECT
Change Summand1$, "date.minimum", Term1$
Change Summand1$, "date.status", Status$
Change Summand1$, "date.maximum", Term2$

'=====
EndAdding:
'=====
END SUB 'Add _____

'=====
SUB Change (Data$, Variable$, Contents$)
'=====
' Will put `Contents$` into the correct position of `Data$`.
' `length%` is the length of the considered `Variable$` string.
' `p%` is the length until the occurrence of the next point ".".
'
' Handling:
' 6/27/2001 - 2/12/2003: Norbert Südland
' Translation:
' 10/ 6/2007 - 2/ 5/2008: Norbert Südland
'-----
DIM Seek$      'AS STRING
DIM begin%     'AS INTEGER
DIM i%         'index
DIM Length%
DIM np%        'next point position
DIM p%         'point position
DIM Where%

Length% = LEN(Variable$)
p% = STRLEN%(Variable$, ".")
IF p% = Length% THEN
    Seek$ = Variable$           'no structure point available
ELSE
    'at least one structure point available:
    p% = p% + 1
    p% = p% + STRLEN%(MID$(Variable$, p% + 1, Length% - p%), ".")
    IF p% < Length% THEN
        'seconds structure point available:
        Seek$ = LEFT$(Variable$, p%) 'seek for rough structure first!
    ELSE
        Seek$ = Variable$
    END IF
END IF

begin% = 1
DO
    Where% = QuickPosition%(GVName$(), Seek$)
    IF Where% = 0 THEN
        Pause
    
```

```

        ERROR 111
    END IF

    begin% = begin% + GVBEGIN%(Where%) - 1
    IF p% < Length% THEN      'Consider substructure:
        p% = p% + 1
        np% = STRLEN$(MID$(Variable$, p% + 1, Length% - p%), ".")
        Seek$ = GVType$(Where%) + MID$(Variable$, p%, np% + 1)
        p% = p% + np%
    ELSE
        p% = Length% + 1      'quit the loop!
    END IF
    LOOP WHILE p% <= Length%

    i% = LEN(Contents$)
    IF i% < GVLength%(Where%) THEN
        Contents$ = SPACE$(GVLength%(Where%) - i%) + Contents$
    ELSE
        IF i% > GVLength%(Where%) THEN
            Contents$ = LEFT$(Contents$, GVLength%(Where%))
        END IF
    END IF

    MID$(Data$, begin%, GVLength%(Where%)) = Contents$
END SUB 'Change _____

'=====
SUB Check (Data$)
'=====
' Will check, whether the file record 'Data$' is sensible.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001: Norbert Südland, Munich
' Translation:
' 2/ 6/2008:          Norbert Südland, Aalen
'-----

DIM Buffer$
DIM Variable$
DIM Predecessor$
DIM Relation$
DIM c%          'AS INTEGER
DIM Found%
DIM L%
DIM CheckLength%
DIM s%, y%, m%, d%

'Name Available?
'-----
CheckLength% = SIZEOF$("NAME")
Buffer$ = Part$(Data$, "data.name")
IF Buffer$ = SPACE$(CheckLength%) THEN
    Pause
    ERROR 102
END IF

'Check or Correct Predecessor:
'-----

```

```

FOR c% = 1 TO 2
  Variable$ = "data.p[" + LTRIM$(STR$(c%)) + "]"
  Predecessor$ = Part$(Data$, Variable$)
  Buffer$ = Part$(Predecessor$, "predecessor.name")
  IF Buffer$ <> SPACE$(CheckLength%) THEN
    Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.direct"))
    IF Buffer$ <> "Y" AND Buffer$ <> "N" THEN
      Pause
      ERROR 103
    END IF
    Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment1"))
    IF Moment%(Buffer$) = 0 THEN
      Pause
      ERROR 103
    END IF
    Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment2"))
    IF Moment%(Buffer$) = 0 THEN
      Pause
      ERROR 103
    END IF
  ELSE
    'Delete, if no name is mentioned:
    Change Data$, Variable$, SPACE$(SIZEOF%("predecessor"))
  END IF
NEXT c%

'Push Eventually Predecessor Upward:
'-----'
IF Part$(Data$, "data.p[1]") = SPACE$(SIZEOF%("predecessor")) THEN
  Change Data$, "data.p[1]", Part$(Data$, "data.p[2]")
  Change Data$, "data.p[2]", SPACE$(SIZEOF%("predecessor"))
END IF

'Delete or Question for Source Eventually:
'-----'
L% = SIZEOF%("date")
IF Part$(Data$, "data.p[1].name") = SPACE$(CheckLength%) THEN
  Found% = 0
  FOR c% = 1 TO 6
    Buffer$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
    IF STRLEN%(Part$(Data$, Buffer$), " ") < L% THEN
      Found% = 1
      'Pre-dating found
      c% = 6
    END IF
  NEXT c%
  '-----'
  ' Hint: The source must be mentioned in the same data record '
  '       which contains the pre-dating. '
  '-----'
  IF Found% = 0 THEN
    Change Data$, "data.source", SPACE$(CheckLength%)
  ELSE
    IF Part$(Data$, "data.source") = SPACE$(CheckLength%) THEN
      Pause
      ERROR 107
    END IF
  END IF
END IF

```

```

'Check and Correct Relations:
'-----'
FOR c% = 1 TO 2
  Variable$ = "data.r[" + LTRIM$(STR$(c%)) + "]"
  Relation$ = Part$(Data$, Variable$)
  Buffer$ = Part$(Relation$, "relation.name")
  IF Buffer$ <> SPACE$(CheckLength%) THEN
    Buffer$ = ToUser$(Part$(Relation$, "relation.moment1"))
    IF Moment%(Buffer$) = 0 THEN
      Pause
      ERROR 104
    END IF
    Buffer$ = Part$(Relation$, "relation.tol1")
    IF Buffer$ = "±" OR Buffer$ = "=" OR Buffer$ = "<" OR Buffer$ = ">" THEN
      TimeToNumber Part$(Relation$, "relation.date"), y%, m%, d%
      s% = 1
      RestSystem s%, y%, m%, d%          'Standardize Date
      IF s% <> 1 OR (y% = 0 AND m% = 0 AND d% = 0) THEN
        Pause
        ERROR 104
      END IF
      Change Relation$, "relation.date", NumberToTime$(y%, m%, d%, 0)
    ELSE
      Pause
      ERROR 104
    END IF
  ELSE
    'Delete, if no relation name is available:
    Buffer$ = SPACE$(SIZEOF$("relation.moment1"))
    Change Relation$, "relation.moment1", Buffer$
    Change Relation$, "relation.tol1", SPACE$(SIZEOF$("relation.tol1"))
    Change Relation$, "relation.date", SPACE$(SIZEOF$("relation.date"))
  END IF

'Check and Correct Duration:
'-----'
Buffer$ = Part$(Relation$, "relation.tol2")
IF Buffer$ = "±" OR Buffer$ = "=" OR Buffer$ = "<" OR Buffer$ = ">" THEN
  Buffer$ = ToUser$(Part$(Relation$, "relation.moment2"))
  IF Moment%(Buffer$) <> 0 THEN
    TimeToNumber Part$(Relation$, "relation.duration"), y%, m%, d%
    s% = 1
    RestSystem s%, y%, m%, d%          'Standardize duration
    IF s% <> 1 OR (y% = 0 AND m% = 0 AND d% = 0) THEN
      Pause
      ERROR 105
    END IF
    Change Relation$, "relation.duration", NumberToTime$(y%, m%, d%, 0)
  ELSE
    Pause
    ERROR 105
  END IF
ELSE
  'Delete, if no tolerance of duration is available:
  Change Relation$, "relation.tol2", SPACE$(SIZEOF$("relation.tol2"))
  Buffer$ = SPACE$(SIZEOF$("relation.duration"))
  Change Relation$, "relation.duration", Buffer$
END IF

Change Data$, Variable$, Relation$

```

```

NEXT c%

'Push Eventually Relation Upward:
'-----'
L% = SIZEOF%("relation")
IF Part$(Data$, "data.r[1]") = SPACE$(L%) THEN
    Change Data$, "data.r[1]", Part$(Data$, "data.r[2]")
    Change Data$, "data.r[2]", SPACE$(SIZEOF%("relation"))

'Information on Simultaneous Moments:
'-----'
IF Part$(Data$, "data.simultaneous") <> SPACE$(5) THEN

    'Copy Source Information:
    '-----'
    Buffer$ = Part$(Data$, "data.r[1].source")
    Change Data$, "data.r[2].source", Buffer$

    'Delete Eventually Double Source Information:
    '-----'
    IF Part$(Data$, "data.r[1].name") = SPACE$(CheckLength%) THEN
        IF Part$(Data$, "data.r[1].tol2") = " " THEN
            Change Data$, "data.r[1].source", SPACE$(CheckLength%)
        END IF
    END IF
END IF
END IF
END IF

'Simultaneous Moments:
'-----'
Buffer$ = ToUser$(Part$(Data$, "data.simultaneous"))
Buffer$ = SimultaneousnessCheck$(Buffer$)
Change Data$, "data.simultaneous", ToFile$(Buffer$)
END SUB 'Check _____

'=====
SUB Connection (Num&)
'=====
' Will find the connection between several dates.
'
' Handling:
' 6/16/2001 - 12/23/2002:    Norbert Südland
' 10/ 8/2016:              Norbert Südland
' Translation:
' 2/19/2008 - 10/13/2016:   Norbert Südland
'-----'

DIM Array$      'AS STRING
DIM Data$
DIM Name$
DIM c%          'AS INTEGER
DIM L%
DIM First&      'AS LONG
DIM Place&

'Prepare Calculation:
'-----'
First& = ReadIn&(Num&, Data$)

```

```

Array$ = SPACE$(ArrayLength%)
Change Array$, "array.end", "|" + CHR$(13) + CHR$(10)
FOR c% = 1 TO 6
    FOR Place& = 1 TO RL&(c%)
        PUT #RelationFile%(c%), Place& * ArrayLength% + 1, Array$
    NEXT Place&
    RL&(c%) = 0          'New beginning of the file
NEXT c%
CopyDate First&

'Set Up Relations:
'-----'
Name$ = Part$(Data$, "data.name")
Place& = First&
WHILE Place& <> 0
    RelationsForward Place&
    IF Place& = First& THEN
        RelationsBackwards Place&
    END IF
    Place& = Find$(Name$, Place& + 1, 1)
WEND

'Check, whether a Change is Necessary:
'-----'
ProtocolMessage "Optimize dates...", 1
Optimize
END SUB 'Connection _____'

'=====
FUNCTION Contradiction% (DateValue$)
'=====
' Will check, whether 'DateValue$' is a self-contradicting date.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 2/19/2008:                Norbert Südland, Aalen
'-----'

DIM Sum1$                'AS STRING
DIM Sum2$
DIM t1$                  'term 1
DIM t2$                  'term 2
DIM L%                   'AS INTEGER
DIM IntermediateResult%

L% = SIZEOF$("date")
Sum1$ = SPACE$(L%)
Sum2$ = Sum1$
L% = SIZEOF$("term")
t1$ = Part$(DateValue$, "date.minimum")
t2$ = Part$(DateValue$, "date.maximum")
IF t1$ <> SPACE$(L%) AND t2$ <> SPACE$(L%) THEN
    Change Sum1$, "date.minimum", t1$
    NewStatus Sum1$
    Change Sum2$, "date.minimum", t2$
    NewStatus Sum2$
    Add Sum2$, "U", "-", Sum1$, 1, 1, "-"          'Difference in Y M D

```

```

    IF Part$(Sum2$, "date.minimum.sign") = "-" THEN
        IntermediateResult% = 1          'maximum date < minimum date
    END IF
END IF
Contradiction% = IntermediateResult%
END FUNCTION 'Contradiction% _____'

'=====
SUB CopyDate (Num&)
'=====
' Will copy the already existing dates and give back as 'Num&' the first
' occurence of the file record which belongs to the same name.
'
' Handling:
' 8/ 4/2001 - 1/22/2003:    Norbert Südland
' Translation:
' 2/19/2008:                Norbert Südland
'-----
DIM Data$      'AS STRING
DIM Array$
DIM Variable$
DIM DateValue$
DIM c%         'AS INTEGER

PRINT "Copy dates..."
Num& = ReadIn&(Num&, Data$)
Array$ = SPACE$(ArrayLength%)
FOR c% = 1 TO 6

    'Save the Corresponding Name:
    '-----
    GET #RelationFile%(c%), 1, Array$
    Change Array$, "array.date", NameRegister$(Num&) + SPACE$(ArrayLength%)
    PUT #RelationFile%(c%), 1, Array$

    'Save the Corresponding Dates:
    '-----
    Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
    DateValue$ = Part$(Data$, Variable$)
    WriteTo RelationFile%(c%), RL&(c%), DateValue$, c%, 0, c%, Num&, Num&
NEXT c%
END SUB 'CopyDate _____'

'=====
SUB Correction (Array$, Rest%, RL&)
'=====
' Will correct the dates where needed after finding a new date.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' 1/15/2003 - 1/24/2003:    Norbert Südland, Aalen
' 3/29/2009 - 10/ 8/2016:    Norbert Südland, Aalen
' Translation:
' 2/19/2008 - 10/13/2016:    Norbert Südland, Aalen
'-----
DIM Buffer$          'AS STRING
DIM Comparison$

```

```

DIM Data$
DIM Destination$
DIM Difference$
DIM Direction$
DIM IntermediateResult$
DIM Remainder$
DIM Source$
DIM Status$
DIM Text$
DIM Tol$
DIM Variable$
DIM correct%           'AS INTEGER
DIM FoundCase%
DIM L%
DIM s1%, y1%, m1%, d1%
DIM s2%, y2%, m2%, d2%
DIM Dummy&            'AS LONG
DIM Place&

Dummy& = ReadIn&(VAL(Part$(Array$, "array.to")), Data$)
Variable$ = LTRIM$(STR$(Moment%(Part$(Array$, "array.destination"))))
Variable$ = "data.date[" + Variable$ + "]"
Destination$ = Part$(Data$, Variable$)
IntermediateResult$ = Destination$
Place& = VAL(Part$(Array$, "array.from"))
Dummy& = ReadIn&(Place&, Comparison$)
Variable$ = LTRIM$(STR$(Moment%(Part$(Array$, "array.source"))))
Variable$ = "data.date[" + Variable$ + "]"
Source$ = Part$(Comparison$, Variable$)
FoundCase% = VAL(Part$(Array$, "array.name"))

