APPENDIX B: BALM LISTING

Given below is a listing of a version of the BALM system written in BALM. This is actually a minimal version of the system used to bootstrap itself, and does not include certain features described in the text. In particular, BREAKUP and CONSTRUCT are not included, the semicolon is not detected as an absolute terminator, bit-strings and reals are not supported, operators for manipulating strings are rather primitive, and the compiler is badly deficient in error detection. Also, arguments and local variables are usually compiled as stack locations, and so are not accessible inside another procedure or block respectively. If preceded in the declaration by a $, the compiler compiles references to the symbol table, thus permitting access to the value from other blocks or procedures. Thus in general any argument or variable whose scope is not strictly local should be preceded by $.

```
COMMENT
(*****************************************************************
                                MAIN PROGRAM
*****************************************************************)

   BALM=PROC(), BEGIN( ),
        INITIATE( ),
        EXECUTE(INPUT,OUTPUT),
        STOP()
        END END;
EXECUTE=PROC($INPUT,$OUTPUT),
        BEGIN(ST,LS,$CURCOM,$PREVCOMMAND),
   MOR, PRINT(),
        ST=RDSEMI(INPUT),
        LS=TRANSLAT(ST),
        IF TALKATIVE EQ 2 THEN DO
           PRINT(=SYNTAX,=TREE), PRINT(LS) END,
        IF ¬ZR ERCOUNT THEN GOTO ERR,
        IF LS EQ =RESUME THEN RETURN NIL,
        IF LS EQ =STOP THEN RETURN NIL,
        LS = =LAMBDA:NIL:LS:NIL,
        CODEGEN(=CURCOM,LS),
        CURCOM(),
        GOTO MOR,
   ERR, PRINT(LIST(ERCOUNT,=ERRORS)),
        PREVCOMMAND=ST, GO MOR
        END END;

 RDSEMI=PROC(I), BEGIN(B,E,TOK), B=E=NIL:NIL,
   MOR, TOK=READ(I), IF TOK EQ IEOS THEN RETURN TL B,
        E=ADDON(E,TOK), GOTO MOR  END END;
COMMENT
(**************************************************************
                        UTILITY ROUTINES
**************************************************************)
 VFROML=PROC(L),BEGIN(N,V,I),
        N=0, V=L,
        WHILE V REPEAT DO N=N+1, V=TL V END, V=MAKVECTOR(N),
        FOR I=(1,N) REPEAT DO V[I]=HD L, L=TL L END,
        RETURN V  END END;
  ADDON=PROC(X,Y), TL RPLACD(X,Y:NIL) END;
 LOOKUP=PROC(X,L), BEGIN(P),
   MOR, IF IDQ(L) AND IDQ(X) THEN DO P=L, L=PROPL(X), X=P END,
        IF ¬PAIRQ(L) THEN RETURN NIL,
        IF X EQ HD HD L THEN RETURN HD TL HD L,
        L=TL L, GOTO MOR  END END;
ORDINAL=PROC(A,L),ORD1(A,L,1) END;
   ORD1=PROC(A,L,I),
        IF NULL(L) THEN NIL ELSEIF A EQ HD L THEN I
           ELSE ORD1(A,TL L,I+1)  END;
 LENGTH=PROC(X),IF PAIRQ(X) THEN 1+LENGTH(TL X)
        ELSEIF VECTQ(X) OR STRQ(X) OR CODEQ(X) THEN OLENGTH(X)
        ELSE 0  END;
IFROMID=PROC(X),X+0 END;
 GENSYM=PROC(), BEGIN(I),
        I=5, GENSYMB[I]=GENSYMB[I]+1,
        WHILE GENSYMB[I] GT NINE REPEAT DC
           GENSYMB[I]=ZERO,I=I-1,GENSYMB[I]=GENSYMB[I]+1 END,
        RETURN IDFROMS(SFROMV(GENSYMB))  END END;
   MAPX=PROC(X,P), IF PAIRQ(X) THEN MAPL(X,P)
        ELSEIF VECTQ(X) THEN MAPV(X,P)
        ELSE P(X)  END;
   MAPL=PROC(L,P), BEGIN(B,E), B=E=NIL:NIL,
   MOR, IF ¬PAIRQ(L) THEN RETURN TL B,
        E=ADDON(E,P(HD L)), L=TL L, GOTO MOR  END END;
   MAPV=PROC(V,P), BEGIN(N,VV,I), N=OLENGTH(V),
           VV=MAKVECTOR(N),
        FOR I=(1,N) REPEAT VV[I]=P(V[I]), RETURN VV  END END;
    SET=PROC(ID,X), IF IDQ(ID) THEN SETVALUE(ID,X)
        ELSE PRINT(ID:=IS:=NOT:=AN:=ID:NIL)  END;
 MEMBER=PROC(X,L), BEGIN(),
   MOR, IF PAIRQ(L) THEN
           (IF X EQ HD L THEN RETURN TRUE
