```
                PRINT COMMENT "MEMORY LIST FOLLOWS"
                PRINT COMMENT " "
                Through MEMLIST, FOR I=1 , 1, I > 4
MEMLST          TXTPRT.(MYTRAN(I),0)
                Transfer To CHANGE
            End Conditional
            THEME=POPTOP.(INPUT)
            SUBJECT=KEY(HASH.(THEME,5))
            S=SEQRDR.(SUBJECT)
LOOK        TERM=SEQLR.(S,F)
            Whenever F > 0, Transfer To FAIL
            Whenever TOP.(TERM) = THEME, Transfer To FOUND
            Transfer To LOOK
FOUND       Transfer To DELTA(J)
DELTA(1)    TPRINT.(TERM,0)
            Transfer To CHANGE
FAIL        PRINT COMMENT "LIST NOT FOUND"
            Transfer To CHANGE
DELTA(2)    S=SEQRDR.(TERM)
            OLD=POPTOP.(INPUT)
READ(1)     OBJCT=SEQLR.(S,F)
            Whenever F > 0, Transfer To FAIL
            Whenever F <> 0, Transfer To READ(1)
            INSIDE=SEQRDR.(OBJECT)
READ(2)     IT=SEQLR.(INSIDE,F)
            Whenever F > 0, Transfer To READ(1)
            SIT=SEQRDR.(IT)
            SOLD=SEQRDR.(OLD)
ITOLD       TOLD=SEQLR.(SOLD,FOLD)
            DIT=SEQLR.(SIT,FIT)
            Whenever TOLD = DIT AND FOLD <= 0,Transfer To ITOLD
            Whenever FOLD > 0, Transfer To OK(J)
            Transfer To READ(2)
OK(2)       SUBST.(POPTOP.(INPUT),LSPNTR.(INSIDE))
            Transfer To CHANGE
OK(3)       NEWBOT.(POPTOP.(INPUT),OBJCT)
            Transfer To CHANGE
DELTA(3)    Transfer To DELTA(2)
DELTA(4)    Whenever NAMTST.(BOT.(TERM)) = 0
                BOTTOM=POPBOT.(TERM)
                NEWBOT.(POPTOP.(INPUT),TERM)
                NEWBOT.(BOTTOM,TERM)
            Otherwise
                NEWBOT.(POPTOP.(INPUT),TERM)
            End Conditional
            Transfer To CHANGE
DELTA(6)    S=SEQRDR.(TERM)
READ(6)     OBJCT=SEQLR.(S,F)
            Whenever F > 0, Transfer To FAIL
            Whenever F <> 0, Transfer To READ(6)
            OBJCT=SEQLL.(S,F)
            Whenever LNKLL.(OBJECT) = 0
                SUBST.(POPTOP.(INPUT),LSPNTR.(S))
            Otherwise
                NEWTOP.(POPTOP.(INPUT),LSPNTR.(S))
            End Conditional
            Transfer To CHANGE
            End Function

          R* * * * * * * * * * END OF MODIFICATION ROUTINE
```

```
        LPRIN
            EXTERNAL FUNCTION (LST,TAPE)
            NORMAL MODE IS INTEGER
            ENTRY TO LPRINT.
            BLANK = "      "
            EXECUTE PLACE.(TAPE,0)
            LEFTP = 606074606060K
            RIGHTP= 606034606060K
            BOTH  = 607460603460K
            EXECUTE NEWTOP.(SEQRDR.(LST),LIST.(STACK))
            S=POPTOP.(STACK)
BEGIN       EXECUTE PLACE.(LEFTP,1)
NEXT        WORD=SEQLR.(S,FLAG)
            Whenever FLAG < 0
            EXECUTE PLACE.(WORD,1)
            Whenever S > 0, PLACE.(BLANK,1)
            Transfer To NEXT
            OR Whenever FLAG > 0
            EXECUTE PLACE.(RIGHTP,1)
            Whenever LISTMT.(STACK) = 0, Transfer To DONE
            S=POPTOP.(STACK)
            Transfer To NEXT
            OTHERWISE
            Whenever LISTMT.(WORD) = 0
            EXECUTE PLACE.(BOTH,1)
            Transfer To NEXT
            OTHERWISE
            EXECUTE NEWTOP.(S,STACK)
            S=SEQRDR.(WORD)
            Transfer To BEGIN
            End Conditional
            End Conditional
DONE        EXECUTE PLACE.(0,-1)
            EXECUTE IRALST.(STACK)
            FUNCTION RETURN LST
            END OF FUNCTION

        TESTS
            EXTERNAL FUNCTION(CAND,S)
            NORMAL MODE IS INTEGER
            DIMENSION FIRST(5),SECOND(5)
            ENTRY TO TESTS.