SELECT CASE FoundCase%
CASE 0          'carry-over by simultaneous moments
    IntermediateResult$ = Part$(Array$, "array.date")
CASE 1, 2
    Variable$ = "data.p[" + LTRIM$(STR$(FoundCase%)) + "].direct"
    IF ToUser$(Part$(Data$, Variable$)) = "N" THEN
        Change IntermediateResult$, "date.minimum", SPACE$(SIZEOF%("term"))
        NewStatus IntermediateResult$
    END IF
CASE 3, 4
    Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 2)) + "].tol1"
    Tol$ = Part$(Data$, Variable$)
    Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 2)) + "].date"
    Buffer$ = Part$(Data$, Variable$)
    IntermediateResult$ = Total$(Buffer$, Tol$, Destination$, "--")
CASE 5, 6
    Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 4)) + "].tol2"
    Tol$ = Part$(Data$, Variable$)
    IF Moment%(Part$(Array$, "array.destination")) <= 3 THEN
        Direction$ = "+"
    ELSE
        Direction$ = "--"
    END IF
    Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 4)) + "].duration"
    Buffer$ = Part$(Data$, Variable$)
    IntermediateResult$ = Total$(Buffer$, Tol$, Destination$, Direction$)

```

```

CASE 7, 8
  Variable$ = "data.p[" + LTRIM$(STR$(FoundCase% - 6)) + "].direct"
  IF ToUser$(Part$(Comparison$, Variable$)) = "N" THEN
    Change IntermediateResult$, "date.maximum", SPACE$(SIZEOF%("term"))
    NewStatus IntermediateResult$
  END IF
CASE 9, 10
  Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 8)) + "].tol1"
  Tol$ = Part$(Comparison$, Variable$)
  Variable$ = "data.r[" + LTRIM$(STR$(FoundCase% - 8)) + "].date"
  Buffer$ = Part$(Comparison$, Variable$)
  IntermediateResult$ = Total$(Buffer$, Tol$, Destination$, "+")
END SELECT

'Check, Whether 'IntermediateResult$' will Change the Value of 'Source$':
'-----'
correct% = 0
L% = SIZEOF%("date")
IF IntermediateResult$ <> SPACE$(L%) THEN
  IF Source$ = SPACE$(L%) THEN
    correct% = 1          'date is not yet given
  ELSE
    Status$ = Part$(Source$, "date.status")
    IF Status$ = "<" THEN
      Buffer$ = Part$(IntermediateResult$, "date.minimum")
      Change Source$, "date.minimum", Buffer$
      NewStatus Source$
      IF Part$(Source$, "date.status") = " " THEN
        correct% = 1
      END IF
    END IF
    IF Status$ = ">" THEN
      Buffer$ = Part$(IntermediateResult$, "date.maximum")
      Change Source$, "date.maximum", Buffer$
      NewStatus Source$
      IF Part$(Source$, "date.status") = " " THEN
        correct% = 1
      END IF
    END IF
    Difference$ = IntermediateResult$
    Add Difference$, "U", "-", Source$, 1, 1, "-"
    TermToNumber Part$(Difference$, "date.minimum"), s1%, y1%, m1%, d1%
    TermToNumber Part$(Difference$, "date.maximum"), s2%, y2%, m2%, d2%
    L% = SIZEOF%("term")
    Buffer$ = Part$(IntermediateResult$, "date.maximum")
    IF s2% = -1 AND Buffer$ <> SPACE$(L%) THEN
      correct% = 1
      Change Source$, "date.maximum", Buffer$
    END IF
    Reverse s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, "-"
    Buffer$ = Part$(IntermediateResult$, "date.minimum")
    IF s1% = -1 AND Buffer$ <> SPACE$(L%) THEN
      correct% = 1
      Change Source$, "date.minimum", Buffer$
    END IF
    IF correct% = 1 THEN
      IntermediateResult$ = Source$
      SELECT CASE Part$(IntermediateResult$, "date.status")

```

```

        CASE PreDatingEarliest$           ' " | "
            Pause
            ERROR 114
        CASE PreDatingComplete$           ' " | "
            Pause
            ERROR 114
        CASE PreDatingLatest$             ' " | "
            Pause
            ERROR 114
        CASE ELSE
            NewStatus IntermediateResult$
        END SELECT
    END IF
END IF
END IF
IF correct% = 1 THEN
    ProtocolMessage IntermediateResult$, 0
    IF Contradiction$(IntermediateResult$) = 1 THEN
        Pause
        ERROR 113
    END IF
    L% = SIZEOF$("remainder")
    Remainder$ = SPACE$(L%)
    RL% = RL% + 1
    Change Remainder$, "remainder.position", STR$(RL%)
    Change Remainder$, "remainder.number", STR$(Place%)
    Change Remainder$, "remainder.moment", Part$(Array$, "array.source")
    Change Remainder$, "remainder.name", NameRegister$(Place%)
    Change Remainder$, "remainder.date", IntermediateResult$
    Change Remainder$, "remainder.end", "|" + CHR$(13) + CHR$(10)
    PUT #Rest%, RL% * L% + 1, Remainder$

    Variable$ = LTRIM$(STR$(Moment$(Part$(Array$, "array.source"))))
    Variable$ = "data.date[" + Variable$ + "]"
    Change Comparison$, Variable$, IntermediateResult$
    SaveDate Place%, Comparison$

    Buffer$ = Part$(Remainder$, "remainder.moment")
    Text$ = " " + STR$(RL%) + RIGHT$(OrdinaryNumber$(RL% MOD 100), 2)
    Text$ = Text$ + " position to" + Buffer$ + " " + NameRegister$(Place%)
    Text$ = Text$ + STR$(Place%)
    ProtocolMessage Text$, 1
END IF
END SUB 'Correction _____

'=====
SUB Evaluate (Rest%, RL%, Num%, What%, CountingMode%)
'=====
' Will evaluate a data record.
'
' Handling:
' 6/29/2001 - 3/18/2017: Norbert Südland
' Translation:
' 3/ 3/2008 - 3/22/2017: Norbert Südland
'-----
DIM Array$           'AS STRING
DIM Buffer$
DIM Data$

```

```
DIM DateValue$
DIM Remainder$
DIM Simultaneous$
DIM Variable$
DIM Text$
DIM evaluation%      'AS INTEGER
DIM Found%
DIM L%
DIM m%(3)
DIM NameType%
DIM x%
DIM c%               'AS LONG
DIM First%
DIM p%               'position
DIM counter%

NameType% = What%
p% = RL$(NameType%)
Array$ = Load$(RelationFile$(NameType%), ArrayLength%, p%, p% - 1)
IF RL$(NameType%) = 1 THEN
    Text$ = "Date without connection:"
    ProtocolMessage Text$, 1
    ProtocolMessage Array$, 1
    Pause
    ERROR 112
ELSE
    'Are Relations Available?

    'Find Evaluation Difference:
    '-----'
    L% = SIZEOF$("term")
    evaluation% = 0      'Date is akceptable
    IF Part$(Array$, "array.date.minimum") <> SPACE$(L%) THEN
        evaluation% = 1  'Date will be changed
    END IF
    IF Part$(Array$, "array.date.maximum") <> SPACE$(L%) THEN
        evaluation% = evaluation% + 2
    END IF

    'Find the Newly Calculated Date:
    '-----'
    p% = RL$(NameType%)
    Array$ = Load$(RelationFile$(NameType%), ArrayLength%, p%, p%)
    DateValue$ = Part$(Array$, "array.date")
    NewStatus DateValue$
    First% = ReadIn$(Num%, Data$)
    Simultaneous$ = ToUser$(Part$(Data$, "data.simultaneous"))

    'Correct the Date:
    '-----'
    IF evaluation% >= 1 THEN
        ProtocolMessage DateValue$, 0
        IF Contradiction$(DateValue$) = 1 THEN
            Pause
            ERROR 113
        END IF
        IF NameType% = Moment$(Part$(Array$, "array.destination")) THEN
```

```

'Build up New Record:
'-----'
L% = SIZEOF%("remainder")
Remainder$ = SPACE$(L%)
RL& = RL& + 1
Change Remainder$, "remainder.position", STR$(RL&)
Change Remainder$, "remainder.number", STR$(Num&)
Change Remainder$, "remainder.moment", Symbol$(NameType%)
Change Remainder$, "remainder.name", NameRegister$(Num&)
Change Remainder$, "remainder.date", DateValue$
Change Remainder$, "remainder.end", "|" + CHR$(13) + CHR$(10)
PUT #Rest%, RL& * L% + 1, Remainder$

'Change the Date, but Not the Pre-dating:
'-----'
Variable$ = "data.date[" + LTRIM$(STR$(NameType%)) + "]"
SELECT CASE Part$(Data$, Variable$ + ".status")
CASE PreDatingEarliest$   "I"
    IF evaluation% MOD 2 = 1 THEN
        Pause
        ERROR 114
    END IF
    Buffer$ = Part$(DateValue$, "date.maximum")
    Change Data$, Variable$ + ".maximum", Buffer$
CASE PreDatingComplete$   "I"
    Pause
    ERROR 114
CASE PreDatingLatest$     "I"
    IF evaluation% MOD 2 = 0 THEN
        Pause
        ERROR 114
    END IF
    Buffer$ = Part$(DateValue$, "date.minimum")
    Change Data$, Variable$ + ".minimum", Buffer$
CASE ELSE
    Change Data$, Variable$, DateValue$
END SELECT
SaveDate Num&, Data$

Text$ = " " + STR$(RL&) + RIGHT$(OrdinaryNumber$(RL& MOD 100), 2)
Text$ = Text$ + " position to "
Text$ = Text$ + ToUser$(LTRIM$(Part$(Remainder$, "remainder.moment")))
Text$ = Text$ + " " + NameRegister$(Num&) + STR$(Num&)
ProtocolMessage Text$, 1
ELSE
    Pause
    ERROR 115
END IF
END IF

'Check Back the Simultaneous Moments:
'-----'
Found% = INSTR(Simultaneous$, Symbol$(NameType%))
IF Found% > 0 THEN
    IF Found% > 3 THEN
        FOR x% = 4 TO 5
            m%(x%) = Moment%(MID$(Simultaneous$, x%, 1))
        NEXT x%
    
```

```

        m%(3) = 0
        Found% = Found% - 3
    ELSE
        FOR x% = 1 TO 3
            m%(x%) = Moment%(MID$(Simultaneous$, x%, 1))
        NEXT x%
    END IF
    FOR x% = 1 TO 3
        IF x% <> Found% AND m%(x%) > 0 THEN
            Change Array$, "array.destination", Symbol$(m%(x%))
            Change Array$, "array.from", Part$(Array$, "array.to")
            Correction Array$, Rest%, RL&
        END IF
    NEXT x%
    NameType% = m%(1)          'All relations are there!
END IF

'Check Back the New Calculated Date:
'-----'
FOR counter& = 1 TO RL&(NameType%) - 3
    c& = SequenceOrder&(CountingMode%, RL&(NameType%) - 3, counter&) + 1
    p& = RL&(NameType%)
    Array$ = Load$(RelationFile%(NameType%), ArrayLength%, p&, c&)
    'Pause
    Change Array$, "array.date", DateValue$
    Correction Array$, Rest%, RL&
NEXT counter&
END IF

END SUB 'Evaluate _____'

'=====
FUNCTION Find& (Name$, Place&, NameType%)
'=====
' Will seek for `Name$` in the name lists and will give
' the first file position of `Name$` since `Place&` <= `GIL&`.
'
' Handling:
' 6/27/2001 - 1/30/2003:    Norbert Südland
' Translation:
' 3/ 3/2008:                Norbert Südland
'-----'
DIM Comparison$      'AS STRING
DIM List$
DIM GoOn%            'AS INTEGER
DIM L%
DIM Start&           'AS LONG
DIM Finish&
DIM midth&
DIM Found&
DIM IntermediateResult&
DIM res&              'result

'General Seek:
'-----'
Start& = 1
Finish& = NL&(NameType%)

```

```

L% = SIZEOF("list")
List$ = SPACE$(L%)
Found& = 0
WHILE Found& = 0 AND Start& <= Finish&
    midth& = CLNG((Start& + Finish&) / 2)
    GET #Names%(NameType%), midth& * L% + 1, List$
    Comparison$ = Part$(List$, "list.name")
    SELECT CASE Comparison$
        CASE IS = Name$
            Found& = midth&
        CASE IS < Name$
            Start& = midth& + 1
        CASE IS > Name$
            Finish& = midth& - 1
    END SELECT
WEND

'Find First Occurence Since `Place&` (Concerning Sorted Order):
'-----'
GoOn% = RIGHT
IF Found& > 0 THEN
    IntermediateResult& = VAL(Part$(List$, "list.number"))
ELSE
    IntermediateResult& = 0
END IF
SELECT CASE IntermediateResult&
CASE 0
CASE IS > Place&
    WHILE Found& > 1 AND IntermediateResult& > Place& AND GoOn% = RIGHT
        GET #Names%(NameType%), (Found& - 1) * L% + 1, List$
        Comparison$ = Part$(List$, "list.name")
        IF Name$ = Comparison$ THEN
            IF VAL(Part$(List$, "list.number")) >= Place& THEN
                Found& = Found& - 1
                IntermediateResult& = VAL(Part$(List$, "list.number"))
            ELSE
                GoOn% = WRONG
            END IF
        ELSE
            GoOn% = WRONG
        END IF
    WEND
CASE IS < Place&
    res& = IntermediateResult&
    WHILE Found& < NL&(NameType%) AND res& < Place& AND GoOn% = RIGHT
        IntermediateResult& = res&
        GET #Names%(NameType%), (Found& + 1) * L% + 1, List$
        Comparison$ = Part$(List$, "list.name")
        IF Name$ = Comparison$ THEN
            Found& = Found& + 1
            IntermediateResult& = VAL(Part$(List$, "list.number"))
        ELSE
            GoOn% = WRONG
        END IF
        res& = IntermediateResult&
    WEND
END SELECT

```

```

'Result:
'-----'
IF IntermediateResult& < Place& THEN
    IntermediateResult& = 0
END IF
Find& = IntermediateResult&
END FUNCTION 'Find& _____'

'=====
FUNCTION Intersection$ (DateValue1$, DateValue2$)
'=====
'Generates the intersection of 'DateValue1$' and 'DateValue2$'.
'If the intersection is empty, a contradiction will be found.
'This is one of the few sensible applications of the command GOTO
'to avoid encapsulated IF-commands.
'
'Handling: 09/20/2016 - 09/28/2016 Norbert Südland, Aalen
'-----'

DIM Term1$
DIM Term2$
DIM Variable$
DIM ResultTerm1$
DIM ResultTerm2$
DIM ResultDate$