            ELSE DO L=TL L, GOTO MOR END)
        ELSE RETURN NIL  END END;
 LFROMV=PROC(V), BEGIN(L,N,I),
        N=OLENGTH(V), L=NIL,
        FOR I=(N,1,-1) REPEAT L=V[I]:L,
        RETURN L  END END;
COMMENT

(*************************************************************
                     INITIATION ROUTINES
*************************************************************)

 INITIATE=PROC(), BEGIN(),
          INITIO(),
          INITUNARY(),
          INITINFIX(),
          INITEXP(),
          INITCODG(),
          INITOPL(),
          INITMISC(),
          RETURN NIL  END END;
   INITIO=PROC(), BEGIN(LIN),
      LIN=RDLINE(1),
      LIN=VFROMS(LIN),
    BLANK=LIN[1], PERIOD=LIN[2],
     ZERO=LIN[3],  STAR=LIN[4],
    MSIGN=LIN[5],  STRQU=LIN[6],
     LPAR=LIN[7],  RPAR=LIN[8],
      LBR=LIN[9], RBR=LIN[10],
     NINE=LIN[11],  LETTB=LIN[12],
    SEMIC=LIN[13],  IEOS=IDFROMS(SFROMV(VECTOR(SEMIC))),
   IDTRUE=IDFROMS(SFROMV(VECTOR(LIN[14],LIN[15],LIN[16],
             LIN[17]))),
    IDNIL=IDFROMS(SFROMV(VECTOR(LIN[18],LIN[19],LIN[20]))),
    LPVAR=IDFROMS(SFROMV(VECTOR(LPAR))),
    RPVAR=IDFROMS(SFROMV(VECTOR(RPAR))),
    LBVAR=IDFROMS(SFROMV(VECTOR(LBR))),
    RBVAR=IDFROMS(SFROMV(VECTOR(RBR))),
   PERVAR=IDFROMS(SFROMV(VECTOR(PERICD))),
    INPUT=MAKFILE(1,72),
   OUTPUT=MAKFILE(2,72),
BLANKLINE=MAKVECTOR(72),
          FOR I=(1,72) REPEAT BLANKLINE[I]=BLANK,
          RETURN NIL  END END;
INITUNARY=PROC(), BEGIN(),
UNARYLIST=LIST(
          LIST(≥DO,LIST(≥BRCKT,≥PROGN,100,100)),
          LIST(≥BEGIN,LIST(≥BRCKT,≥PROG,100,100)),
          LIST(≥PROC,LIST(≥BRCKT,≥LAMBDA,100,100)),
          LIST(≥IF,LIST(≥UNARY,≥IF,200,200)),
          LIST(≥RETURN,LIST(≥UNARY,≥RETURN,500,500)),
          LIST(≥WHILE,LIST(≥UNARY,≥WHILE,500,500)),
          LIST(=FOR, LIST(=UNARY,=FOR,500,500)),
          LIST(≥GOTO,LIST(≥UNARY,≥GO,500,500)),
          LIST(≥GO,LIST(≥UNARY,≥GO,500,500)),
          LIST(≥-,LIST(≥UNARY,≥NILQ,1200,1200)),
          LIST(=NOT,LIST(=UNARY,=NILQ,1200,1200)),
          LIST(≥NULL,LIST(≥UNARY,≥NILQ,1200,1200)),
          LIST(=PL,LIST(=UNARY,=IPOSQ,1200,1200)),
          LIST(=ZR,LIST(=UNARY,=IZEROQ,1200,1200)),
          LIST(≥-,LIST(≥UNARY,≥INEG, 1700,1700)),
          LIST(=SIZE,LIST(=UNARY,=OLENGTH,1900,1900)),
          LIST(≥$,LIST(≥UNARY,≥EVAL,1900,1900)),
          LIST(≥TL,LIST(≥UNARY,≥TL ,2000,2000)),
          LIST(≥HD,LIST(≥UNARY,≥HD ,2000,2000))
          ),
          RETURN NIL  END END;
INITINFIX=PROC(),BEGIN(),
INFIXLIST=LIST(
          LIST(≥TERMINATOR,LIST(≥INFIX,≥TERMINATOR,0,-1)),
          LIST(≥END,LIST(≥INFIX,≥END,0,0)),
          LIST(≥,,LIST(≥INFIX,≥COMMA,100,100)),
          LIST(≥ELSEIF,LIST(≥INFIX,≥ELSEIF,300,300)),
          LIST(≥ELSE,LIST(≥INFIX,≥ELSE,300,300)),
          LIST(≥THEN,LIST(≥INFIX,≥THEN,400,400)),
          LIST(≥REPEAT,LIST(≥INFIX,≥REPEAT,600,600)),
          LIST(≥=,LIST(≥INFIX,≥SETQ,700,701)),
          LIST(≥:,LIST(≥INFIX,≥PAIR,800,801)),
          LIST(≥OR,LIST(≥INFIX,≥OR,1000,1001)),
          LIST(≥AND,LIST(≥INFIX,≥AND,1100,1101)),
          LIST(=NE,LIST(=INFIX,=NE,1200,1200)),
          LIST(=LT,LIST(=INFIX,=LT,1200,1200)),
          LIST(=GE,LIST(=INFIX,=GE,1200,1200)),
          LIST(=GT,LIST(=INFIX,=GT,1200,1200)),
          LIST(=LE,LIST(=INFIX,=LE,1200,1200)),
          LIST(=SIM,LIST(=INFIX,=SIMQ,1200,1200)),
          LIST(≥-,LIST(≥INFIX,≥ISUB   ,1501,1500)),
          LIST(≥+,LIST(≥INFIX,≥IADD,1501,1500)),
          LIST(≥*,LIST(≥INFIX,≥IMPY ,1601,1600)),
          LIST(=≡,LIST(=INFIX,=IDENTQ,1400,1400)),
          LIST(≥/,LIST(≥INFIX,≥IDIV   ,1601,1600)).
