

DATA

LINK L1 TO ANY, L2 TO ANY, L3 TO ANY
TYPE CONS IS
CAR TO ANY
CDR TO ANY
STRING LNIL IS ↓NIL↓

END

ROUTINE FLATTEN(X TO ANY) TO STRING IS NAME(X)

ROUTINE LIST.STR(X TO ANY, FLAG TO BOOLEAN\FALSE) TO STRING IS

LINK T TO ANY

IF X=LNIL THEN RETURN ↓()↓

IF TYPE.OF(X)=CONS THEN RETURN FLATTEN(X)

RESULT → LIST.STR(CAR OF X) ; T → CDR OF X

IF T=LNIL THEN GOTO EXIT

≥
≥
IF TYPE.OF(T)=CONS THEN RESULT → RESULT ^ ↓ . ↓ ^ ≥
FLATTEN(T) ; GOTO EXIT

RESULT → RESULT ^ ↓ ↓ ^ LIST.STR(T, TRUE)

EXIT: IF NOT FLAG THEN RESULT → ↓(↓ ^ RESULT ^ ↓)↓

RETURN

END

≤

ROUTINE TOKEN.LIST(NXT.TOK TO ROUTINE) TO ANY IS

≤ NEXT.TOK IS A PARAMETERLESS ROUTINE EXPECTED TO RETURN ONE OF:

≤ ↓(↓ -- BEGINNING OF LIST

≤ ↓)↓ -- END OF LIST

≤ ↓.↓ -- DELIMITER FOR DOTTED PAIR

≤ ANYTHING ELSE -- AN ATOM

≤

≤ THE LANGUAGE ACCEPTED BY TOKEN.LIST IS THAT OF <S.EXP> BELOW:

≤ <S.EXP> ::= <ATOM> ≤ () ≤ (<EXPR.SEQ>)

≤ <EXPR.SEQ> ::= <S.EXPR> ≤ <S.EXPR> . <S.EXPR>

≤ ≤ <S.EXPR> <EXPR.SEQ>

≤

ROUTINE TOKEN.LIST0

LINK T → NXT.TOK()

IF (T=↓.↓) OR (T=↓)↓) THEN GOTO ERROR

IF T=↓(↓ THEN RETURN T

RETURN TOKEN.LIST0(NXT.TOK)\ERROR

END

≤

ROUTINE TOKEN.LIST0(NXT.TOK TO ROUTINE) TO ANY IS

LINK T → NXT.TOK(), L TO CONS

IF T=↓)↓ THEN RETURN LNIL

IF T=↓.↓ THEN GOTO ERROR

RESULT → CREATE(CONS) ; L → RESULT

LOOP: IF T=↓(↓ THEN

CAR OF L → TOKEN.LIST0(NXT.TOK)\ERROR

ELSE

CAR OF L → T

ENDIF

T → NXT.TOK()

IF T=↓)↓ THEN CDR OF L → LNIL ; RETURN

≥
≥
IF T=↓.↓ THEN CDR OF L → CREATE(CONS) ; L → CDR OF L ;
GOTO LOOP

CDR OF L → TOKEN.LIST0(NXT.TOK)\ERROR

IF NXT.TOK()=↓.↓ THEN RETURN ELSE GOTO ERROR

```
ROUTINE CLEARNODE (ND TO NODE) IS  
LINK S  
FOR S IN SEL (ND) DO DETACH (S, ND)  
END
```

```

DATA IN P11,P1
TYPE NODESET IS
(NODE1 TO INTEGER
NODESET ALREADY
LINK 1 TO INTEGER → 0
END
ROUTINE P11(T) TO STRING IS
LINK SELS
LINK P.S
IF T FROM ALREADY THEN
IF NAME(T) = ↓NODE↓ THEN
RETURN ↓NODE↓ ^ NAME(T OF ALREADY)
ELSE
RETURN NAME(T)
ENDIF
ENDIF
IF (T IS NODE) AND (T ∈ I) THEN
I → I+1
ATTACH(T,ALREADY)
T OF ALREADY → I
ENDIF
IF T IS ATOM THEN RETURN NAME(T)
IF T IS LINK THEN RETURN ↓S↓ ^ P11(S↑)
IF T IS SEQ THEN
I → I+1
FOR S ∈ I DO
IF S ∈ C OF FIRST OF T THEN R → R^S↓
R → P11(S)
ENDFOR
RETURN R^I↓
ENDIF
I → I+1
SELS → SEL(T)
FOR S ∈ IN SELS DO
IF S ∈ C OF FIRST OF SELS THEN R → R^S↓
R → P11(S OF T)
ENDFOR
RETURN R^I↓
END
ROUTINE P1(T) IS
LEARNODE(ALREADY)
→ 0
END