DIM Length%
DIM Sign1%, Year1%, Month1%, Day1%
DIM Sign2%, Year2%, Month2%, Day2%

'Preparation:
'-----'
IF DateValue1$ = DateValue2$ THEN
    ResultDate$ = DateValue1$
    GOTO ReturnIntersection
END IF
ResultDate$ = SPACE$(SIZEOF%("date"))
Length% = SIZEOF%("term")

'Find New Minimum:
'-----'
Variable$ = "date.minimum"
Term1$ = Part$(DateValue1$, Variable$)
Term2$ = Part$(DateValue2$, Variable$)

IF Term1$ = Term2$ THEN
    ResultTerm1$ = Term1$
    GOTO InsertMinimum
END IF

IF Term1$ = SPACE$(Length%) THEN
    ResultTerm1$ = Term2$
    GOTO InsertMinimum
END IF

IF Term2$ = SPACE$(Length%) THEN
    ResultTerm1$ = Term1$

```

```
    GOTO InsertMinimum
END IF

TermToNumber Term1$, Sign1%, Year1%, Month1%, Day1%
TermToNumber Term2$, Sign2%, Year2%, Month2%, Day2%

IF Sign1% <> Sign2% THEN
    IF Sign1% > 0 THEN
        ResultTerm1$ = Term1$
    ELSE
        ResultTerm1$ = Term2$
    END IF
    GOTO InsertMinimum
END IF
IF Sign1% * Year1% <> Sign2% * Year2% THEN
    IF Sign1% * Year1% > Sign2% * Year2% THEN
        ResultTerm1$ = Term1$
    ELSE
        ResultTerm1$ = Term2$
    END IF
    GOTO InsertMinimum
END IF

IF Month1% <> Month2% THEN
    IF Month1% > Month2% THEN
        ResultTerm1$ = Term1$
    ELSE
        ResultTerm1$ = Term2$
    END IF
    GOTO InsertMinimum
END IF

IF Day1% <> Day2% THEN
    IF Day1% > Day2% THEN
        ResultTerm1$ = Term1$
    ELSE
        ResultTerm1$ = Term2$
    END IF
    GOTO InsertMinimum
END IF
```

Pause 'If this is reached, then a case has been overseen during programming.

```
'===== '
InsertMinimum:
'===== '
```

```
    Change ResultDate$, Variable$, ResultTerm1$
```

```
'Find New Maximum:
'-----'
```

```
Variable$ = "date.maximum"
Term1$ = Part$(DateValue1$, Variable$)
Term2$ = Part$(DateValue2$, Variable$)
```

```
IF Term1$ = Term2$ THEN
    ResultTerm2$ = Term1$
    GOTO InsertMaximum
```

END IF

IF Term1\$ = SPACE\$(Length%) THEN

ResultTerm2\$ = Term2\$

GOTO InsertMaximum

END IF

IF Term2\$ = SPACE\$(Length%) THEN

ResultTerm2\$ = Term1\$

GOTO InsertMaximum

END IF

TermToNumber Term1\$, Sign1%, Year1%, Month1%, Day1%

TermToNumber Term2\$, Sign2%, Year2%, Month2%, Day2%

IF Sign1% <> Sign2% THEN

IF Sign1% < 0 THEN

ResultTerm2\$ = Term1\$

ELSE

ResultTerm2\$ = Term2\$

END IF

GOTO InsertMaximum

END IF

IF Sign1% \* Year1% <> Sign2% \* Year2% THEN

IF Sign1% \* Year1% < Sign2% \* Year2% THEN

ResultTerm2\$ = Term1\$

ELSE

ResultTerm2\$ = Term2\$

END IF

GOTO InsertMaximum

END IF

IF Month1% <> Month2% THEN

IF Month1% < Month2% THEN

ResultTerm2\$ = Term1\$

ELSE

ResultTerm2\$ = Term2\$

END IF

GOTO InsertMaximum

END IF

IF Day1% <> Day2% THEN

IF Day1% < Day2% THEN

ResultTerm2\$ = Term1\$

ELSE

ResultTerm2\$ = Term2\$

END IF

GOTO InsertMaximum

END IF

Pause 'If this is reached, then a case has been overseen during programming.

'====='

InsertMaximum:

'====='

Change ResultDate\$, Variable\$, ResultTerm2\$

```

'Generate New Status:
'-----'
NewStatus ResultDate$

'=====
ReturnIntersection:
'=====
Intersection$ = ResultDate$
END FUNCTION 'Intersection$ _____

'=====
FUNCTION Load$ (File%, LengthOfDataSet%, FileLength%, Position%)
'=====
' Will load the `Position`th record of a binary `File` with
' `FileLength` records of the data set length `LengthOfDataSet`.
' The 0th record is used for documentation.
'
' Handling:
' 8/ 4/2001 - 1/20/2003:    Norbert Südland
' Translation:
' 3/ 3/2008:                Norbert Südland
'-----
DIM Array$      'AS STRING
DIM L%          'AS INTEGER

Array$ = SPACE$(LengthOfDataSet%)
IF Position% = 0 THEN
    Position% = FileLength%
END IF
IF File% = 0 THEN
    Pause        'Problems with translating this program?
ELSE
    GET #File%, Position% * LengthOfDataSet% + 1, Array$
END IF

Load$ = Array$
END FUNCTION 'Load$ _____

'=====
FUNCTION Moment% (SymbolText$)
'=====
' Will transform the moment symbol `SymbolText$` into an ordinary number
' of date. This function will work correctly with old data files, too.
'
' Handling:
' 9/ 4/2001 - 1/22/2003:    Norbert Südland
' Translation:
' 10/ 9/2007 - 11/19/2007:  Norbert Südland
' Addition:
' 11/20/2007:                Norbert Südland
'-----
DIM IntermediateResult% 'AS INTEGER

SELECT CASE RTRIM$(LTRIM$(SymbolText$))
CASE "*"
    IntermediateResult% = 1

```

```

CASE "B"
    IntermediateResult% = 2
CASE CHR$(224)      'alpha
    IntermediateResult% = 3
CASE CHR$(234)      'Omega
    IntermediateResult% = 4
CASE "E"
    IntermediateResult% = 5
CASE "+"
    IntermediateResult% = 6
CASE ELSE
    IntermediateResult% = 0
END SELECT

Moment% = IntermediateResult%
END FUNCTION 'Moment% _____'

'=====
FUNCTION NameRegister$ (Position&)
'=====
' Will read the `Position&`th name entry from the `InputFile%`.
' The 0th file record is used for documentation.
'
' Handling:
' 12/23/2002 - 1/20/2003:    Norbert Südland
' Translation:
' 3/ 3/2008:                Norbert Südland
'-----
DIM Data$      'AS STRING

Data$ = Load$(InputFile%, DataLength%, GIL&, Position&)
NameRegister$ = Part$(Data$, "data.name")
END FUNCTION 'NameRegister$ _____'

'=====
SUB NewStatus (DateValue$)
'=====
' Will check and correct the status of `DateValue$`.
'
' Handling:
' 9/ 4/2001: Norbert Südland, Munich
' Translation:
' 3/ 3/2008: Norbert Südland, Aalen
'-----
DIM Status$    'AS STRING
DIM Length%    'AS INTEGER

Length% = SIZEOF$("term")
IF Part$(DateValue$, "date.minimum") = SPACE$(Length%) THEN
    Status$ = "≤"
ELSE
    Status$ = " "
END IF
IF Part$(DateValue$, "date.maximum") = SPACE$(Length%) THEN
    IF Status$ = " " THEN
        Status$ = "≥"
    
```

```

    ELSE
        Status$ = " "
    END IF
END IF
Change DateValue$, "date.status", Status$
END SUB 'NewStatus _____'

'=====
FUNCTION NumberToTerm$ (Sign%, Year%, Month%, Day%)
'=====
' Will convert a series of date numbers to a string.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland
' Translation:
' 3/15/2008:                Norbert Südland
'-----
DIM Term$      'AS STRING

Term$ = SPACE$(SIZEOF$("term"))
Year% = Sign% * Year%
IF Year% < 0 THEN
    Sign% = -1
    Year% = -Year%
ELSE
    Sign% = 1
END IF
IF Sign% = 1 THEN
    Change Term$, "term.sign", " "
ELSE
    Change Term$, "term.sign", "-"
END IF
IF Year% <> 0 THEN
    Change Term$, "term.year", LTRIM$(STR$(Year%))
    Change Term$, "term.ys", "."
END IF
IF Month% <> 0 THEN
    Change Term$, "term.month", LTRIM$(STR$(Month%))
    Change Term$, "term.ms", "."
END IF
IF Day% <> 0 THEN
    Change Term$, "term.day", LTRIM$(STR$(Day%))
    Change Term$, "term.ds", "."
END IF
NumberToTerm$ = Term$
END FUNCTION 'NumberToTerm$ _____'

'=====
FUNCTION NumberToTime$ (Year%, Month%, Day%, Mode%)
'=====
' Will change a series of date numbers to a string.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 3/15/2008:                Norbert Südland, Aalen
'-----

```

```

DIM Instant$ 'AS STRING

Instant$ = SPACE$(SIZEOF%("time"))
IF Year% > 0 THEN
    Change Instant$, "time.year", LTRIM$(STR$(Year%))
    IF Mode% = 1 THEN
        Change Instant$, "time.y", " y "
    ELSE
        Change Instant$, "time.y", ".y "
    END IF
END IF
IF Month% > 0 THEN
    Change Instant$, "time.month", LTRIM$(STR$(Month%))
    IF Mode% = 1 THEN
        Change Instant$, "time.m", " m "
    ELSE
        Change Instant$, "time.m", ".m "
    END IF
END IF
IF Day% > 0 THEN
    Change Instant$, "time.day", LTRIM$(STR$(Day%))
    IF Mode% = 1 THEN
        Change Instant$, "time.d", " d"
    ELSE
        Change Instant$, "time.d", ".d"
    END IF
END IF
NumberToTime$ = Instant$
END FUNCTION 'NumberToTime$ _____

'=====
SUB Optimize
'=====
' Will build the intersection set of the current dates that are available.
'
' Handling:
' 6/27/2001 - 1/20/2003: Norbert Suedland
' Translation:
' 3/15/2008: Norbert Suedland
'-----

DIM Addition$ 'AS STRING
DIM Array$
DIM Buffer$
DIM Comparison$
DIM Comparison1$
DIM Comparison2$
DIM Data$
DIM Default$
DIM Sum$
DIM Status$
DIM Variable$
DIM c% 'AS INTEGER
DIM Found%
DIM L%
DIM Length%
DIM m%(5)
DIM rf%

```

```

DIM s1%, y1%, m1%, d1%
DIM s2%, y2%, m2%, d2%
DIM y%
DIM IntermediateResult&          'AS LONG
DIM Num&
DIM Place&
DIM Start&

'Consider the Simultaneousness of the Moments:
'=====
Array$ = Load$(RelationFile%(1), ArrayLength%, RL%(1), 1)
Num& = VAL(Part$(Array$, "array.to"))
IntermediateResult& = ReadIn&(Num&, Data$)
Buffer$ = ToUser$(Part$(Data$, "data.simultaneous"))
Buffer$ = SimultaneousnessCheck$(Buffer$)
FOR c% = 1 TO 5
    m%(c%) = Moment%(MID$(Buffer$, c%, 1))
NEXT c%
IF Buffer$ <> SPACE$(SIZEOF("data.simultaneous")) THEN
    FOR c% = 2 TO 3
        IF m%(c%) <> 0 THEN
            Shift RelationFile%(), RL%(), m%(c%), m%(1)
        END IF
    NEXT c%
    IF m%(4) <> 0 THEN
        Shift RelationFile%(), RL%(), m%(5), m%(4)
    END IF
END IF

'New Limits for Minimal/Maximal Date:
'=====
FOR c% = 1 TO 6
    Default$ = Load$(RelationFile%(c%), ArrayLength%, RL%(c%), 1)
    SELECT CASE Part$(Default$, "array.date.status")
    CASE PreDatingEarliest$      ' "I"
        Start& = 1
    CASE PreDatingComplete$      ' "II"
        Start& = 1
    CASE PreDatingLatest$        ' "III"
        Start& = 1
    CASE ELSE
        Start& = 2
    END SELECT
    Length% = SIZEOF("term")
    IF RL%(c%) > 1 THEN
        Comparison1$ = Load$(RelationFile%(c%), ArrayLength%, RL%(c%), Start&)
        Comparison2$ = Comparison1$
        FOR y% = 1 TO 2          'Minimum und Maximum getrennt behandeln
            FOR Place& = Start& + 1 TO RL%(c%)
                Buffer$ = Load$(RelationFile%(c%), ArrayLength%, RL%(c%), Place&)
                Comparison2$ = Buffer$
                L% = SIZEOF("date")
                IF Part$(Comparison1$, "array.date") = SPACE$(L%) THEN
                    Comparison1$ = Comparison2$
                ELSE
                    IF y% = 1 THEN
                        Variable$ = "array.date.minimum"
                    
```

```

ELSE
    Variable$ = "array.date.maximum"
END IF
IF Part$(Comparison2$, Variable$) = SPACE$(Length%) THEN
    Found% = 0
ELSE
    Found% = 1
END IF
IF Found% = 1 THEN
    Sum$ = Part$(Comparison1$, "array.date")
    Addition$ = Part$(Comparison2$, "array.date")
    Add Sum$, "U", "-", Addition$, 1, 1, "-"
    TermToNumber Part$(Sum$, "date.minimum"), s1%, y1%, m1%, d1%
    TermToNumber Part$(Sum$, "date.maximum"), s2%, y2%, m2%, d2%
    IF y% = 1 THEN
        Status$ = Part$(Comparison1$, "array.date.status")
        IF Status$ = "<" OR s1% = -1 THEN
            Addition$ = Part$(Comparison2$, Variable$)
            Change Comparison1$, Variable$, Addition$
        END IF
    ELSE
        Reverse s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, "-"
        Status$ = Part$(Comparison1$, "array.date.status")
        IF Status$ = ">" OR s2% = -1 THEN
            Addition$ = Part$(Comparison2$, Variable$)
            Change Comparison1$, Variable$, Addition$
        END IF
    END IF
    Addition$ = Part$(Comparison1$, "array.date")
    NewStatus Addition$
    Change Comparison1$, "array.date", Addition$
END IF
END IF
NEXT Place&
NEXT y%

'Save Difference:
'-----'

Result RelationFile%(c%), RL%(c%), Default$, Comparison1$, c%, c%, Num&
END IF
NEXT c%