          LIST(≥↑,LIST(≥INFIX,≥IEXP,1800,1800)),
          LIST(=EQ,LIST(=INFIX,=IDENTQ,1400,1400))
          ),
          RETURN NIL  END END;
  INITEXP=PROC(), BEGIN(),
MACROLIST=LIST(
          LIST(=IF,MIF),
          LIST(=THEN,EXPERR),
          LIST(=ELSE,EXPERR),
          LIST(=ELSEIF,EXPERR),
          LIST(=LAMBDA,MXLMBDA),
          LIST(=PROG,MPROG),
          LIST(=PROGN,MPROGN),
          LIST(=QUOTE,DUMMY),
          LIST(=SETQ,MSETQ),
          LIST(=REPEAT,EREMSPS),
          LIST(=COMMA,EREMSPS)    ),
LMACROLIST=LIST(
          LIST(=HD,LCAR),
          LIST(=TL,LCDR),
          LIST(=EVAL,LEVAL),
          LIST(=INDEX,LINDEX),
          LIST(=SUBV,LSUBV),
          LIST(=QUOTE,LQUOTE)    ),
          RETURN NIL  END END;
 INITCODG=PROC(),BEGIN(),
CODGENLIST=LIST(
          LIST(=LAMBDA,GLAMBDA),LIST(=PROG,GPROG),
          LIST(=RETURN,GRETURN), LIST(=PROGN,GPROGN),
          LIST(=GO,GGO), LIST(=COND,GCOND),
          LIST(=AND,GAND), LIST(=OR,GOR),
          LIST(=QUOTE,GQUOTE), LIST(=SETG,GSETG),
          LIST(=WHILE,GWHILE), LIST(=FOR,GFOR),
          LIST(=LIST,GLIST), LIST(=VECTOR,GVECTOR)
```

```
   KCALL=27B, KRETPROC=115B,
  KNVARS=36B, KRETPROG=131B, KSETSTK=137B, KPOP=35B,
    KARG=33B, KVAR=31B, KGLOB=5B,
 KASTORE=34B, KVSTORE=32B, KGSTORE=6B,
        KNUM1=26B, KNUM2=4B, KNUM3=37B,
        KNIL=136B, KTRUE=135B,
        KJMP=3B, KJMPT=1B, KJMPF=2B,
        KLBL=11B, KJMPI=52B,
        KLIST=44B, KVECTOR=56B,
        KTLOOP=14B, KSTEPLOOP=15B,
        KINEG=76B, KMAKVAR=41B,
        RETURN NIL   END END;
 INITOPL=PROC(),BEGIN(),
  OPLIST=LIST(
        LIST(=PAIR,61B), LIST(=HD,123B), LIST(=TL,124B),
        LIST(=RPLACA, 121B), LIST(=RPLACD,122B),
        LIST(=OLIST,44B),
        LIST(=MAKVECTOR,140B), LIST(=OVECTOR,56B),
        LIST(=INDEX,117B), LIST(=SETINDEX,120B),
        LIST(=SUBV,163B), LIST(=SETSUBV,164B),
        LIST(=CONCATV,165B),
        LIST(=OLENGTH,114B), LIST(=OSTRING,55B),
        LIST(=INTQ,77B), LIST(=STRQ,101B),
        LIST(=CODEQ,104B),
        LIST(=IDQ,105B), LIST(=LBLQ,107B),LIST(=VECTQ,102B),
        LIST(=PAIRQ,103B), LIST(=LOGQ,161B),
        LIST(=IDENTQ,113B), LIST(=SIMQ,162B),
        LIST(=IPOSQ,111B), LIST(=IZEROG,112B),
        LIST(=NILQ,132B), LIST(=OAND,134B), LIST(=OOR,133B),
        LIST(=VFROMS,46B), LIST(=SFROMV,45B),
        LIST(=IDFROMS,60B), LIST(=SFROMID,130B),
        LIST(=CFROMV,150B),
        LIST(=IADD,71B), LIST(=ISUB,72E), LIST(=INEG,76B),
        LIST(=IMPY,73B), LIST(=IDIV,74E), LIST(=IEXP,75B),
        LIST(=PROPL,160B), LIST(=SETPROPL,110B),
        LIST(=VALUE,50B), LIST(=SETVALUE,51B),
        LIST(=MODE,152B), LIST(=SETMODE,151B),
        LIST(=RDLINE,141B), LIST(=WRLINE,142B),
        LIST(=REWIND,143B), LIST(=BACKSPACE,144B),
        LIST(=GARBCOLL,153B),
        LIST(=SAVEALL,145B), LIST(=RESLMEALL,146B),
        LIST(=ENDFILE,147B),LIST(=PROTECT,155B),
        LIST(=TIME,154B),
        LIST(=NUMARGS,47B), LIST(=ARGUMENT,42B),
        LIST(=NE,LIST(113B,132B)),
        LIST(=LT,LIST(72B,111B,132B)),
            LIST(=GE,LIST(72B,111B)),
            LIST(=GT,LIST(72B,76B,111B,132B)),
            LIST(=LE,LIST(72B,76B,111B)),
            LIST(=STOP,116B)     ),
            RETURN NIL  END END;
 INITMISC=PROC(),BEGIN(),
            FALSE=NIL,
            SYSLIST=NIL,
            TTYFLAG=NIL,
            TALKATIVE=0,
            GENSYMB=VECTOR(STAR,ZERO,ZERO,ZERO,ZERO),
            RETURN NIL
            END END;
COMMENT
(*************************************************************
                         I/O ROUTINES
*************************************************************)

 MAKFILE=PROC(FN,LLEN),
           BEGIN(LIN,I),
           LIN=MAKVECTOR(LLEN), FOR I=(1,LLEN) REPEAT
              LIN[I]=BLANK,
           RETURN VECTOR(FN,LIN,LLEN,2)
           END END;
    READ=PROC(FIL),
              BEGIN(ITM,$LIN,$LLEN,$NEXT,$TERMLINE),
           FN=FIL[1], LIN=FIL[2], LLEN=FIL[3], NEXT=FIL[4].