            STORE=S
            READER=SEQRDR.(CAND)
            Through ONE, FOR I=0,1, I > 100
            FIRST(I)=SEQLR.(READER,FR)
ONE         Whenever READER > 0, Transfer To ENDONE
ENDONE      SEQLL.(S,F)
            Through TWO, FOR J=0,1, J > 100
            SECOND(J)=SEQLR.(S,F)
TWO         Whenever S > 0, Transfer To ENDTWO
ENDTWO      Whenever I <> J, Function Return 0
            Through LOOK, FOR K=0,1, K> J
LOOK        Whenever FIRST(K) <> SECOND(K), Function Return 0
            EQL=SEQLR.(READER,FR)
            Whenever EQL <> "="
            SEQLL.(READER,FR)
            Function Return READER
            Otherwise
            POINT=LNKL.(STORE)
```

```
            MTLIST.(INPUT)
            Through MLST, FOR I=1,1, I > 4
MLST        LIST.(MYTRAN(I))
            MINE=0
            LIST.(MYLIST)
            Through KEYLST, FOR I=0,1, 1 > 32
KEYLST      LIST.(KEY(I))

           R* * * * * * * * * * READ NEW SCRIPT

BEGIN       MTLIST.(INPUT)
            NODLST.(INPUT)
            LISTRD.(INPUT,SCRIPT)
            Whenever LISTMT.(INPUT) = 0
                TXTPRT.(JUNK,0)
                MTLIST.(JUNK)
                Transfer To START
            End Conditional
            Whenever TOP.(INPUT) = "NONE"
                NEWTOP.(LSSCPY.(INPUT,LIST.(9)),KEY(32))
                Transfer To BEGIN
               OR Whenever TOP.(INPUT) = "MEMORY"
                POPTOP.(INPUT)
                MEMORY=POPTOP.(INPUT)
                Through MEM, FOR I=1,1, I > 4
MEM             LSSCPY.(POPTOP.(INPUT),MYTRAN(I))
                Transfer To BEGIN
               Otherwise
                NEWBOT.(LSSCPY.(INPUT,LIST.(9)),KEY(HASH.(TOP.(INPUT),5)))
                Transfer To BEGIN
            End Conditional

           R* * * * * * * * * * BEGIN MAJOR LOOP

START       TREAD.(MTLIST.(INPUT),0)
            KEYWRD=0
            PREDNC=0
            LIMIT=LIMIT+1
            Whenever LIMIT = 5, LIMIT=1
            Whenever LISTMT.(INPUT) = 0, Transfer To ENDPLA
            IT=0
            Whenever TOP.(INPUT) = "+"
                CHANGE.(KEY,MYTRAN)
                Transfer To START
            End Conditional
            Whenever TOP.(INPUT) = "*", Transfer To NEWLST
            S=SEQRDR.(INPUT)
NOTYET      Whenever S < 0
                SEQLR.(S,F)
                Transfer To NOTYET
               Otherwise
                WORD=SEQLR.(S,F)
                Whenever WORD = "." OR WORD = "," OR WORD = "BUT"
                    Whenever IT = 0
                        NULSTL.(INPUT,LSPNTR.(S),JUNK)
                        MTLIST.(JUNK)
                        Transfer To NOTYET
                       Otherwise
                        NULSTR.(INPUT,LSPNTR.(S),JUNK)
                        MTLIST.(JUNK)
                        Transfer To ENDTXT
```

```
               R* * * * * * * * * * END OF MAJOR LOOP

ENDTXT          Whenever IT = 0
                    Whenever LIMIT = 4 AND LISTMT.(MYLIST) <> 0
                        OUT=POPTOP.(MYLIST)
                        TXTPRT.(OUT,0)
                        IRALST.(OUT)
                        Transfer To START
                       Otherwise
                        ES=BOT.(TOP.(KEY(32)))
                        Transfer To TRY
                    End Conditional
                   OR Whenever KEYWRD = MEMORY
                    I=HASH.(BOT.(INPUT),2)+1
                    NEWBOT.(REGEL.(MYTRAN(I),INPUT,LIST.(MINE)),MYLIST)
                    SEQLL.(IT,FR)
                    Transfer To MATCH
                   Otherwise
                    SEQLL.(IT,FR)

               R* * * * * * * * * * MATCHING ROUTINE

MATCH               ES=SEQLR.(IT,FR)
                    Whenever TOP.(ES) = "="
                        S=SEQRDR.(ES)
                        SEQLR.(S,F)
                        WORD=SEQLR.(S,F)
                        I=HASH.(WORD,5)
                        SCANER=SEQRDR.(KEY(I))
SCAN                    ITS=SEQLR.(SCANER,F)
                        Whenever F > 0, Transfer To NOMATCH(LIMIT)
                        Whenever WORD = TOP.(ITS)
                            S=SEQRDR.(ITS)
SCANI                       ES=SEQLR.(S,F)
                            Whenever F <> 0, Transfer To SCANI
                            IT=S
                            Transfer To TRY
                        Otherwise
                            Transfer To SCAN
                        End Conditional
                    End Conditional
                    Whenever FR > 0, Transfer To NOMATCH(LIMIT)

;;; A lot of the core work is done by the complex SLIP matching and
;;; rebuilding functions YMATCH and ASSMBL (see the latter at HIT)
;;; These are described on pages 62L28-29 of the SLIP manual:
;;;    https://drive.google.com/file/d/1XtF7EM1KhwMPKsp5t6F0gwN-8LsNDPOl

TRY                 Whenever YMATCH.(TOP.(ES),INPUT,MTLIST.(TEST)) = 0,Transfer To MA
                    ESRDR=SEQRDR.(ES)
                    SEQLR.(ESRDR,ESF)
                    POINT=SEQLR.(ESRDR,ESF)
                    POINTR=LSPNTR.(ESRDR)
                    Whenever ESF = 0
                        NEWBOT.(1,POINTR)
                        TRANS=POINT
                        Transfer To HIT
                       Otherwise
                        Through FNDHIT,FOR I=0,1, I > POINT
FNDHIT                  TRANS=SEQLR.(ESRDR,ESF)
```

```
WRITE               CONTINUE
                    LPRINT.(MTLIST.(INPUT),SCRIPT)
                    EXIT.

                   R* * * * * * * * * * SCRIPT ERROR EXIT

NOMATCH(1)          PRINT COMMENT "PLEASE CONTINUE "
                    Transfer To START
NOMATCH(2)          PRINT COMMENT "HMMM "
                    Transfer To START
NOMATCH(3)          PRINT COMMENT "GO ON , PLEASE "
                    Transfer To START
NOMATCH(4)          PRINT COMMENT "I SEE "
                    Transfer To START
                    VECTOR VALUES SNUMB= "I3 * "
                    End of Program
```