'Simultaneous Moments:
'-----'
FOR c% = 2 TO 3
    IF m%(c%) <> 0 THEN
        'Pause
        Default$ = Load$(RelationFile%(m%(c%)), ArrayLength%, RL%(m%(c%)), 0)
        Comparison$ = Load$(RelationFile%(m%(1)), ArrayLength%, RL%(m%(1)), 0)
        rf% = RelationFile%(m%(c%))
        Result rf%, RL%(m%(c%)), Default$, Comparison$, m%(c%), m%(1), Num&
    END IF
NEXT c%
IF m%(5) <> 0 THEN
    Default$ = Load$(RelationFile%(m%(5)), ArrayLength%, RL%(m%(5)), 0)
    Comparison$ = Load$(RelationFile%(m%(4)), ArrayLength%, RL%(m%(4)), 0)
    rf% = RelationFile%(m%(5))
    Result rf%, RL%(m%(5)), Default$, Comparison$, m%(5), m%(4), Num&

```

```

END IF
END SUB 'Optimize _____'

'=====
FUNCTION OrdinaryNumber$ (Num%)
'=====
' Will generate the correct ending of a written ordinary number.
'
' Handling:
' 9/15/2007: Norbert Suedland
' Check:
' 9/15/2007: Norbert Suedland
'-----

DIM current&
DIM IntermediateResult$

IntermediateResult$ = LTRIM$(STR$(Num%))
IF Num% < 0 THEN
    current& = Num%
    current& = (-current&) MOD 100
ELSE
    current& = Num% MOD 100
END IF
IF current& - (current& MOD 10) <> 10 THEN
    SELECT CASE current& MOD 10
    CASE 1
        IntermediateResult$ = IntermediateResult$ + "st"
    CASE 2
        IntermediateResult$ = IntermediateResult$ + "nd"
    CASE 3
        IntermediateResult$ = IntermediateResult$ + "rd"
    CASE ELSE
        IntermediateResult$ = IntermediateResult$ + "th"
    END SELECT
ELSE
    IntermediateResult$ = IntermediateResult$ + "th"
END IF

OrdinaryNumber$ = IntermediateResult$
END FUNCTION 'OrdinaryNumber$ _____'

'=====
FUNCTION Part$ (Data$, Variable$)
'=====
' Will extract the part of `Data$` that is given by `Variable$`.
'
' Handling:
' 8/ 4/2001 - 2/12/2003: Norbert Suedland
' Translation:
' 3/20/2008: Norbert Suedland
'-----

DIM FurtherSeek$ 'AS STRING
DIM IntermediateResult$
DIM Seek$
DIM L% 'AS INTEGER
DIM p%
DIM Where%

```

```

L% = LEN(Variable$)
p% = STRLEN$(Variable$, ".") + 1
IF p% = L% THEN
    Seek$ = Variable$
ELSE
    p% = p% + STRLEN$(MID$(Variable$, p% + 1, L% - p%), ".") + 1
    IF p% < L% THEN
        Seek$ = LEFT$(Variable$, p% - 1)
    ELSE
        Seek$ = Variable$
    END IF
END IF
Where% = QuickPosition$(GVName$(), Seek$)
IF Where% = 0 THEN
    IntermediateResult$ = ""
ELSE
    IntermediateResult$ = MID$(Data$, GVBegin$(Where%), GVLength$(Where%))
    IF p% < L% THEN
        FurtherSeek$ = GVType$(Where%) + MID$(Variable$, p%, L% - p% + 1)
        IntermediateResult$ = Part$(IntermediateResult$, FurtherSeek$)
    END IF
END IF
Part$ = IntermediateResult$
END FUNCTION 'Part$ _____'

'=====
SUB Pause
'=====
' Will present a statement in line 25 and wait for a pressed key.
' This pause gives the opportunity to interrupt the program and look at the
' program code.
' Under Windows XP, the command STOP not always works reliably.
'
' Handling:
' 9/ 4/2001: Norbert Südland, Munich
' 10/10/2016: Norbert Südland, Aalen
' 10/21/2022: Norbert Südland, Aalen
' Translation:
' 2/ 5/2008: Norbert Südland, Aalen
' 10/13/2016: Norbert Südland, Aalen
' 3/ 7/2017: Norbert Südland, Aalen
'-----

DIM Answer$ 'AS STRING
DIM x%      'AS INTEGER
DIM y%

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

'Clear Line 25:
'-----
COLOR 7, 0
LOCATE 25, 1
PRINT SPACE$(80);

```

```

'Print Text Messages:
'-----'
LOCATE 25, 6
COLOR 0, 7
PRINT " Press any key to go on ";
COLOR 15, 0
PRINT " View Program Code: ";
COLOR 0, 7
IF ENVIRON$("COMSPEC") = "Z:\COMMAND.COM" THEN
    PRINT " [ Ctrl ] - [ Scroll ] ";
ELSEIF ENVIRON$("DOSDIR") <> "" THEN
    PRINT " [ Ctrl ] - [ Scroll ] ";
ELSE
    PRINT " [ Ctrl ] - [ Pause ] ";
END IF
COLOR 7, 0

'Wait for Pressed Key:
'-----'
WHILE INKEY$ <> ""                'Empty the keyboard buffer!
WEND
DO
    Answer$ = INKEY$
LOOP WHILE LEN(Answer$) = 0

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

'Set Cursor to Old Position:
'-----'
LOCATE y%, x%

END SUB 'Pause _____'

'=====
SUB ProtocolMessage (MessageText$, NewLine%)
'=====
' Appends `MessageText$` to `ProtocolFile$`.
'
' Handling:
' 10/ 7/2016 - 10/ 8/2016 Norbert Südland, Aalen
'
' Translation:
' 10/13/2016                Norbert Südland, Aalen
'-----'
DIM Protocol%

IF NewLine% = 0 THEN
    PRINT MessageText$;
ELSE
    PRINT MessageText$
END IF
IF ProtocolFile$ <> "" THEN
    Protocol% = FREEFILE

```

```

501 OPEN ProtocolFile$ FOR APPEND AS #Protocol%
    IF NewLine% = 0 THEN
        PRINT #Protocol%, MessageText$;
    ELSE
        PRINT #Protocol%, MessageText$
    END IF
    CLOSE #Protocol%
END IF

END SUB 'ProtocolMessage _____'

'=====
FUNCTION QuickPosition% (List$(), Name$)
'=====
' Will seek in `List$()` for `Name$`.
' - It is assumed, that the `List$()` is sorted alphabetically, where all
'   special characters are dealt with due to their ASCII numbers.
' - The result will be definite, if `Name$` occurs not more than once within
'   the `List$()`.
' - The internal index number within `List$()` is not necessary to know of.
'
' Handling:
' 8/ 4/2001 - 12/27/2002:    Norbert Südland
' Check:
'
' Translation:
' 3/20/2008:                Norbert Südland
'-----
DIM SeekStart%      'AS INTEGER
DIM SeekEnd%
DIM Found%
DIM CheckResult%

SeekStart% = LBOUND(List$)
SeekEnd% = UBOUND(List$)
Found% = 0
WHILE Found% = 0 AND SeekStart% <= SeekEnd%
    CheckResult% = INT(SeekStart% / 2 + SeekEnd% / 2)    'Overflow avoided!
    SELECT CASE List$(CheckResult%)
        CASE IS < Name$
            SeekStart% = CheckResult% + 1
        CASE IS = Name$
            Found% = CheckResult%
        CASE IS > Name$
            SeekEnd% = CheckResult% - 1
    END SELECT
WEND

QuickPosition% = Found%
END FUNCTION 'QuickPosition _____'

'=====
FUNCTION ReadIn& (Place&, Data$)
'=====
' Will load the information of the 'Place&'-th file record and overwrite
' them by the dates of the first record, whose position is given back.
' Here the possibility is considered, that as many records as you like can

```

```
' occur for one name. First the simultaneous moments of this name are found
' out, then by this the dates, which are already existant, are put together.
' The lastest at this point contradictions of intricately structured
' pre-datings can be recognized.
```

```
' Handling:
```

```
' 6/16/2001 - 1/24/2003: Norbert Südland
```

```
' Translation:
```

```
' 3/20/2008 - 9/28/2016: Norbert Südland
```

```
-----
DIM Addition$ 'AS STRING
DIM Buffer$
DIM DateValue$
DIM Name$
DIM NewDate$
DIM NewTerm$
DIM Simultaneous$
DIM Term$
DIM Var$
DIM Variable$
DIM c% 'AS INTEGER
DIM Length%
DIM Together%(3)
DIM NextRecord% 'AS LONG
DIM s%
```

```
'Combine Simultaneous Moments:
```

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

```
Data$ = Load$(InputFile%, DataLength%, GIL%, Place%)
Name$ = Part$(Data$, "data.name")
s% = Find$(Name$, 1, 1) 'Seek the first record
Addition$ = Load$(InputFile%, DataLength%, GIL%, s%)
Buffer$ = ToUser$(Part$(Addition$, "data.simultaneous"))
Simultaneous$ = SimultaneousnessCheck$(Buffer$)
NextRecord% = Find$(Name$, s% + 1, 1) 'Further record?
WHILE NextRecord% > 0
    Addition$ = Load$(InputFile%, DataLength%, GIL%, NextRecord%)
    Buffer$ = ToUser$(Part$(Addition$, "data.simultaneous"))
    Buffer$ = SimultaneousnessCheck$(Buffer$)
    SELECT CASE LEFT$(Simultaneous$, 3)
    CASE SPACE$(3)
        Simultaneous$ = Buffer$
    CASE "*Bα", "QE+" 'All other entries must be empty!
    CASE "*B ", "Bα ", "αE ", "QE ", "Q+ ", "E+ "
        IF RIGHT$(Simultaneous$, 2) = SPACE$(2) THEN
            Simultaneous$ = LEFT$(Buffer$, 3) + LEFT$(Simultaneous$, 2)
            Simultaneous$ = SimultaneousnessCheck$(Simultaneous$)
            IF RIGHT$(Simultaneous$, 2) = SPACE$(2) THEN
                Simultaneous$ = LEFT$(Simultaneous$, 3) + RIGHT$(Buffer$, 2)
                Simultaneous$ = SimultaneousnessCheck$(Simultaneous$)
            END IF
        END IF
    END SELECT
```

```
'Hardly Overwrite by Pre-dating:
```

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

```

FOR c% = 1 TO 6
  Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
  DateValue$ = Part$(Addition$, Variable$)
  Buffer$ = Part$(Data$, Variable$)
  SELECT CASE Part$(DateValue$, "date.status")
  CASE PreDatingEarliest$      ' "I"
    Var$ = "date.minimum"
    SELECT CASE Part$(Buffer$, "date.status")
    CASE PreDatingEarliest$, PreDatingComplete$
      IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
        Pause
        ERROR 114
      END IF
    CASE PreDatingLatest$
      Change Data$, Variable$ + ".minimum", Part$(DateValue$, Var$)
      Change Data$, Variable$ + ".status", PreDatingComplete$
    CASE ELSE
      Change Data$, Variable$ + ".minimum", Part$(DateValue$, Var$)
      Change Data$, Variable$ + ".status", PreDatingEarliest$
    END SELECT
  CASE PreDatingComplete$      ' "■"
    SELECT CASE Part$(Buffer$, "date.status")
    CASE PreDatingEarliest$
      Var$ = "date.minimum"
      IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
        Pause
        ERROR 114
      END IF
      Var$ = "date.maximum"
      Change Data$, Variable$ + ".maximum", Part$(DateValue$, Var$)
      Change Data$, Variable$ + ".status", PreDatingComplete$
    CASE PreDatingComplete$
      Var$ = "date.minimum"
      IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
        Pause
        ERROR 114
      END IF
      Var$ = "date.maximum"
      IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
        Pause
        ERROR 114
      END IF
    CASE PreDatingLatest$
      Var$ = "date.maximum"
      IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
        Pause
        ERROR 114
      END IF
      Var$ = "date.minimum"
      Change Data$, Variable$ + ".minimum", Part$(DateValue$, Var$)
      Change Data$, Variable$ + ".status", PreDatingComplete$
    CASE ELSE
      Change Data$, Variable$, DateValue$
    END SELECT
  CASE PreDatingLatest$      ' "I"
    Var$ = "date.maximum"
    SELECT CASE Part$(Buffer$, "date.status")
    CASE PreDatingEarliest$

```

```

        Change Data$, Variable$ + ".maximum", Part$(DateValue$, Var$)
        Change Data$, Variable$ + ".status", PreDatingComplete$
    CASE PreDatingComplete$, PreDatingLatest$
        IF Part$(Buffer$, Var$) <> Part$(DateValue$, Var$) THEN
            Pause
            ERROR 114
        END IF
    CASE ELSE
        Change Data$, Variable$ + ".maximum", Part$(DateValue$, Var$)
        Change Data$, Variable$ + ".status", PreDatingLatest$
    END SELECT
END SELECT
NEXT c%
NextRecord& = Find&(Name$, NextRecord& + 1, 1)
WEND
Change Data$, "data.simultaneous", ToFile$(Simultaneous$)

'Take the Date from the First Record:
'-----'
IF Place& <> s& THEN
    Addition$ = Load$(InputFile$, DataLength%, GIL&, s&)
    FOR c% = 1 TO 6
        Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
        DateValue$ = Part$(Addition$, Variable$)
        SELECT CASE Part$(Data$, Variable$ + ".status")
        CASE PreDatingEarliest$
            ' " "
            Var$ = Variable$ + ".minimum"
            Change DateValue$, "date.minimum", Part$(Data$, Var$)
            Change DateValue$, "date.status", "PreDatingEarliest$"
        CASE PreDatingComplete$
            ' " "
            DateValue$ = Part$(Data$, Variable$)
        CASE PreDatingLatest$
            ' " "
            Var$ = Variable$ + ".maximum"
            Change DateValue$, "date.maximum", Part$(Data$, Var$)
            Change DateValue$, "date.status", "PreDatingLatest$"
        CASE ELSE
            NewStatus DateValue$
        END SELECT
        Change Data$, Variable$, DateValue$
    NEXT c%
END IF

'Each Information of Simultaneous Results must be Listed:
'-----'
IF LEFT$(Simultaneous$, 3) <> SPACE$(3) THEN
    Length% = SIZEOF$("term")
    DateValue$ = SPACE$(SIZEOF$("date"))
    NewDate$ = DateValue$
    FOR c% = 1 TO 3
        Together%(c%) = Moment$(MID$(Simultaneous$, c%, 1))
        IF Together%(c%) > 0 THEN
            Variable$ = "data.date[" + LTRIM$(STR$(Together%(c%))) + "]"
            Buffer$ = Part$(Data$, Variable$)
            Var$ = "date.minimum"
            IF Part$(Buffer$, Var$) <> SPACE$(Length%) THEN
                Change DateValue$, Var$, Part$(Buffer$, Var$)
            END IF
            Var$ = "date.maximum"
        END IF
    NEXT c%
END IF

```