              TERMLINE=READIN,
           ITM=RDITEM(),
           FIL[2]=LIN, FIL[4]=NEXT, FIL[3]=LLEN, RETURN ITM
           END END;
 RDTOKEN=PROC(FIL), BEGIN(ITM,$LIN,$LLEN,$NEXT,$TERMLINE),
           FN=FIL[1], LIN=FIL[2], LLEN=FIL[3], NEXT=FIL[4],
              TERMLINE=READIN,
           ITM=LXSCAN(), FIL[2]=LIN, FIL[4]=NEXT, FIL[3]=LLEN,
           RETURN ITM  END END;
  RDITEM=PROC(),
           BEGIN(ITM),
           ITM=LXSCAN(),
           IF ITM EQ LPVAR THEN ITM=GETLIST()
           ELSEIF ITM EQ LBVAR THEN ITM=GETVECT(),
              IF ITM EQ IDTRUE THEN RETURN TRUE,
              IF ITM EQ IDNIL THEN RETURN NIL,
           RETURN ITM
           END END;
```

```
GETLIST=PROC(),
        BEGIN(ITM),
        ITM=RDITEM(),
        IF ITM EQ RPVAR THEN RETURN NIL
        ELSEIF ITM EQ PERVAR THEN RETURN HD GETLIST()
        ELSE RETURN ITM:GETLIST()
        END END;
GETVECT=PROC(), VFROML(GETV()) END;
   GETV=PROC(),
        BEGIN(ITM),
        ITM=RDITEM(),
        IF ITM EQ RBVAR THEN RETURN NIL
        ELSE RETURN ITM:GETV()
        END END;
 LXSCAN=PROC(),
        BEGIN(C,J,E),
   NXT, IF NEXT GT LLEN THEN TERMLINE(),
        C=LIN[NEXT], NEXT=NEXT+1,
        IF C EQ BLANK THEN GOTO NXT,
        J=NEXT- 1,
        IF C LT ZERO THEN GO SYMB
        ELSEIF C LE NINE THEN GO NUMB
        ELSEIF C EQ STRQU THEN GO STR,
        RETURN IDFROMS(SFROMV(VECTOR(C))),
  SYMB, WHILE NEXT LE LLEN AND LIN[NEXT] LE NINE REPEAT
           NEXT = NEXT+1,
        RETURN IDFROMS(SFROMV(SUBV(LIN,J,NEXT-J))).
  NUMB, E=C-ZERO,
        WHILE NEXT LE LLEN AND (C=LIN[NEXT]) GE ZERO AND C LE
           NINE REPEAT
           DO E=E*10+C-ZERO, NEXT=NEXT+1 END,
        IF C EQ LETTB THEN DO NEXT=NEXT+1, RETURN  MAKOCT(E)
           END,
        RETURN E,
   STR, E=MAKVECTOR(0),
  MSTR, IF NEXT GT LLEN THEN DO
           E=CONCATV(E,SUBV(LIN,J+1,LLEN-1)), J=0,
           TERMLINE()  END,
        IF ¬IDENTQ(LIN[NEXT],STRQU) THEN DO NEXT=NEXT+1.
           GOTO MSTR END.
        E=CONCATV(E,SUBV(LIN,J+1,NEXT-J-1)),
        NEXT=NEXT+1, RETURN SFROMV(E)
        END END;
 MAKOCT=PROC(I), BEGIN(B,M,J), M=1, B=0,
   MOR, IF I EQ 0 THEN RETURN B,
          J=I/10, I=I-J*10, B=B+M*I, I=J, M=M*B, GOTO MOR
             END END;
   READIN=PROC(), DO LIN=RDLINE(FN), LIN=VFROMS(LIN),
          IF FN EQ 1 AND ¬TTYFLAG THEN
             WRLINE(SFROMV(CONCATV(VECTOR(BLANK),LIN)),
             OUTPUT[1])
          ELSE NIL,
          LLEN=OLENGTH(LIN), NEXT=1,  END  END;
    WRITE=PROC(L,FIL),
          BEGIN($FN,$LIN,$LLEN,$NEXT,$BPCNT,$TERMLINE),
          FN=FIL[1], LIN=FIL[2], LLEN=FIL[3], NEXT=FIL[4].
             TERMLINE=WRITOUT,
          BPCNT=0, PUTITEM(L), TERMLINE(),
          FIL[2]=LIN, FIL[4]=NEXT, RETURN L
          END END;
    PRINT=PROC(),
          BEGIN($FN,$LIN,$LLEN,$NEXT,$BPCNT,$TERMLINE,I.N,
            FIL,TR),
          TR=TRACE, TRACE=0,
          N=NUMARGS(), FIL=OUTPUT, TERMLINE=WRITOUT,
          FN=FIL[1], LIN=FIL[2], LLEN=FIL[3], NEXT=FIL[4],
          BPCNT=0, FOR I=(1,N) REPEAT PUTITEM(ARGUMENT(1)),
             TERMLINE().
          FIL[2]=LIN, FIL[4]=NEXT, TRACE=TR,
          RETURN ARGUMENT(N)  END END;
  WRITOUT=PROC(), BEGIN(I),
          WRLINE(SFROMV(LIN),FN), NEXT=BPCNT+2,
             SETSUBV(LIN,1,LLEN,BLANKLINE)
          END END;
 PUTBLANK=PROC(), IF NEXT GT LLEN THEN TERMLINE() ELSE
             PUTCH(BLANK) END;
  PUTITEM=PROC(L),
          IF VECTQ(L) THEN PUTVECT(L)
          ELSEIF PAIRQ(L) THEN DO
             IF NEXT GT LLEN-10 THEN TERMLINE() ELSE NIL,
                BPCNT=BPCNT+1,
               PUTCH(LPAR), PUTLIST(L)  END
          ELSEIF STRQ(L) THEN PUTSTR(L)
          ELSEIF IDQ(L) THEN PUTCHV(VFROMS(SFROMID(L)))
          ELSEIF INTQ(L) THEN PUTINT(L)
             ELSEIF IDENTQ(L,TRUE) THEN PUTCHV(VFROMS(SFROMID
                (IDTRUE)))
             ELSEIF IDENTQ(L,NIL) THEN PUTCHV(VFROMS(SFROMID
                (IDNIL)))
          ELSE PUTCHV(VECTOR(STAR,STAR,STAR))
          END;
 PUTVECT=PROC(L), BEGIN(N,I),
            IF NEXT GT LLEN-10 THEN TERMLINE(), PUTCH(LBR),
         N=OLENGTH(L), BPCNT=BPCNT+1,
         FOR I=(1,N) REPEAT PUTITEM(L[I]),
         NEXT=NEXT-1, PUTCH(RBR), BPCNT=BPCNT-1,
            PUTBLANK()
         END END;
 PUTLIST=PROC(L),
         IF NULL L THEN DO NEXT=NEXT-1, PUTCH(RPAR),
            BPCNT=BPCNT-1,
               PUTBLANK()  END
         ELSEIF PAIRQ(L) THEN DO PUTITEM(HD L), PUTLIST(TL L)
            END
         ELSE DO PUTCHK(PERIOD), PUTCHK(BLANK), PUTITEM(I),
            NEXT=NEXT-1, PUTCH(RPAR), BPCNT=BPCNT-1,
               PUTCHK(BLANK) END
         END;
  PUTSTR=PROC(S), BEGIN(N,I),
         S=VFROMS(S), N=OLENGTH(S),
         PUTCHK(STRQU), FOR I=(1,N) REPEAT PUTCHK(S[I]),
         PUTCHK(STROU), PUTCHK(BLANK)
         END END;
  PUTCHV=PROC(S), BEGIN(N,I),
         N=OLENGTH(S), IF N GE LLEN-NEXT THEN TERMLINE().