```

```

DATA IN P12,P2
TYPE NODESET IS
[ NODE] TO INTEGER
NODESET ALREADY
LINK T TO INTEGER →
END
ROUTINE P12(T) TO STRING IS
LINK SELS
LINK P.S
IF T FROM ALREADY THEN
IF NAME(T) = ↓NODE↓ THEN
RETURN ↓NODE↓ ^ NAME(T OF ALREADY)
ELSE
RETURN NAME(T)
ENDIF
ENDIF
IF (T IS NODE) AND (TEND) THEN
I → I + 1
ATTACH(T,ALREADY)
T OF ALREADY → I
ENDIF
IF T IS ATOM THEN RETURN NAME(T)
IF T IS LINK THEN RETURN ↓↓↓ ^ P12(↓T)
IF T IS SEQ THEN
R → ↓(↓
FOR S IN T DO
IF S ∈ C OF FIRST OF T THEN R → R ^ ↓↓
R → R ^ P12(S)
ENDFOR
RETURN R ^ ↓↓
ENDIF
R → ↓(↓
SELS → SEL(T)
FOR S → IN SELS DO
IF S ∈ C OF FIRST OF SELS THEN R → R ^ ↓↓
R → R ^ P12(S)
IF S OF T ∈ END THEN
R → R ^ ↓ TO ↓ ^ P12(S OF T)
ENDIF
ENDFOR
RETURN R ^ ↓↓
END
CLEARNODE(ALREADY)
I → 0
PRINT(P12(T))
END

```

ROUTINE P12(T) IS

?

```
ROUTINE PACK(S TO SEQ) TO STRING IS  
LINK T  
RESULT → ↓↓  
FOR T → IN S DO  
IF T IS STRING THEN RESULT → RESULT ^ T ELSE ERROR  
ENDFOR  
END
```

```
ROUTINE SCAN() TO SEQ IS
LINK CH TO STRING, WORD TO STRING P++
RESULT ← CREATE (SEQ)
FOR CH IN UNPACK(READ()) DO
IF CH = ' ' THEN
IF WORD ≠ '' THEN ADDL(WORD, RESULT); WORD = ''
ELSE
WORD ← WORD ^ CH
ENDIF
ENDIFOR
END
```

ROUTINE ADD (A, S TO SEQ) IS
LINK T TO LISTEL
T ← FIRST OF S
FIRST OF S → CREATE (LISTEL)
C OF FIRST OF S → A
N OF FIRST OF S → T
IF T = UND THEN LAST OF S → FIRST OF S
END

ROUTINE DELF(S TO SEQ) TO ANY IS
IF FIRST OF S = UND THEN BREAK
RESULT → C OF FIRST OF S
FIRST OF S → N OF FIRST OF S
END

```
ROUTINE ADDL(A,S TO SEQ) IS
IF LAST OF S = END THEN
N OF LAST OF S → CREATE(LISTEL)
LAST OF S → N OF LAST OF S
C OF LAST OF S → A
ELSE
LAST OF S → CREATE ( LISTEL)
FIRST OF S → LAST OF S
C OF LAST OF S → A
ENDIF
```

ROUTINE DELL(S TO SEQ) TO ANY IS
LINK T TO LISTEL
IF FIRST OF S = UND THEN ERROR
FOR T → FIRST OF S • N OF T WHILE N OF T ≠ LAST OF S DO NO•05()
RESULT → C OF LAST OF S
LAST OF S → T
END