```

    IF Part$(Buffer$, Var$) <> SPACE$(Length%) THEN
        Change DateValue$, Var$, Part$(Buffer$, Var$)
    END IF
    NewStatus DateValue$
END IF
NewDate$ = Intersection$(NewDate$, DateValue$)
NEXT c%
FOR c% = 1 TO 3
    IF Together%(c%) > 0 THEN
        Variable$ = "data.date[" + LTRIM$(STR$(Together%(c%))) + "]"
        SELECT CASE Part$(Data$, Variable$ + ".status")
            CASE PreDatingEarliest$
                Var$ = Variable$ + ".minimum"
                Term$ = Part$(Data$, Var$)
                NewTerm$ = Part$(NewDate$, "date.minimum")
                IF Term$ <> NewTerm$ THEN
                    Pause
                    ERROR 114
                END IF
                Var$ = Variable$ + ".maximum"
                Change Data$, Var$, Part$(NewDate$, "date.maximum")
            CASE PreDatingComplete$
                Var$ = Variable$ + ".minimum"
                Term$ = Part$(Data$, Var$)
                NewTerm$ = Part$(NewDate$, "date.minimum")
                IF Term$ <> NewTerm$ THEN
                    Pause
                    ERROR 114
                END IF
                Var$ = Variable$ + ".maximum"
                Term$ = Part$(Data$, Var$)
                NewTerm$ = Part$(NewDate$, "date.maximum")
                IF Term$ <> NewTerm$ THEN
                    Pause
                    ERROR 114
                END IF
            CASE PreDatingLatest$
                Var$ = Variable$ + ".maximum"
                Term$ = Part$(Data$, Var$)
                NewTerm$ = Part$(NewDate$, "date.maximum")
                IF Term$ <> NewTerm$ THEN
                    Pause
                    ERROR 114
                END IF
                Var$ = Variable$ + ".minimum"
                Change Data$, Var$, Part$(NewDate$, "date.minimum")
            CASE ELSE
                Change Data$, Variable$, NewDate$
        END SELECT
    END IF
NEXT c%
END IF

IF RIGHT$(Simultaneous$, 2) <> SPACE$(2) THEN
    DateValue$ = SPACE$(SIZEOF$("date"))
    NewDate$ = DateValue$
    FOR c% = 1 TO 2
        Together%(c%) = Moment$(MID$(Simultaneous$, c% + 3, 1))
    
```

```
IF Together%(c%) > 0 THEN
    Variable$ = "data.date[" + LTRIM$(STR$(Together%(c%))) + "]"
    Buffer$ = Part$(Data$, Variable$)
    Var$ = "date.minimum"
    IF Part$(Buffer$, Var$) <> SPACE$(Length%) THEN
        Change DateValue$, Var$, Part$(Buffer$, Var$)
    END IF
    Var$ = "date.maximum"
    IF Part$(Buffer$, Var$) <> SPACE$(Length%) THEN
        Change DateValue$, Var$, Part$(Buffer$, Var$)
    END IF
    NewStatus DateValue$
END IF
NewDate$ = Intersection$(NewDate$, DateValue$)
NEXT c%
FOR c% = 1 TO 2
    IF Together%(z%) > 0 THEN
        Variable$ = "data.date[" + LTRIM$(STR$(Together%(c%))) + "]"
        SELECT CASE Part$(Data$, Variable$ + ".status")
        CASE PreDatingEarliest$
            Var$ = Variable$ + ".minimum"
            Term$ = Part$(Data$, Var$)
            NewTerm$ = Part$(NewDate$, "date.minimum")
            IF Term$ <> NewTerm$ THEN
                Pause
                ERROR 114
            END IF
            Var$ = Variable$ + ".maximum"
            Change Data$, Var$, Part$(NewDate$, "date.maximum")
        CASE PreDatingComplete$
            Var$ = Variable$ + ".minimum"
            Term$ = Part$(Data$, Var$)
            NewTerm$ = Part$(NewDate$, "date.minimum")
            IF Term$ <> NewTerm$ THEN
                Pause
                ERROR 114
            END IF
            Var$ = Variable$ + ".maximum"
            Term$ = Part$(Data$, Var$)
            NewTerm$ = Part$(NewDate$, "date.maximum")
            IF Term$ <> NewTerm$ THEN
                Pause
                ERROR 114
            END IF
        CASE PreDatingLatest$
            Var$ = Variable$ + ".minimum"
            Term$ = Part$(Data$, Var$)
            NewTerm$ = Part$(NewDate$, "date.minimum")
            IF Term$ <> NewTerm$ THEN
                Pause
                ERROR 114
            END IF
            Var$ = Variable$ + ".maximum"
            Change Data$, Var$, Part$(NewDate$, "date.minimum")
        CASE ELSE
            Change Data$, Variable$, NewDate$
        END SELECT
    END IF
```

```

    NEXT c%
END IF

'Save the Intersection Dates at First Entry:
'-----'
IF Simultaneous$ <> SPACE$(5) THEN
    Addition$ = Load$(InputFile%, DataLength%, GIL&, s&)
    FOR c% = 1 TO 6
        Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
        NewDate$ = Part$(Data$, Variable$)
        Change Addition$, Variable$, NewDate$
    NEXT c%
    PUT #InputFile%, s& * DataLength% + 1, Addition$
END IF

'Number of the First Record:
'-----'
ReadIn& = s&
END FUNCTION 'ReadIn& _____'

'=====
SUB RelationsBackwards (Num&)
'=====
' Will generate the backward relations of the current file record
' to other file records.
'
' Handling:
' 6/16/2001 - 12/23/2002:    Norbert Südland
' 3/29/2009 - 10/ 8/2016:    Norbert Südland
' Translation:
' 4/10/2008 - 10/13/2016:    Norbert Südland
'-----'

DIM Buffer$ 'AS STRING
DIM Comparison$
DIM Data$
DIM DateValue$
DIM Name$
DIM Text$
DIM Predecessor$
DIM Relation$
DIM Tol$
DIM Sum$
DIM Summand$
DIM Variable$
DIM c% 'AS INTEGER
DIM f%
DIM Mom1% 'Moment 1
DIM Mom2% 'Moment 2
DIM x%
DIM y%
DIM IntermediateResult& 'AS LONG
DIM Place&

IntermediateResult& = ReadIn&(Num&, Data$)
Name$ = Part$(Data$, "data.name")
x% = POS(0)
y% = CSRLIN

```

```

IF y% = 24 THEN
    PRINT
    y% = 23
END IF

' Consider Predecessors:
'-----'
FOR c% = 1 TO 2

    Place& = Find&(Name$, 1, c% + 1)

    WHILE (Place& <> 0)
        IntermediateResult& = ReadIn&(Place&, Comparison$)

        Text$ = "Backward relations for entry "
        Text$ = Text$ + Part$(Comparison$, "data.name") + " ("
        Text$ = Text$ + LTRIM$(STR$(Place&)) + "/" + LTRIM$(STR$(GIL&))
        Text$ = Text$ + ") are generated."
        LOCATE y%, x%
        ProtocolMessage Text$, 1

        Variable$ = "data.p[" + LTRIM$(STR$(c%)) + "]"
        Predecessor$ = Part$(Comparison$, Variable$)

        IF Name$ = Part$(Predecessor$, "predecessor.name") THEN

            Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment1"))
            Mom1% = Moment%(Buffer$)

            Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment2"))
            Mom2% = Moment%(Buffer$)

            Variable$ = "data.date[" + LTRIM$(STR$(Mom2%)) + "]"
            DateValue$ = Part$(Comparison$, Variable$)

            Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.direct"))
            IF Buffer$ = "N" THEN
                Change DateValue$, "date.minimum", SPACE$(SIZEOF$("term"))
                NewStatus DateValue$
            END IF

            f% = RelationFile%(Mom1%)
            WriteTo f%, RL&(Mom1%), DateValue$, Mom1%, c% + 6, Mom2%, Place&, Num&
        END IF
        Place& = Find&(Name$, Place& + 1, c% + 1)
    WEND
NEXT c%

'Consider Relation Names:
'-----'
FOR c% = 1 TO 2

    Place& = Find&(Name$, 1, c% + 3)

    WHILE (Place& <> 0)
        IntermediateResult& = ReadIn&(Place&, Comparison$)

        LOCATE y%, x%

```

```

PRINT "Backward relations for entry "; Part$(Comparison$, "data.name");
PRINT " ("; LTRIM$(STR$(Place&)); "/" ; LTRIM$(STR$(GIL&)); ") are ";
PRINT "generated."

Variable$ = "data.r[" + LTRIM$(STR$(c%)) + "]"
Relation$ = Part$(Comparison$, Variable$)

IF Name$ = Part$(Relation$, "relation.name") THEN

    Buffer$ = ToUser$(Part$(Relation$, "relation.moment1"))
    Mom1% = Moment%(Buffer$)

    Buffer$ = ToUser$(Part$(Relation$, "relation.moment2"))
    Mom2% = Moment%(Buffer$)

    DateValue$ = Part$(Relation$, "relation.date")

    Tol$ = Part$(Relation$, "relation.tol1")

    Variable$ = "data.date[" + LTRIM$(STR$(Mom2%)) + "]"
    Summand$ = Part$(Comparison$, Variable$)

    Sum$ = Total$(DateValue$, Tol$, Summand$, "-")

    f% = RelationFile%(Mom1%)
    WriteTo f%, RL$(Mom1%), Sum$, Mom1%, c% + 8, Mom2%, Place&, Num&
END IF
Place& = Find$(Name$, Place& + 1, c% + 3)
WEND
NEXT c%
END SUB 'RelationsBackwards _____'

'=====
SUB RelationsForward (Num&)
'=====
' Will generate the forward relations for file record number `Num&` of
' the file `#InputFile%` to all other file records.
'
' The file record is build up in such a way, that first the relative, then
' the absolute datings occur.
' "Relative datings" allow only to fix the order, "absolute datings" enable
' to determine a date.
'
' Handling:
' 6/16/2001 - 1/30/2003: Norbert Südland
' 3/25/2009 - 10/ 8/2016: Norbert Südland
'
' Translation:
' 4/10/2008 - 10/13/2016: Norbert Südland
'-----
DIM Buffer$ 'AS STRING
DIM Comparison$
DIM Data$
DIM DateValue$
DIM Direction$
DIM Duration$
DIM Predecessor$
DIM Relation$

```

```

DIM Sum$
DIM Summand$
DIM Text$
DIM Tol$
DIM Variable$
DIM c%
DIM f%
DIM Mom1%           'moment vor Predecessor
DIM Mom2%           'moment for Name/Event
DIM First%          'data record number of the corresponding first entry
DIM Place%
DIM IntermediateResult%

'Read In Data Record with All Datings and Simultaneous Relations:
'-----'
First% = ReadIn$(Num%, Data$)
Text$ = " Forward relations for entry " + Part$(Data$, "data.name") + " ("
Text$ = Text$ + LTRIM$(STR$(Num%)) + "/" + LTRIM$(STR$(GIL%))
Text$ = Text$ + ") are generated."
ProtocolMessage Text$, 1

'Consider Predecessors:
'-----'
FOR c% = 1 TO 2
    Variable$ = "data.p[" + LTRIM$(STR$(c%)) + "]"

    Predecessor$ = Part$(Data$, Variable$)

    Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment2"))
    Mom2% = Moment$(Buffer$)

    IF Mom2% <> 0 THEN
        Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.moment1"))
        Mom1% = Moment$(Buffer$)
        Place% = Find$(Part$(Predecessor$, "predecessor.name"), 1, 1)
        IF Place% = 0 THEN
            Text$ = "Mentioned predecessor "
            Text$ = Text$ + Part$(Predecessor$, "predecessor.name")
            Text$ = Text$ + " is missing"
            ProtocolMessage Text$, 0
            Pause
            ERROR 108
        ELSE
            IntermediateResult% = ReadIn$(Place%, Comparison$)
        END IF

        Variable$ = "data.date[" + LTRIM$(STR$(Mom1%)) + "]"
        DateValue$ = Part$(Comparison$, Variable$)
        Buffer$ = ToUser$(Part$(Predecessor$, "predecessor.direct"))
        IF Buffer$ = "N" THEN
            Change DateValue$, "date.maximum", SPACE$(SIZEOF$("term"))
            NewStatus DateValue$
        END IF

        f% = RelationFile$(Mom2%)
        WriteTo f%, RL$(Mom2%), DateValue$, Mom2%, c%, Mom1%, Place%, Num%
    END IF

```

NEXT c%

'Consider Relation Names:  
'-----'

FOR c% = 1 TO 2

Variable\$ = "data.r[" + LTRIM\$(STR\$(c%)) + "]"

Relation\$ = Part\$(Data\$, Variable\$)

Buffer\$ = ToUser\$(Part\$(Relation\$, "relation.moment2"))

Mom2% = Moment%(Buffer\$)

IF Mom2% <> 0 THEN

Buffer\$ = ToUser\$(Part\$(Relation\$, "relation.moment1"))

Mom1% = Moment%(Buffer\$)

IF Mom1% <> 0 THEN

Place& = Find\$(Part\$(Relation\$, "relation.name"), 1, 1)

IF Place& = 0 THEN

PRINT "Mentioned relation name ";

PRINT Part\$(Relation\$, "relation.name"); " is missing";

Pause

ERROR 109

ELSE

IntermediateResult& = ReadIn\$(Place&, Comparison\$)

END IF

DateValue\$ = Part\$(Relation\$, "relation.date")

Tol\$ = Part\$(Relation\$, "relation.tol1")

Variable\$ = "data.date[" + LTRIM\$(STR\$(Mom1%)) + "]"

Summand\$ = Part\$(Comparison\$, Variable\$)

Sum\$ = Total\$(DateValue\$, Tol\$, Summand\$, "+")

f% = RelationFile%(Mom2%)

WriteTo f%, RL\$(Mom2%), Sum\$, Mom2%, c% + 2, Mom1%, Place&, Num&

END IF

'Duration:  
'-----'

Duration\$ = Part\$(Relation\$, "relation.duration")

IF Duration\$ <> SPACE\$(SIZEOF("time")) THEN

Tol\$ = Part\$(Relation\$, "relation.tol2")

Variable\$ = "data.date[" + LTRIM\$(STR\$(Mom2%)) + "]"

Summand\$ = Part\$(Data\$, Variable\$)

IF Mom2% > 3 THEN

Direction\$ = "--"

ELSE

Direction\$ = "+"

END IF

Sum\$ = Total\$(Duration\$, Tol\$, Summand\$, Direction\$)

'Here the Dest. `Mom1` is Coupled Closely to the Source `Mom2`:  
'-----'

Mom1% = 7 - Mom2%

f% = RelationFile%(Mom1%)

WriteTo f%, RL\$(Mom1%), Sum\$, Mom1%, c% + 4, Mom2%, Num&, Num&

'-----'  
'Here, simultaneous moments are not yet considered of!'  
'-----'

'Now Source and Destination are Swapped:  
'-----'

Variable\$ = "data.date[" + LTRIM\$(STR\$(Mom1%)) + "]"

Summand\$ = Part\$(Data\$, Variable\$)