            SETSUBV(LIN,NEXT,N,S), NEXT=NEXT+N, PUTBLANK()
         END END;
   PUTCH=PROC(C), DO LIN[NEXT]=C, NEXT=NEXT+1 END END;
  PUTCHK=PROC(C), DO IF NEXT GT LLEN THEN TERMLINE() ELSE NIL,
         LIN[NEXT]=C, NEXT=NEXT+1  END END;
  PUTINT=PROC(N), BEGIN(S,NN,Q), S=NIL,
         IF NEXT GT LLEN-10 THEN TERMLINE(),
         IF PL N THEN NN=N ELSE DO PUTCH(MSIGN), NN=-N END,
    MOR, Q=NN/10, S=(NN-Q*10+ZERO):S, NN=Q,
         IF ¬ZR NN THEN GOTO MOR,
         WHILE S REPEAT DO PUTCH(HD S), S=TL S END,
            PUTCH(BLANK)
         END END;
COMMENT

(************************************************************
                     TRANSLATOR ROUTINES
************************************************************)

TRANSLAT=PROC(B1),BEGIN(X),
         ERCOUNT=0,
         X=FNOTN(B1),
         IF ZR ERCOUNT THEN RETURN EXPAND(X),
              RETURN(X)
              END END;
   MACDEF=PROC(B1,B2),
              MACROLIST=LIST(B1,B2):MACROLIST END;
   LMACRO=PROC(B1,B2),
              LMACROLIST=LIST(B1,B2):LMACROLIST END;
    INFIX=PROC(B1,B2,B3,B4),
              INFIXLIST=LIST(B1,LIST(≥INFIX,B4,B2,B3)):INFIXLIST
                 END;
    UNARY=PROC(B1,B2,B3),
              UNARYLIST=LIST(B1,LIST(≥UNARY,B3,B2,B2)):UNARYLIST
                 END;
  BRACKET=PROC(B1,B2,B3),
              UNARYLIST=LIST(B1,LIST(≥BRCKT,B3,B2,B2)):UNARYLIST
                 END;
COMMENT

(************************************************************
                        PARSER ROUTINES
************************************************************)

    FNOTN=PROC(B1),
             BEGIN(P,I1,I2,U,P1,$LST,UL,INFL,TERM),
             IF ¬PAIRQ(B1) THEN RETURN(B1),
             LST=B1, TERM=≥TERMINATOR,
             UL=UNARYLIST, INFL=INFIXLIST,
             P=NIL,
      DOF,   I1=GETOKEN(),
             IF PAIRQ(I1) THEN DO I1=FNOTN(I1), GOTO TEST END,
             IF I1 EQ =COMMENT THEN DO GETOKEN(), GOTO DOF END,
             IF I1 EQ =NOOP THEN DO I1=GETOKEN(), GOTO NOTU END,
             IF I1 EQ == OR I1 EQ =≥ THEN
                DO I1 = =QUOTE:GETOKEN():NIL, GOTO NOTU END,
             IF U =LOOKUP(I1,UL) THEN DO P=U:P, GOTO DOF END,
    NOTU,    I2=GETOKEN(),
             IF VECTQ(I2) THEN
                DO I1=LIST(≥INDEX,I1,FNOTN(LFRCMV(I2))),GOTO NOTU
                END,
             IF PAIRQ(I2) THEN DO
                I2=REMCOM(FNOTN(I2)),
                IF NULL(I2) THEN I2=I2:NIL ELSE NIL,
                I1=I1:I2, GOTO NOTU END,
             IF NULL(12) THEN DO I1=I1:I2, GOTO NOTU END,
             GOTO TEST1,
    TEST,    I2=GETOKEN(),
 TEST1, IF PAIRQ(I1) THEN NIL ELSEIF LOOKUP(I1, INFL) THEN
           PRINT(=WARNING:I1:=IS:=AN:=INFIX:=OPERATOR:
              NIL),
        U=LOOKUP(I2,INFL),
        IF NULL(U) THEN DO OPERROR(I2), GOTO TEST END.
 TEST2, IF NULL P AND I2 EQ TERM THEN RETURN I1,
        IF NULL(P) THEN GOTO PSH,
        IF HD TL TL HD P GT HD TL TL TL U THEN GOTO PIL,
   PSH, P=U:(I1:P), GOTO DOF,
   PLL, IF HD HD P EQ =UNARY THEN GOTO UNRY,
        IF HD HD P EQ =BRCKT THEN GOTO BRKT,
        I1=LIST(HD TL HD P,HD TL P,I1),
        P=TL TL P, GOTO TEST2,
  BRKT, I1=LIST(HD TL HD P,I1), P=TL P, GOTO NOTU,
  UNRY, I1=LIST(HD TL HD P,I1), P=TL P, GOTO TEST2
        END END;
GETOKEN=PROC(), BEGIN(TOK),
        IF ¬PAIRQ(LST) THEN RETURN =TERMINATOR,
        TOK = HD LST, LST = TL LST,
        RETURN TOK  END END;
 REMCOM=PROC(B1),BEGIN(), IF NULL(B1) THEN RETURN(NIL),
        RETURN(REMSEP(B1,≥COMMA)) END END;
 REMSEP=PROC(B1,B2),BEGIN(),
        IF ¬PAIRQ(B1) THEN RETURN B1:NIL,
        IF HD B1 EQ B2 THEN RETURN
           HD TL B1:REMSEP(HD TL TL B1,B2),
        RETURN B1:NIL  END END;
OPERROR=PROC(B1),BEGIN(),
        ERCOUNT=ERCOUNT+1,
        PRINT(B1:LIST(≥IS,≥NOT,≥AN,≥OPERATOR)),
        RETURN(NIL) END END;
COMMENT

(************************************************************
                    SYNTAX TREE PROCESSORS
************************************************************)

  EXPAND=PROC(B1), BEGIN(OP,M),
        IF ¬PAIRQ(B1) THEN RETURN(B1),
        IF PAIRQ(OP=HD B1) THEN GOTO NOTM,
        M=LOOKUP(OP,MACROLIST),
        IF NULL(M) THEN GOTO NOTM,
        RETURN(M(B1)),
   NOTM, RETURN(EXLIS(B1)) END END;
   EXLIS=PROC(B1),BEGIN(),
        IF NULL(B1) THEN RETURN(NIL),
        RETURN(EXPAND(HD B1):EXLIS (TL B1)) END END;
  EXPERR=PROC(B1),BEGIN(), PRINT(B1),
        PRINT(LIST( ≥SYNTAX,≥ERROR,≥IN,≥ABOVE,≥-,≥PASS,≥TWO
           )),
        ERCOUNT=ERCOUNT+1,RETURN(NIL) END END;
 EREMSPS=PROC(B1),EREMSP(B1,HD B1) END;
  EREMSP=PROC(B1,B2),BEGIN(),
        IF ¬PAIRQ(B1) THEN RETURN LIST(B1),
        IF HD B1 EQ B2 THEN RETURN
           EXPAND(HD TL B1):EREMSP (HD TL TL, B1,B2),
        RETURN(EXPAND(B1):NIL) END END;
    LCAR=PROC(B1),  LIST(≥RPLACA,
        EXPAND(HD TL HD TL B1),EXPAND(HD TL TL B1)  )  END;
    LCDR=PROC(B1),   LIST(≥RPLACD),
        EXPAND(HD TL HD TL B1),EXPAND(HD TL TL B1))  END;
   LEVAL=PROC(B1),   LIST(≥SET,
        EXPAND(HD TL HD TL B1),EXPAND(HD TL TL B1) ) END;
  LINDEX=PROC(B1),BEGIN(F), F=TL HD TL B1,
        RETURN(LIST(=SETINDEX,EXPAND(HD F),
           EXPAND(HD TL F), EXPAND(HD TL TL B1) )) END END;
   LSUBV=PROC(X), BEGIN(L,R), L=TL HD TL X, R=HD TL TL X,
        RETURN EXLIS(=SETSUBV:HD L:HD TL L:HD TL TL L:R:NIL)
        END END;
MXLMBDA=PROC(L), BEGIN(P), P = TL HD TL L,
        RETURN LIST(=QUOTE,LIST(HD L,REMCOM(HD P),EXPAND(HD
           TL P)))
        END END;
   DUMMY=PROC(X),X END;
MELSEIF=PROC(B1,B2,B3),BEGIN(P),
        IF B1 EQ=THEN THEN RETURN(
           LIST(LIST(EXPAND(B2),EXPAND(B3)),LIST(TRUE,NIL))),
        IF ¬ HD B2 EQ =THEN THEN GOTO IFERR,
        P=LIST(EXPAND(HD TL B2),EXPAND(HD TL TL B2)).
        IF B1 EQ =ELSE THEN RETURN(
           LIST(P,(LIST(TRUE,EXPAND(B3))))),
        IF ¬PAIRQ(B3) THEN GOTO IFERR,
        IF B1 EQ =ELSEIF THEN RETURN(
        P:MELSEIF (HD B3,HD TL B3,HD TL TL B3)),
  IFERR, RETURN(EXPERR(LIST(B1,B2,B3))) END END;
   MPROG=PROC(B1),BEGIN(P,Q),
        Q=HD (P=EXPAND(HD TL B1)),
        IF Q AND ¬PAIRQ(Q) THEN RPLACA(P,LIST(Q)),
        RETURN((HD B1):P) END END;
  MPROGN=PROC(B1),
           (HD B1):EREMSP(HD TL B1,≥COMMA) END;
```

```
     MIF=PROC(B1)BEGIN(X),X=HD TL B1,
         IF ¬PAIRQ(X) THEN RETURN(EXPERR(E1)),
         RETURN(≥COND:MELSEIF(HD X,HD TL X,HD TL TL X)) END
            END;
   MSETQ=PROC(B1),BEGIN(F1,F2,U),
         F1=HD TL B1, F2=HD TL TL B1,
         IF ¬PAIRQ(F1) THEN GOTO NOTLM,
         U=LOOKUP(HD F1,LMACROLIST),
         IF NULL(U) THEN GOTO NOTLM,
         RETURN(U(B1)),
  NOTLM, RETURN(LIST(HD B1,EXPAND(F1),EXPAND(F2))) END END;
COMMENT

(************************************************************
                     MAIN CODE GENERATOR
************************************************************)

 CODEGEN=PROC(NAM,X),
         BEGIN($ERRCNT,$LIST2,$END2,LIST3,
            $GLOBL,$LBLVALS,$NBYTE,$LBLNO,
            I,CODEV),
         ERRCNT=0, GLOBL=NIL, NBYTE=3, LBLNO=1, LBLVALS=NIL,
         LIST2=END2=(0:NIL), END2=ADDON(END2,GREFS(NAM)),
         COMP(X),
         IF TALKATIVE GE 1 THEN PRINT(=GLOBAL,=VARS,GLOBL),
         IF TALKATIVE EQ 2 THEN DO
            PRINT(=BINARY,=CODE), PRINT(LIST2) END,