```

        IF Mom2% > 3 THEN
            Direction$ = "+"
        ELSE
            Direction$ = "-"
        END IF
        Sum$ = Total$(Duration$, Tol$, Summand$, Direction$)
        f% = RelationFile%(Mom2%)
        WriteTo f%, RL$(Mom2%), Sum$, Mom2%, c% + 4, Mom1%, Num&, Num&
    END IF
END IF
NEXT c%
END SUB 'RelationsForward _____'

'=====
SUB RestSystem (Sign%, Year%, Month%, Day%)
'=====
' Will clarify a row of date numbers to get an unequivocal situation.
'
' Handling:
' 8/ 4/2001 - 3/27/2003:    Norbert Südland
' Translation:
' 11/19/2007:                Norbert Südland
'-----
DIM Days&      'AS LONG

Days& = Sign% * Year% * 360& + Month% * 30& + Day%
Year% = INT(Days& / 360&)
Days& = Days& - Year% * 360&
Month% = INT(Days& / 30&)
Day% = Days& - Month% * 30&
IF Year% < 0 THEN
    Year% = -Year%
    Sign% = -1
ELSE
    Sign% = 1
END IF
IF ABS(Year%) >= 10000 THEN ERROR 106
END SUB 'RestSystem _____'

'=====
SUB Result (File%, FL&, Default$, Comparison$, Target%, Source%, Num&)
'=====
' Will determine the difference of a date to be changed by the intersection
' of all relations, that already exist by evaluation of the whole data.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 4/12/2008:                Norbert Südland, Aalen
'-----
DIM IntermediateResult$      'AS STRING
DIM Sum$
DIM Term$
DIM Length%                  'AS INTEGER
DIM s1%, y1%, m1%, d1%
DIM s2%, y2%, m2%, d2%
```

```

Length% = SIZEOF%("date")
IntermediateResult$ = SPACE$(Length%)
IF Part$(Comparison$, "array.date") <> SPACE$(Length%) THEN

    'Calculate Difference:
    '-----'
    Sum$ = Part$(Default$, "array.date")
    IF Sum$ = SPACE$(Length%) THEN
        Term$ = NumberToTerm$(1, 1, 1, 1)          'Dummy at 1.Y 1.M 1.D
        Change Sum$, "date.minimum", Term$
        Change Sum$, "date.maximum", Term$
    END IF
    Add Sum$, "U", "-", Part$(Comparison$, "array.date"), 1, 1, "-"

    'Absolute Value of the Difference:
    '-----'
    TermToNumber Part$(Sum$, "date.minimum"), s1%, y1%, m1%, d1%
    TermToNumber Part$(Sum$, "date.maximum"), s2%, y2%, m2%, d2%
    IF s1% = -1 THEN
        Reverse s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, "-"
    END IF
    Term$ = NumberToTerm$(s1%, y1%, m1%, d1%)
    Change IntermediateResult$, "date.minimum", Term$
    Change IntermediateResult$, "date.status", " "
    IF s2% = -1 THEN
        Reverse s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, "-"
    END IF
    Term$ = NumberToTerm$(s2%, y2%, m2%, d2%)
    Change IntermediateResult$, "date.maximum", Term$

    'Elucidate the Result:
    '-----'
    IF y1% > 0 THEN
        Change IntermediateResult$, "date.minimum.ys", "Y"
    END IF
    IF m1% > 0 THEN
        Change IntermediateResult$, "date.minimum.ms", "M"
    END IF
    IF d1% > 0 THEN
        Change IntermediateResult$, "date.minimum.ds", "D"
    END IF
    IF y2% > 0 THEN
        Change IntermediateResult$, "date.maximum.ys", "Y"
    END IF
    IF m2% > 0 THEN
        Change IntermediateResult$, "date.maximum.ms", "M"
    END IF
    IF d2% > 0 THEN
        Change IntermediateResult$, "date.maximum.ds", "D"
    END IF
END IF

'Save the Result:
'-----'
WriteTo File%, FL&, IntermediateResult$, 0, 0, Source%, 0, 0
IntermediateResult$ = Part$(Comparison$, "array.date")
WriteTo File%, FL&, IntermediateResult$, Target%, 0, Source%, 0, Num&

```

```

END SUB 'Result _____'

'=====
SUB Reverse (s1%, y1%, m1%, d1%, s2%, y2%, m2%, d2%, Direction$)
'=====
' If `Direction$` = "-", then the date `DateValue$` will be mirrored.
' `Direction$` = "+" causes no effect.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001: Norbert Südland, Munich
' Translation:
' 11/19/2007 - 4/12/2008: Norbert Südland, Aalen
'-----

DIM c% 'AS INTEGER
DIM s%
DIM y%
DIM m%
DIM d%

IF Direction$ = "-" THEN
FOR c% = 1 TO 2
IF c% = 1 THEN
s% = s1%: y% = y1%: m% = m1%: d% = d1%
ELSE
s% = s2%: y% = y2%: m% = m2%: d% = d2%
END IF
s% = -s%
m% = -m%
d% = -d%
RestSystem s%, y%, m%, d%
IF c% = 1 THEN
s1% = s%: y1% = y%: m1% = m%: d1% = d%
ELSE
s2% = s%: y2% = y%: m2% = m%: d2% = d%
END IF
NEXT c%
END IF
END SUB 'Reverse _____'

'=====
SUB SaveDate (Position&, Data$)
'=====
' Will save `Data$` as `Position&`th position in `#InputFile%`, and will
' check at the datings, that a mark sign of pre-dating and its content
' will not be deleted.
' The `FirstEntry$` to a certain name will get all datings, at the
' following entries of the same name the datings will be deleted.
'
' Handling:
' 8/ 4/2001 - 1/30/2003: Norbert Südland
' Translation:
' 4/12/2008: Norbert Südland
'-----

DIM Cutting$ 'AS STRING
DIM FirstEntry$
DIM Original$

```

```

DIM Variable$
DIM c%           'AS INTEGER
DIM FirstPosition& 'AS LONG

'Load Data Records to be Changed:
'-----'
Original$ = Load$(InputFile%, DataLength%, GIL&, Position&)
FirstPosition& = Find&(Part$(Original$, "data.name"), 1, 1)

'The Datings are Changed Only:
'-----'
IF FirstPosition& = Position& THEN
  FOR c% = 1 TO 6
    Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
    SELECT CASE Part$(Original$, Variable$ + ".status")
      CASE PreDatingEarliest$ " "
        Cutting$ = Part$(Data$, Variable$ + ".maximum")
        Change Original$, Variable$ + ".maximum", Cutting$
      CASE PreDatingComplete$ " "
      CASE PreDatingLatest$ " "
        Cutting$ = Part$(Original$, Variable$ + ".minimum")
        Change Original$, Variable$ + ".minimum", Cutting$
      CASE ELSE
        Change Original$, Variable$, Part$(Data$, Variable$)
    END SELECT
    IF Contradiction%(Part$(Original$, Variable$)) = 1 THEN
      Pause
      ERROR 113
    END IF
  NEXT c%
ELSE
  FirstEntry$ = Load$(InputFile%, DataLength%, GIL&, FirstPosition&)
  FOR c% = 1 TO 6
    Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"

    'Save Datings at `FirstEntry$`:
    '-----'
    SELECT CASE Part$(FirstEntry$, Variable$ + ".status")
      CASE PreDatingEarliest$ " "
        Cutting$ = Part$(Data$, Variable$ + ".maximum")
        Change FirstEntry$, Variable$ + ".maximum", Cutting$
      CASE PreDatingComplete$ " "
      CASE PreDatingLatest$ " "
        Cutting$ = Part$(Original$, Variable$ + ".minimum")
        Change FirstEntry$, Variable$ + ".minimum", Cutting$
      CASE ELSE
        Change FirstEntry$, Variable$, Part$(Data$, Variable$)
    END SELECT
    IF Contradiction%(Part$(FirstEntry$, Variable$)) = 1 THEN
      Pause
      ERROR 113
    END IF

    'Delete Datings at Following Entries as Much as Possible:
    '-----'
    SELECT CASE Part$(Original$, Variable$ + ".status")
      CASE PreDatingEarliest$ " "

```

```

        Change Original$, Variable$ + ".maximum", SPACE$(SIZEOF%("term"))
CASE PreDatingComplete$      ' "■"
CASE PreDatingLatest$        ' "■"
        Change Original$, Variable$ + ".minimum", SPACE$(SIZEOF%("term"))
CASE ELSE
        Change Original$, Variable$, SPACE$(SIZEOF%("date"))
END SELECT
NEXT c%
PUT #InputFile%, FirstPosition& * DataLength% + 1, FirstEntry$
END IF

PUT #InputFile%, Position& * DataLength% + 1, Original$
END SUB 'SaveDate _____'

'=====
FUNCTION SequenceOrder& (Mode%, ListLength&, ListElement&)
'=====
'Gives in 4 modes a sequence order from 1 to list length.
'
'Handling: 10/ 1/2016 - 3/15/2017 Norbert Suedland, D-73431 Aalen, Germany
'-----
DIM Value&

'Parameter Check:
'-----
IF ListElement& < 1 OR ListElement& > ListLength& THEN
PRINT "Programming error: list element ="; ListElement&;
PRINT "is valid only between 1 and"; ListLength&; "."
Pause
Value& = 0
ELSE
SELECT CASE Mode%
CASE 1
Value& = ListElement&
CASE 2
Value& = ListLength& + 1 - ListElement&
CASE 3
Value& = (-1) ^ (ListElement& MOD 2)
Value& = Value& * INT(ListElement& / 2)
Value& = Value& + INT((ListLength& + 1) / 2)
CASE 4
Value& = (-1) ^ ((ListLength& + 1 - ListElement&) MOD 2)
Value& = Value& * INT((ListLength& + 1 - ListElement&) / 2)
Value& = Value& + INT((ListLength& + 1) / 2)
CASE ELSE
PRINT "Programming error: series sequence mode"; Mode%;
PRINT "was called instead of 1 to 4."
Pause
Value& = 0
END SELECT
END IF

SequenceOrder& = Value&
END FUNCTION 'SequenceOrder& _____'

'=====

```

```

SUB Shift (RelationFile%(), RL&(), Source%, Target%)
'=====
' Will shift a date from `Source%` to `Target%`.
' This feature is especially needed for simultaneous moments.
'
' Handling:
' 8/ 4/2001 - 1/ 8/2003:    Norbert Südland
' Translation:
' 4/12/2008:                Norbert Südland
'-----
DIM Array$      'AS STRING
DIM L%          'AS INTEGER
DIM c&          'AS LONG

L% = SIZEOF%("array")
FOR c& = 1 TO RL&(Source%)
    Array$ = Load$(RelationFile%(Source%), ArrayLength%, RL&(Source%), c&)
    RL&(Target%) = RL&(Target%) + 1
    PUT #RelationFile%(Target%), RL&(Target%) * L% + 1, Array$
NEXT c&
RL&(Source%) = 1
END SUB 'Shift
'-----

'=====
FUNCTION SimultaneousnessCheck$ (Text$)
'=====
' Will correct `Text$` to give sensible entries of simultaneous moments.
' `Text$` must own the length of 5 characters!
'
' Handling:
' 9/ 4/2001 - 12/24/2002:    Norbert Südland
' Translation:
' 11/19/2007 - 11/22/2007:    Norbert Südland
'-----
DIM a$(2)      'AS STRING
DIM empty$
DIM c%         'AS INTEGER
DIM ly%
DIM x%(2)
DIM m%

FOR c% = 1 TO 2
    a$(c%) = SPACE$(6)
    x%(c%) = 0
    FOR ly% = 1 + (c% - 1) * 3 TO 3 + (c% - 1) * 2
        m% = Moment%(MID$(Text$, ly%, 1))
        IF m% <> 0 THEN
            IF MID$(a$(c%), m%, 1) = " " THEN
                MID$(a$(c%), m%, 1) = MID$(Text$, ly%, 1)
                x%(c%) = x%(c%) + 1
            END IF
        END IF
    NEXT ly%

'Has Just a Single Character been Typed in?
'-----

```

```

    IF x%(c%) < 2 THEN
        a$(c%) = SPACE$(6)
        x%(c%) = 0
    END IF
NEXT c%

'Reduction to Fitting Size:
'-----
FOR c% = 1 TO 2
    a$(c%) = RTRIM$(LTRIM$(a$(c%)))
    IF LEN(a$(c%)) > 0 THEN
        IF MID$(a$(c%), 2, 1) = " " THEN 'at least a further character rhs!
            a$(c%) = LEFT$(a$(c%), 1) + LTRIM$(RIGHT$(a$(c%), LEN(a$(c%)) - 1))
        END IF
        IF LEN(a$(c%)) > 2 THEN
            IF MID$(a$(c%), 3, 1) = " " THEN
                a$(c%) = LEFT$(a$(c%), 2) + LTRIM$(RIGHT$(a$(c%), LEN(a$(c%)) - 2))
            END IF
        END IF
    END IF
    a$(c%) = a$(c%) + SPACE$((4 - c%) - LEN(a$(c%)))
NEXT c%

```

'Comparison of the Combinations:

'-----

'There are 8 combinations of moments possible in principle:

'"\*B $\alpha$ ", " $\Omega$ E+", "\*B", "B $\alpha$ ", " $\alpha$ E", " $\Omega$ E", " $\Omega$ +", "E+".

'The following pairs of combinations are valid:

		*B	B $\alpha$	$\alpha$ E	$\Omega$ E	$\Omega$ +	E+
*B		O	O	X	X	X	-
B $\alpha$		O	O	-	-	X	-
$\alpha$ E		X	-	O	-	X	-
$\Omega$ E		X	-	-	O	O	O
$\Omega$ +		X	X	X	O	O	O
E+		-	-	-	O	O	O

'The concrete symbols mean:

' O possibility of reduction to something sensible  
' X valid combination  
' - invalid combination or reduction