         IF TALKATIVE EQ 2 THEN PRINT(=LABEL,=LIST,LBLVALS),
         LIST3=LIST2,  WHILE LIST2 REPEAT DO
            HD LIST2=SUBLBLS(HD LIST2), LIST2=TL LIST2 END,
         IF ZR ERRCNT THEN DO
            NBYTE=NBYTE-1, CODEV=MAKVECTOR(NBYTE),
            FOR I=(1,NBYTE) REPEAT DO
               CODEV[I]=IFROMID(HD LIST3), LIST3=TL LIST3 END,
            FOR I=(NBYTE,2,-1) REPEAT
               IF CODEV[I] GT 177B THEN DO
                  CODEV[I-1]=CODEV[I]/200B,
                  CODEV[I]=CODEV[I]-200B*CODEV[I-1]  END,
               ELSE NIL,
            $NAM=CFROMV(CODEV), RETURN NAM  END,
         PRINT(ERRCNT:COMP:ERRORS:NIL),
         RETURN NAM
         END END;
 SUBLBLS=PROC(X), IF ¬PAIRQ(X) THEN X ELSE
         BEGIN(Y),
         Y=LOOKUP(HD X,LBLVALS),
          IF Y THEN RETURN Y,
          ERRCNT=ERRCNT+1,
          PRINT(X:=IS:=UNDEF:=LABEL:NIL),
          RETURN NIL
          END END;
     COMP=PROC(X), BEGIN(FN,ARGL,GENR),
          IF IDQ(X) THEN RETURN GVAR(X),
          IF ¬PAIRQ(X) THEN RETURN GCON(X),
          FN=HD X, ARGL=TL X,
          IF IDQ(FN) THEN DO
             GENR=LOOKUP(FN,CODGENLIST),
             IF GENR THEN RETURN GENR(X) ELSE NIL
             END,
          RETURN CALLS(X)
          END END;
     GVAR=PROC(ATM),
          BEGIN(Y),
          IF Y=ORDINAL(ATM,ARGS) THEN ASS(KARG,Y)
          ELSEIF Y=ORDINAL(ATM,VARS) THEN ASS(KVAR,Y)
          ELSEIF MEMBER(ATM,LBLIST) THEN ASS(KLBL,0,LIST(ATM))
          ELSE ASS(KGLOB,0,GREFS(ATM)),
          RETURN NIL
          END END;
     GCON=PROC(X), BEGIN(N),
          IF NULL(X) THEN ASS(KNIL)
          ELSEIF X EQ TRUE THEN ASS(KTRUE)
          ELSEIF INTQ(X) THEN
             (IF X LT 0 THEN DO GCON(-X), ASS(KINEG) END
              ELSEIF X LE 177B THEN ASS(KNUM1,X)
              ELSEIF X LE 37777B THEN ASS(KNUM2,0,X)
              ELSE ASS(KNUM3,0,0,X)   )
          ELSEIF IDQ(X) THEN DO
             ASS(KNUM2,0,GREFS(X)), ASS(KMAKVAR) END
          ELSEIF HD X EQ =LAMBDA THEN DO
             N=GENSYM(), CODEGEN(N,X), ASS(KGLOB,0,GREFS(N))
                END
          ELSE DO N=GENSYM(), SN=X, ASS(KGLCB,0,N) END
          END END;
    CALLS=PROC(X), BEGIN(ARG,FN,ARGL,SARGL,CP),
          FN=HD X, ARGL=TL X, SARGL=ARGL,
          WHILE ARGL REPEAT DO COMP(HD ARGL), ARGL=TL ARGL END,
          IF OP=LOOKUP(FN,OPLIST) THEN
             (IF INTQ(OP) THEN RETURN ASS(OP)
              ELSE RETURN MAPX(OP,ASS)   ),
          COMP(FN),
        ASS(KCALL,LENGTH(SARGL))
        END END;
    GREFS=PROC(A),
        DO  IF MEMBER(A,SYSLIST) THEN
           DO A=VFROMS(SFROMID(A)),
           A=CONCATV(VECTOR(STAR),A), A=IDFROMS(SFROMV(A))
              END
           ELSE NIL,
        IF MEMBER(A,GLOBL) THEN NIL ELSE GLOBL=A:GLOBL,
        A  END END;
      ASS=PROC(), BEGIN(I,OP),
        NBYTE=NBYTE+NUMARGS(),
        FOR I=(1,NUMARGS()) REPEAT END2=ADDON(END2,
           ARGUMENT(I))
        END END;
      LBL=PROC(X), LBLVALS=(X:NBYTE:NIL):LBLVALS END;
   GENLBL=PROC(), LBLNO=LBLNO+1 END;
   COMMENT
(**************************************************************
                          CODE GENERATORS
**************************************************************)
  GLAMBDA=PROC(X),
        BEGIN($ARGS,$VARS,EXPX,SARGS,$LBLIST),
        VARS=NIL, LBLIST=NIL,
        ARGS=HD TL X, EXPX=HD TL TL X,
        IF  HD ARGS EQ =GLOBAL  THEN ARGS=ARGS:NIL.
        SARGS=ARGS, ARGS=EXCHANGE(SARGS,1),
        COMP(EXPX),
        EXCHANGE(SARGS,1),
        ASS(KRETPROC)
        END END;
EXCHANGE=PROC(L,I), IF NULL(L) THEN NIL
        ELSEIF ¬PAIRQ(HD L) THEN HD L : EXCHANGE(TL L.I+1)
        ELSE DO ASS(KARG,I), ASS(KGLOB,O,HD TL HD L),
        ASS(KASTORE,I), ASS(KPOP,1),
        ASS(KGSTORE,O,HD TL HD L), ASS(KPCP,1),
        I : EXCHANGE(TL L,I+1) END END;
    GPROG=PROC(X),
        BEGIN($VARS,PROGRAM,T1,$LBLIST,$RET,SVARS),
        IF HD VARS EQ =GLOBAL THEN VARS=VARS:NIL,
        VARS=HD TL X, PROGRAM=TL TL X,
        ASS(KNVARS,LENGTH(VARS)),
        SVARS=VARS, VARS=SAVLOCS(SVARS,1),
        RET=GENLBL(),
        X=PROGRAM,
          WHILE PROGRAM REPEAT DO
             T1=HD PROGRAM, PROGRAM=TL PROGRAM,
             IF ¬PAIRQ(T1) THEN LBLIST=T1:LBLIST ELSE NIL END,
          PROGRAM=X,
          WHILE PROGRAM REPEAT DO
             T1=HD PROGRAM, PROGRAM=TL PROGRAM,
             IF ¬PAIRQ(T1) THEN LBL(T1)
                ELSE DO ASS(KSETSTK), COMP(T1) END END.
          ASS(KNIL), LBL(RET),
          RESTLOCS(SVARS,1),
          ASS(KRETPROG)
          END END;
  SAVLOCS=PROC(L,I), IF NULL(L) THEN NIL
          ELSEIF ¬PAIRQ (HD L) THEN HD L : SAVLOCS(TL L.I+1)
          ELSE DO ASS(KGLOB,0,HD TL HD L), ASS(KVSTORE,I),
          I:SAVLOCS(TL L,I+1) END END;
RESTLOCS=PROC(L,I), IF NULL(L) THEN NIL
          ELSEIF ¬PAIRQ(HD L) THEN RESTLOCS(TL L,I+1)
          ELSE DO ASS(KVAR,I), ASS(KGSTORE,0,HD TL HD L),
             ASS(KPOP,1), RESTLOCS(TL L,I+1) END END;
 GRETURN=PROC(X),
          DO COMP(HD TL X), ASS(KJMP,0,LIST(RET)) END END;
  GPROGN=PROC(L), BEGIN(E),
          L=TL L, COMP(HD L), L=TL L,
          WHILE L REPEAT DO
             E=HD L, L=TL L, ASS(KPOP,1), COMP(E) END
          END END;
     GGO=PROC(X), BEGIN(ARG),
          ARG=HD TL X,
          IF MEMBER(ARG,LBLIST) THEN ASS(KJMP,0,LIST(ARG))
          ELSE DO COMP(ARG), ASS(KJMP1) END
          END END;
   GCOND=PROC(X), BEGIN(P,E,NTRUE,LAST),
          LAST=GENLBL(),
          X=TL X,
          WHILE X REPEAT DO
             E=HD X, X=TL X, P=HD E, E=HD TL E,
             IF P EQ TRUE THEN DO COMP(E), X=NIL END
             ELSEIF HD E EQ=GO AND MEMBER(HD TL P,LBLIST) THEN
                DO COMP(P), ASS(KJMPT,0,LIST(HD TL E)) END
             ELSE DO COMP(P), NTRUE=GENLBL(),
                ASS(KJMPF,0,LIST(NTRUE)), COMP(E),
                IF HD E NE=GO AND HD E NE =RETURN AND X THEN
                   ASS(KJMP,0,LIST(LAST)) ELSE NIL,
                LBL(NTRUE)  END  END,
        LBL(LAST)
        END END;
   GAND=PROC(X), BEGIN(L),
        L=GENLBL(), ASS(KNIL),
        COMP(HD TL X) ,ASS(KJMPF,0,LIST(L)),
        ASS(KPOP,1), COMP(HD TL TL X),
        LBL(L)  END END;
    GOR=PROC(X), BEGIN(L),
        L=GENLBL(), ASS(KTRUE),
        COMP(HD TL X) ,ASS(KJMPT,0,LIST(L)),
        ASS(KPOP,1), COMP(HD TL TL X),
        LBL(L)  END END;
 GQUOTE=PROC(X), BEGIN(ARG),
        ARG=HD TL X,
        GCON(ARG)
        END END;
  GSETQ=PROC(X), BEGIN(ATM,VAL),
        ATM=HD TL X, VAL=HD TL TL X,
        COMP(VAL), ASSIGN(ATM)
        END END;
 ASSIGN=PROC(ATM), BEGIN(X),
        IF ¬IDQ(ATM) THEN DO
           ERRCNT=ERRCNT+1,
           PRINT(ATM:==:VAL:NIL),
           PRINT(=ASSIGN:=ERROR:NIL),
           RETURN NIL
           END,
        IF X=ORDINAL(ATM,ARGS) THEN ASS(KASTORE,X)
        ELSEIF X=ORDINAL(ATM,VARS) THEN ASS(KVSTORE,X)
        ELSE ASS(KGSTORE,0,GREFS(ATM))
        END END;
 GWHILE=PROC(X), BEGIN(MORE,NTRUE,P,E),
        X=TL X,
         X=HD X, P=HD X, E=HD TL X,
         ASS(KNIL),
         MORE=GENLBL(),
        LBL(MORE),
        COMP(P),
         NTS(KNIL),
         MORE=GENLBL(),
        LBL(MORE),
        COMP(P),
         NTRUE=GENLBL(),
         ASS(KJMPF,0,LIST(NTRUE)),
         ASS(KPOP,1),
         COMP(E),
          ASS(KJMP,0,LIST(MORE)),
         LBL(NTRUE)
          END END;
    GFOR=PROC(X),
         BEGIN(E,I,J,K,L,LAST,FORL),
         X=HD TL X, L=TL HD X, E=HD TL X,
         I=HD L, L=HD TL L, J=HD L, K=HD TL L, L=TL TL L,
         IF NULL(L) THEN L=1 ELSE L=HD L,
         LAST=GENLBL(),
         COMP(K), COMP(L), COMP(J),
         ASS(KNIL), ASS(KPOP,1),
         FORL=GENLBL(), LBL(FORL),
         ASSIGN(I),
         ASS(KTLOOP,0,LIST(LAST)),
         COMP(E),
         ASS(KSTEPLOOP,0,LIST(FORL)),
         LBL(LAST)
         END END;
   GLIST=PROC(X), BEGIN(),
         X=TL X, MAPX(X,COMP),
         GCON(LENGTH(X)), ASS(KLIST)  END END;
 GVECTOR=PROC(X), BEGIN(),
         X=TL X, MAPX(X,COMP),
         GCON(LENGTH(X)), ASS(KVECTOR)  END END;
```