```

empty$ = SPACE$(2)
SELECT CASE a$(1)                                'Changes of order are rather perturbing!
CASE "*B $\alpha$ ", " $\Omega$ E+"
    a$(2) = empty$
CASE "*B" + SPACE$(1)
    SELECT CASE a$(2)
    CASE "B $\alpha$ "
        a$(1) = "*B $\alpha$ ": a$(2) = empty$
    CASE " $\alpha$ E", " $\Omega$ E", " $\Omega$ +"
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE "B $\alpha$ " + SPACE$(1)
    SELECT CASE a$(2)
    CASE "*B"

```

```

        a$(1) = "*Bα": a$(2) = empty$
    CASE "Q+"
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE "αE" + SPACE$(1)
    SELECT CASE a$(2)
    CASE "*B"
        a$(1) = "*B" + SPACE$(1): a$(2) = "αE"
    CASE "Q+"
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE "QE" + SPACE$(1)
    SELECT CASE a$(2)
    CASE "*B"
        a$(1) = "*B" + SPACE$(1): a$(2) = "QE"
    CASE "Q+", "E+"
        a$(1) = "QE+": a$(2) = empty$
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE "Q+" + SPACE$(1)
    SELECT CASE a$(2)
    CASE "*B"
        a$(1) = "*B" + SPACE$(1): a$(2) = "Q+"
    CASE "Bα"
        a$(1) = "Bα" + SPACE$(1): a$(2) = "Q+"
    CASE "αE"
        a$(1) = "αE" + SPACE$(1): a$(2) = "Q+"
    CASE "QE", "E+"
        a$(1) = "QE+": a$(2) = empty$
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE "E+" + SPACE$(1)
    SELECT CASE a$(2)
    CASE "QE", "Q+"
        a$(1) = "QE+": a$(2) = empty$
    CASE ELSE
        a$(2) = empty$
    END SELECT
CASE ELSE
    SELECT CASE a$(2)
    CASE "*B", "Bα", "αE", "QE", "Q+", "E+"
        a$(1) = a$(2) + SPACE$(1): a$(2) = empty$
    CASE ELSE
        a$(2) = empty$: a$(1) = a$(2) + SPACE$(1)
    END SELECT
END SELECT

'Result:
'-----'

SimultaneousnessCheck$ = a$(1) + a$(2)
END FUNCTION 'SimultaneousnessCheck$ _____'

'=====
FUNCTION SIZEOF% (StructureName$)

```

```

'=====
' Will give the length of the data structure being named `StructureName$`.
'
' Handling:
' 8/ 4/2001 - 2/12/2003:    Norbert Südland
' Check:
' 2/12/2003:                Norbert Südland
' Translation:
' 4/14/2008:                Norbert Südland
'-----
DIM StructurePosition%      'AS INTEGER
DIM StructureLength%        'AS INTEGER

StructurePosition% = QuickPosition%(GVName$(), StructureName$)
IF StructurePosition% = 0 THEN
    Pause
    StructureLength% = 0      'Name not found: Spelling mistake?
ELSE

    'Read Saved Structure Length:
    '-----
    StructureLength% = GVLength%(StructurePosition%)

    IF StructureLength% <= 0 THEN
        Pause
        ERROR 190            'Non-positive structure length has been saved!
    END IF
END IF

SIZEOF% = StructureLength%
END FUNCTION '_____ Ende von `SIZEOF%` _____

'=====
SUB SortInto (Name$, Position&, NameType%)
'=====
' Will construct a list entry and enlarge the name file concerning
' `NameType%` by `Name$` and `Position&`.
' The name file will be sorted alpha-numerically.
'
' Handling:
' 12/18/2002 - 1/ 5/2003:    Norbert Südland
' Check:
'
' Translation:
' 4/14/2008:                Norbert Südland
'-----
DIM Buffer$      'AS STRING
DIM List$
DIM L%          'AS INTEGER
DIM Start&      'AS LONG
DIM midth&
DIM Finish&
DIM Place&

L% = SIZEOF%("list")
IF LTRIM$(Name$) <> "" THEN

```

```

'Build Up a New Entry:
'-----'
List$ = SPACE$(L%)
Change List$, "list.name", Name$
Change List$, "list.number", STR$(Position&)
Change List$, "list.end", CHR$(179) + CHR$(10) + CHR$(13)

'Find the Sorting Position:
'-----'
Start& = 1
midth& = 1 'Default of an empty file.
Finish& = NL&(NameType%)
WHILE Start& <= Finish&
    midth& = CLNG((Start& + Finish&) / 2)
    Buffer$ = Load$(Names%(NameType%), L%, NL&(NameType%), midth&)
    SELECT CASE List$
    CASE IS = Buffer$
        Pause 'Not yet occurred!
        ERROR 120 'Double name entry: Programming mistake?
    CASE IS < Buffer$
        Finish& = midth& - 1
    CASE IS > Buffer$
        Start& = midth& + 1
    END SELECT
WEND

'Sort In the New Entry:
'-----'
IF Start& > midth& THEN
    midth& = Start& 'New Entry Will be Sorted Behind `Midth&`.
END IF
FOR Place& = NL&(NameType%) TO midth& STEP -1
    Buffer$ = Load$(Names%(NameType%), L%, NL&(NameType%), Place&)
    PUT #Names%(NameType%), (Place& + 1) * L% + 1, Buffer$
NEXT Place&
PUT Names%(NameType%), midth& * L% + 1, List$
NL&(NameType%) = NL&(NameType%) + 1
END IF
END SUB 'SortInto _____'

'=====
FUNCTION StartCalculation$ (WorkingPlace$, New%, Rest%, RL&, CM%)
'=====
' Will build up 5 lists of the names of double entries and their occurrence.
' The lists are sorted alpha-numerically.
'
' New Calculation:
' All datings except the pre-datings are deleted.
' Pre-datings are put to the `Rest%` file.
'
' No New Calculation:
' All datings are checked for correctness. At least ONE pre-dating must
' exist. Deviations are put to the `Rest%` file.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001: Norbert Südland, Munich
' 12/18/2002 - 3/18/2003: Norbert Südland, Aalen

```

```

' 10/ 7/2016 - 3/18/2017:      Norbert Südland, Aalen
' Translation:
' 4/14/2008 - 4/14/2025:      Norbert Südland, Aalen
'-----'
DIM Data$                'AS STRING
DIM Name$
DIM Variable$
DIM DateValue$
DIM Text$
DIM Question$
DIM Terminate$

DIM Buffer$
DIM Copy$
DIM Array$

DIM x%                   'AS INTEGER
DIM y%
DIM c%
DIM Length%

DIM Found%
DIM s%
DIM W%
DIM L%

DIM Place&               'AS LONG
DIM Position&
DIM PreDating&
DIM counter&

'Preparation:
'-----'
CLS
PreDating& = 0
Terminate$ = "N"
x% = POS(0)
y% = CSRLIN
Length% = SIZEOF%("date")
L% = SIZEOF%("remainder")
Text$ = "Calculation method: counting mode number" + STR$(CM%) + "."
ProtocolMessage Text$, 1
ProtocolMessage "", 1

FOR counter& = 1 TO GIL&

  'Read Entry:
  '-----'
  Place& = SequenceOrder&(CM%, GIL&, counter&)
  Data$ = Load$(InputFile%, DataLength%, GIL&, Place&)
  Name$ = Part$(Data$, "data.name")

  'Terminate Calculation?
  '-----'
  Question$ = "Terminate calculation?"
  Text$ = "The calculation was terminated."
  Terminate$ = YesNoQuestion$(Question$, Text$)

```

```

IF Terminate$ = "Y" THEN GOTO EndOfStartCalculation

'Screen Message:
'-----'
Text$ = "Entry " + Name$ + " (" + LTRIM$(STR$(Place&)) + "/"
Text$ = Text$ + LTRIM$(STR$(GIL&)) + ") is checked."
Text$ = Text$ + SPACE$(LEN(STR$(GIL&)))
LOCATE y%, x%
ProtocolMessage Text$, 1
Check (Data$)

'List Names Alpha-numerically:
'-----'
SortInto Name$, Place&, 1
Name$ = Part$(Data$, "data.p[1].name")
IF LTRIM$(Name$) <> "" THEN
    SortInto Name$, Place&, 2
END IF
Name$ = Part$(Data$, "data.p[2].name")
IF LTRIM$(Name$) <> "" THEN
    SortInto Name$, Place&, 3
END IF
Name$ = Part$(Data$, "data.r[1].name")
IF LTRIM$(Name$) <> "" THEN
    SortInto Name$, Place&, 4
END IF
Name$ = Part$(Data$, "data.r[2].name")
IF LTRIM$(Name$) <> "" THEN
    SortInto Name$, Place&, 5
END IF

'Deal with Pre-datings:
'-----'
FOR c% = 1 TO 6
    Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
    DateValue$ = Part$(Data$, Variable$)
    IF New% = 1 THEN

        'Delete All Besides Pre-datings:
        '-----'
        SELECT CASE Part$(DateValue$, "date.status")
        CASE PreDatingEarliest$ "■"
            Found% = RIGHT
            Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "].maximum"
            Change Data$, Variable$, SPACE$(SIZEOF$("term"))
        CASE PreDatingComplete$ "■"
            Found% = RIGHT
        CASE PreDatingLatest$ "■"
            Found% = RIGHT
            Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "].minimum"
            Change Data$, Variable$, SPACE$(SIZEOF$("term"))
        CASE ELSE
            Found% = WRONG
            Variable$ = "data.date[" + LTRIM$(STR$(c%)) + "]"
            Change Data$, Variable$, SPACE$(Length%)
        END SELECT

        IF Found% THEN

```

```

'-----'
' Hint: The pre-dating can also occur in file entires, '
'       which are located not at the first position. '
'-----'

PreDating& = Place&

'Complete the List of Calculation Orders:
'-----'

Buffer$ = SPACE$(L%)
RL& = RL& + 1
Change Buffer$, "remainder.position", STR$(RL&)
Change Buffer$, "remainder.number", STR$(Place&)
Change Buffer$, "remainder.moment", Symbol$(c%)
Change Buffer$, "remainder.name", NameRegister$(Place&)
Change Buffer$, "remainder.date", DateValue$
Change Buffer$, "remainder.end", "|" + CHR$(13) + CHR$(10)
PUT #Rest%, RL& * L% + 1, Buffer$
Text$ = DateValue$ + " " + STR$(RL&)
Text$ = Text$ + RIGHT$(OrdinaryNumber$(RL& MOD 100), 2)
Text$ = Text$ + " position to " + ToUser$(Symbol$(c%)) + " "
Text$ = Text$ + NameRegister$(Place&) + STR$(Place&)
ProtocolMessage Text$, 1

'-----'
' In the list of calculation orders are mentioned the numbers '
' of the first entries of a dating to be checked. '
'-----'

END IF
ELSE

'Seek Pre-dating:
'-----'

SELECT CASE Part$(DateValue$, "date.status")
CASE PreDatingEarliest$
    Found% = RIGHT
CASE PreDatingComplete$
    Found% = RIGHT
CASE PreDatingLatest$
    Found% = RIGHT
CASE ELSE
    Found% = WRONG
END SELECT
IF Found% THEN
    PreDating& = Place&
    c% = 6
END IF
END IF
NEXT c%
IF New% = 1 THEN
    SaveDate Place&, Data$
END IF
NEXT counter&
IF PreDating& = 0 THEN
    'Pause
    ERROR 101
END IF

'Close the Name Files, and Open Them Again for Reading:

```

```

'-----'
L% = SIZEOF("list")
FOR c% = 1 TO 5
    CLOSE #Names%(c%)
    OPEN NameFile$(c%) FOR BINARY ACCESS READ AS #Names%(c%)
NEXT c%

IF New% = 0 THEN
    FOR counter% = 1 TO GIL%
        Place% = SequenceOrder%(CM%, GIL%, counter%)
        Name$ = NameRegister$(Place%)
        IF Place% = Find$(Name$, 1, 1) THEN

            'Terminate Calculation?
            '-----'
            Question$ = "Terminate calculation?"
            Text$ = "The calculation was terminated."
            Terminate$ = YesNoQuestion$(Question$, Text$)
            IF Terminate$ = "Y" THEN GOTO EndOfStartCalculation

            'Message:
            '-----'
            CLS
            Text$ = "Relation for entry " + Name$ + " (" + LTRIM$(STR$(Place%))
            Text$ = Text$ + "/" + LTRIM$(STR$(GIL%)) + ") is checked."
            Text$ = Text$ + SPACE$(LEN(STR$(GIL%)))
            LOCATE y%, x%
            ProtocolMessage Text$, 1
            LOCATE y% + 1, x%

            'Check the Relations:
            '-----'
            Connection Place%

            'Enable the Check of the Calculations:
            '-----'
            'FOR c% = 1 TO 6
            '    CLOSE #RelationFile%(c%)
            'NEXT c%
            '-----'
            'Here other editors can have access to the results:
            '-----'
            'SHELL "C:\BIBEL\UED.EXE " + WorkingPlace$ + "*.RL*"
            'FOR c% = 1 TO 6
            '    Buffer$ = RelationFileName$(c%)
            '    OPEN Buffer$ FOR BINARY ACCESS READ WRITE AS #RelationFile%(c%)
            'NEXT c%
            'Pause

            FOR c% = 1 TO 6
                IF RL%(c%) > 1 THEN
                    FOR Position% = 0% TO RL%(c%)
                        GET #RelationFile%(c%), Position% * ArrayLength% + 1, Array$
                        Text$ = LEFT$(Array$, ArrayLength% - 2)
                        ProtocolMessage Text$, 1
                    NEXT Position%
                END IF
                Array$ = Load$(RelationFile%(c%), ArrayLength%, RL%(c%), RL%(c%))

```

```

        IF Part$(Array$, "array.date") <> SPACE$(SIZEOF%("date")) THEN
            IF RL&(c%) = 1 THEN
                Text$ = "Dating without any connection:"
                ProtocolMessage Text$, 1
                ProtocolMessage Array$, 1
                Pause
                ERROR 112
            ELSE
                Evaluate Rest%, RL&, Place&, c%, CM%
            END IF
            ProtocolMessage "", 1
        END IF
    NEXT c%
END IF
NEXT counter&
IF RL& = 0 THEN
    Text$ = "All data are calculated correctly."
    ProtocolMessage Text$, 1
END IF
END IF

'=====
EndOfStartCalculation:
'=====
    StartCalculation$ = Terminate$
END FUNCTION 'StartCalculation$ _____

'=====
FUNCTION STRLEN% (Text$, EndCharacter$)
'=====
' Will find the string length of `Text$` until to the first occurrence of
' `EndCharacter$`. like being usual within the C programming language.
' If `EndCharacter$` does not occur, the string length `LEN(Text$)` is
' returned.
'
'
' Handling:
' 8/ 4/2001:      Norbert Südland, Munich
' Check:
' 8/ 4/2001:      Norbert Südland, Munich
' Translation:
' 4/14/2008:      Norbert Südland, Aalen
'-----
DIM IntermediateResult%      'AS INTEGER

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

STRLEN% = IntermediateResult%
END FUNCTION 'STRLEN% _____

'=====
FUNCTION Symbol$ (MomentNumber%)
'=====

```

```

' Will give the symbol belonging to `MomentNumber%` for output.
'
' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' 1/15/2003:                Norbert Südland, Aalen
' Translation:
' 4/14/2008:                Norbert Südland, Aalen
'-----
DIM SymbolText$          'AS STRING

SELECT CASE MomentNumber%
CASE 1
    SymbolText$ = "*"
CASE 2
    SymbolText$ = "B"
CASE 3
    SymbolText$ = "α"      'alpha
CASE 4
    SymbolText$ = "Ω"      'Omega
CASE 5
    SymbolText$ = "E"
CASE 6
    SymbolText$ = "+"
CASE ELSE
    SymbolText$ = " "
END SELECT

Symbol$ = SymbolText$
END FUNCTION 'Symbol$ _____

'=====
SUB TermToNumber (Term$, Sign%, Year%, Month%, Day%)
'=====
' Will convert a `Term$` to a row of integer numbers.
'
' Handlling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 4/14/2008:                Norbert Südland, Aalen
'-----
IF Part$(Term$, "term.sign") = "-" THEN
    Sign% = -1
ELSE
    Sign% = 1
END IF
Year% = VAL(Part$(Term$, "term.year"))
Month% = VAL(Part$(Term$, "term.month"))
Day% = VAL(Part$(Term$, "term.day"))
END SUB 'TermToNumber _____

'=====
SUB TimeShift (s1%, y1%, m1%, d1%, Status$, s2%, y2%, m2%, d2%, Direction$)
'=====
' Will shift dates with negative year number to plain numbers,
' thus there will be a number zero in date calculation.
' Usually date numbers are ordinary numbers.
' `Direction$` = "-" will change to ordinary numbers, "+" will reverse this.

```

```

'
' Handling:
' 8/ 4/2001 - 9/ 4/2001: Norbert Südland, Munich
' Translation:
' 4/14/2008: Norbert Südland, Aalen
'-----
DIM Start% 'AS INTEGER
DIM Finish%
DIM s%, y%, m%, d%
DIM c%

SELECT CASE Status$
CASE " "
    Start% = 1: Finish% = 2
CASE ">"
    Start% = 1: Finish% = 1
CASE "<"
    Start% = 2: Finish% = 2
END SELECT
FOR c% = Start% TO Finish%
    IF c% = 1 THEN
        s% = s1%: y% = y1%: m% = m1%: d% = d1%
    ELSE
        s% = s2%: y% = y2%: m% = m2%: d% = d2%
    END IF
    IF Direction$ = "+" THEN
        RestSystem s%, y%, m%, d%
        IF s% = 1 THEN
            y% = y% + 1
        END IF
        m% = m% + 1
        d% = d% + 1
    ELSE
        IF s% = 1 THEN
            IF y% < 1 THEN
                Pause
                ERROR 110
            END IF
            y% = y% - 1
        END IF
        m% = m% - 1
        d% = d% - 1
        RestSystem s%, y%, m%, d%
    END IF
    IF c% = 1 THEN
        s1% = s%: y1% = y%: m1% = m%: d1% = d%
    ELSE
        s2% = s%: y2% = y%: m2% = m%: d2% = d%
    END IF
NEXT c%
END SUB 'TimeShift

'=====
SUB TimeToNumber (Instant$, Year%, Month%, Day%)
'=====
' Will change a date string to numbers.
'

```

```

' Handling:
' 8/ 4/2001 - 9/ 4/2001:    Norbert Südland, Munich
' Translation:
' 4/14/2008:                Norbert Südland, Aalen
' -----
Year% = VAL(Part$(Instant$, "time.year"))
Month% = VAL(Part$(Instant$, "time.month"))
Day% = VAL(Part$(Instant$, "time.day"))
END SUB 'TimeToNumber _____

' =====
' FUNCTION ToFile$ (Symbol$)
' =====
' Will change 'Symbol$' to an international code within the data file.
' This function causes also old data files to run correctly.
'
' Handling:
' 11/20/2007 - 11/22/2007:    Norbert Südland
' -----

DIM Buffer$
DIM c%

Buffer$ = SPACE$(LEN(Symbol$))
FOR c% = 1 TO LEN(Symbol$)
    SELECT CASE MID$(Symbol$, c%, 1)
        CASE "J", "Y"          'Yes
            MID$(Buffer$, c%, 1) = "1"
        CASE "N"              'No
            MID$(Buffer$, c%, 1) = "0"
        CASE "*"              'Birth
            MID$(Buffer$, c%, 1) = "{"
        CASE "A", "B"         'Begin 1st Periode
            MID$(Buffer$, c%, 1) = "["
        CASE "W", CHR$(224)    'Begin 2nd Periode
            MID$(Buffer$, c%, 1) = "("
        CASE "X", CHR$(234)    'End 2nd Periode
            MID$(Buffer$, c%, 1) = ")"
        CASE "E"              'End 1st Periode
            MID$(Buffer$, c%, 1) = "]"
        CASE "+"              'Death
            MID$(Buffer$, c%, 1) = "}"
        CASE ELSE
            MID$(Buffer$, c%, 1) = SPACE$(1)
    END SELECT
NEXT c%

ToFile$ = Buffer$
END FUNCTION 'ToFile$ _____

' =====
' FUNCTION Total$ (WorkingTime$, Tol$, DateValue$, Direction$)
' =====
' Adding of two datings by considering of the tolerances.
'
' Handling:
' 6/16/2001 - 9/ 4/2001      Norbert Südland, Munich
' Translation:

```

```

' 4/14/2008:                      Norbert Südland, Aalen
'-----'
DIM Tolerance$      'AS STRING
DIM IntermediateResult$
DIM Term$
DIM Length%         'AS INTEGER
DIM s%, y%, m%, d%

'Preparation:
'-----'
Length% = SIZEOF%("date")
Tolerance$ = SPACE$(Length%)
IntermediateResult$ = SPACE$(Length%)

'Does a Date Exist?
'-----'
IF DateValue$ <> Tolerance$ THEN

    'Calculate Tolerance:
    '-----'
    TimeToNumber WorkingTime$, y%, m%, d%
    IF Tol$ = "<" THEN
        IF m% <> 0 THEN y% = y% - 1
        IF d% <> 0 THEN m% = m% - 1
    END IF

    IF d% <> 0 THEN
        Change Tolerance$, "date.minimum.day", "-1"
        Change Tolerance$, "date.minimum.ds", "."
        Change Tolerance$, "date.maximum.day", "1"
        Change Tolerance$, "date.maximum.ds", "."
    ELSEIF m% <> 0 THEN
        Change Tolerance$, "date.minimum.month", "-1"
        Change Tolerance$, "date.minimum.ms", "."
        Change Tolerance$, "date.maximum.month", "1"
        Change Tolerance$, "date.maximum.ms", "."
    ELSE
        Change Tolerance$, "date.minimum.sign", "-"
        Change Tolerance$, "date.minimum.year", "1"
        Change Tolerance$, "date.minimum.ys", "."
        Change Tolerance$, "date.maximum.sign", " "
        Change Tolerance$, "date.maximum.year", "1"
        Change Tolerance$, "date.maximum.ys", "."
    END IF

    'Consider Tolerance Form:
    '-----'
    Length% = SIZEOF%("term")
    SELECT CASE Tol$
    CASE "="
        'no tolerance
        Change Tolerance$, "date.minimum", SPACE$(Length%)
        Change Tolerance$, "date.maximum", SPACE$(Length%)
    CASE "<"
        'ordinary number
        IF Direction$ = "+" THEN
            Change Tolerance$, "date.maximum", SPACE$(Length%)
        ELSE
            Change Tolerance$, "date.minimum", SPACE$(Length%)
        END IF
    END SELECT

```

```

    END IF
CASE ">" 'Persian ordinary number
    IF Direction$ = "+" THEN
        Change Tolerance$, "date.minimum", SPACE$(Length%)
    ELSE
        Change Tolerance$, "date.maximum", SPACE$(Length%)
    END IF
END SELECT

'Add the Time:
'-----'
Term$ = NumberToTerm$(1, y%, m%, d%)

Change IntermediateResult$, "date.minimum", Term$
Change IntermediateResult$, "date.maximum", Term$

Add Tolerance$, "U", Direction$, IntermediateResult$, 0, 0, " "
IntermediateResult$ = Tolerance$

Add IntermediateResult$, ">", "+", DateValue$, 0, 1, " "
END IF

Total$ = IntermediateResult$
END FUNCTION 'Total$ _____

'=====
FUNCTION ToUser$ (Symbol$)
'=====
' Will transform the international 'Symbol$' code to an English user code.
' This function will work correctly with an old data file, too.
'
' Handling:
' 11/20/2007 - 11/22/2007:    Norbert Südland
'-----'
DIM Buffer$
DIM c%

Buffer$ = SPACE$(LEN(Symbol$))
FOR c% = 1 TO LEN(Symbol$)
    SELECT CASE MID$(Symbol$, c%, 1)
    CASE "1", "J", "Y"
        MID$(Buffer$, c%, 1) = "Y" 'Yes
    CASE "0", "N"
        MID$(Buffer$, c%, 1) = "N" 'No
    CASE "{", "*"
        MID$(Buffer$, c%, 1) = "*" 'Birth
    CASE "[", "A", "B"
        MID$(Buffer$, c%, 1) = "B" 'Begin 1st Periode
    CASE "(", "W", "α"
        MID$(Buffer$, c%, 1) = CHR$(224) 'Begin 2nd Periode
    CASE ")", "X", "Ω"
        MID$(Buffer$, c%, 1) = CHR$(234) 'End 2nd Periode
    CASE "]", "E"
        MID$(Buffer$, c%, 1) = "E" 'End 1st Periode
    CASE "}", "+"
        MID$(Buffer$, c%, 1) = "+" 'Death
    CASE ELSE

```

```

    MID$(Buffer$, c%, 1) = " "
END SELECT
NEXT c%

```

```

ToUser$ = Buffer$
END FUNCTION 'ToUser$ _____'

```

```

'=====
SUB WriteTo (File%, FL%, DateValue$, Target%, Kind%, Source%, Place%, Num%)
'=====
' Write a protocol of found dating relations to the corresponding result
' file.
' `FL%`: Number of data records in `File%`.
' `Place%` and `Source%` show the origin of the dating suggestion,
' `Kind%` stands for one of totally eleven different possibilities of date
' calculation:
'   `Kind%` = 0`:          Copy existing date
'   `Kind%` = 1 and 2`:    Forward relations of predecessors (relative)
'   `Kind%` = 3 and 4`:    Forward relations with absolute relation
'   `Kind%` = 5 and 6`:    Forward relations of duration
'   `Kind%` = 7 and 8`:    Backward relations of predecessors (relative)
'   `Kind%` = 9 and 10`:   Backward relations with absolute relation
' If `Place%` and `Num%` = `0`, then the average end result is described.
'
' Handling:
'   8/ 4/2001 - 1/22/2003:    Norbert Südland
' Check:
'   9/10/2001                  Norbert Südland
' Translation:
'   4/14/2008:                 Norbert Südland
'-----

```

```

DIM Array$ 'AS STRING
DIM L% 'AS INTEGER

```

```

'Hint: `Place%` and `Num%` are ordinary numbers of data records.
'       `Target%` and `Source%` own values between `1` and `6`:
'-----

```

```

IF Num% = 0 THEN
    IF Place% = 0 THEN
        IF Source% = 0 THEN
            IF Target% = 0 THEN
                Pause
            ELSE
                Pause
            END IF
        ELSE
            'Pause
        END IF
    ELSE
        Pause 'Not yet handled!
    END IF
ELSE
    IF Source% >= 1 OR Target% <= 6 THEN
        'Pause
    ELSE
        Pause 'Not yet handled!
    END IF

```

```

    IF Target% >= 1 OR Target% <= 6 THEN
        'Pause
    ELSE
        Pause      'Not yet handled!
    END IF
END IF

'-----
'An Exception is Accepted Only, if All Four Values Give `0`:
'Then No Change of Datings is Needed.
'-----

Array$ = SPACE$(ArrayLength%)
Change Array$, "array.date", DateValue$
Change Array$, "array.destination", Symbol$(Target%)
Change Array$, "array.name", STR$(Kind%)
Change Array$, "array.source", Symbol$(Source%)
Change Array$, "array.from", STR$(Place%)
Change Array$, "array.to", STR$(Num%)
Change Array$, "array.end", CHR$(186) + CHR$(13) + CHR$(10)

FL% = FL% + 1
PUT #File%, FL% * ArrayLength% + 1, Array$
END SUB 'WriteTo _____

'=====
FUNCTION YesNoQuestion$ (Question$, Message$)
'=====
'Asks the Question$ after [ ESC ] and expects "Y" for Yes or "N" for No.
'At [ ESC ] is returned "N", at [ Enter ] ist returned "Y".
'
'Handling: 3/ 4/2017 Norbert Südland, Aalen
'-----
DIM Answer$      'AS STRING

SELECT CASE INKEY$
CASE CHR$(27)
DO
    PRINT Question$; " [ Y / N ] ";
    LOCATE CSRLIN, POS(0) - 2
    DO
        Answer$ = INKEY$
    LOOP UNTIL Answer$ <> ""
    SELECT CASE LEFT$(Answer$, 1)
    CASE "Y", "y", CHR$(13)
        Answer$ = "Y"
        PRINT Answer$
        ProtocolMessage Message$, 1
    CASE "N", "n", CHR$(27)
        Answer$ = "N"
        PRINT Answer$
    CASE ELSE
        Answer$ = ""
        PRINT
    END SELECT
    LOOP UNTIL Answer$ <> ""
END SELECT
END SELECT

```

```
    YesNoQuestion$ = Answer$  
END FUNCTION 'YesNoQuestion$ _____'
```