```
2000                    ORG     2000H   ;publication origin.
2000            TCORG   ORG     ($+0FFH)/100H*100H
                ;error codes
0001 =          STATERR EQU     1
0002 =          CURSERR EQU     2
0003 =          SYMERR  EQU     3
0005 =          RPARERR EQU     5
0006 =          RANGERR EQU     6
0007 =          CLASERR EQU     7
0009 =          SYNXERR EQU     9
000E =          LVALERR EQU     14
0010 =          PUSHERR EQU     16
0011 =          TMFUERR EQU     17
0012 =          TMVRERR EQU     18
0013 =          TMVLERR EQU     19
0014 =          LINKERR EQU     20
0015 =          ARGSERR EQU     21
0016 =          LBRCERR EQU     22
0018 =          MCERR   EQU     24
001A =          SYMERRA EQU     26
0063 =          KILL    EQU     99
                ;recognition length of symbols
0008 =          VLEN    EQU     8
                ;where tc exits to
0000 =          TCEXIT  EQU     0
                ;end-of-line character
000D =          ASCRET  EQU     0DH
                ;
                ;entry points
2000 C37B2C             JMP     COLD
2003 C3A12C             JMP     WARM
2006 C3A42C             JMP     HOT
                ;tailoring vector
2009 00         ECHO    DB      0       ;zero suppresses char echo
```

```
U200A C30000    INCH    JMP     XINCH
U200D C30000    OUTCH   JMP     XOUTCH
U2010 C30000    CHRDY   JMP     XCHRDY
U2013 C30000    FOPEN   JMP     XFOPEN
U2016 C30000    FREAD   JMP     XFREAD
U2019 C30000    FWRITE  JMP     XFWRITE
U201C C30000    FCLOSE  JMP     XFCLOSE
U201F C30000    USERMC  JMP     XUSERMC
 2022 00        PRBEGIN NOP
 2023 00                NOP
 2024 C9                RET
 2025 00        STBEGIN NOP
 2026 00                NOP
 2027 C9                RET
 2028 00        PRDONE  NOP
 2029 00                NOP
 202A C9                RET
                ;MC tools
 202B C34B2F    XMCESET JMP     MCESET
 202E C38421    XTOPTOI JMP     TOPTOI
 2031 C3C921    XPUSHK  JMP     PUSHK
 2034 00        MCARGS  DB      0
                ;escape character
 2035 1B        ESCAPE  DB      1BH
              ;space allocation
                ;these definitions let the assembler do the work. Space
                ;  may be hand allocated as described in Section 6.5.2.
                ;  BFREE is the first free page after the tiny c interpreter,
                ;  SPACE is EFREE minus BFREE.  Both are
                ;  defined at the end of this listing.
 5D00 =         EFREE   EQU     5D00H   ;Highest usable RAM.
                ;
 2036 0031      BSTACK  DW      BFREE
 2038 85CE      ESTACK  DW      -BFREE-80H+5
 203A 8031      BFUN    DW      BFREE+80H
 203C 06CE      EFUN    DW      -BFREE-100H+6
 203E 0032      BVAR    DW      BFREE+100H
 2040 8EC8      EVAR    DW      -BFREE-100H-SPACE/8+VLEN+6
```

```
2042 8037          BPR     DW      BFREE+100H+SPACE/8
2044 00A6          EPR     DW      -EFREE+300H     ;SAVE 3 PAGES FOR 8080 STACK
2046 005D          MSTACK  DW      EFREE
                   ;standard cells
2048 0000          ERR     DW      0
204A 0000          ERRAT   DW      0
204C 00            LEAVE   DB      0
204D 00            BRAKE   DB      0
204E 0000          TOP     DW      0
2050 0000          NXTVAR  DW      0
2052 0000          CURFUN  DW      0
2054 0000          CURGLBL DW      0
2056 0000          FNAME   DW      0
2058 0000          LNAME   DW      0
205A 0000          STCURS  DW      0
205C 0000          CURSOR  DW      0
205E 0000          PRUSED  DW      0
2060 0000          PROGEND DW      0       ;stored negative
2062 00            APPLVL  DB      0
                   ;
                   ;literals
2063 =             BALPHS  EQU     $       ;beginning of alphabetics
2063 696600        XIF     DB      'if',0
2066 656C736500    XELS    DB      'else',0
206B 696E7400      XINT    DB      'int',0
206F 6368617200    XCHAR   DB      'char',0
2074 7768696C65    XWHI    DB      'while',0
207A 7265747572    XRET    DB      'return',0
2081 627265616B    XBRK    DB      'break',0
2087 656E646C69    XENDL   DB      'endlibrary',0
2092 72            XR      DB      'r'     ;loader 'read' command
2093 67            XG      DB      'g'     ;'go' command
2094 78            XX      DB      'x'     ;'exit' command
2095 FF                    DB      0FFH    ;end of alphabetics
2096 5B            LB      DB      '{'
2097 00                    DB      0
2098 5D            RB      DB      '}'
2099 00                    DB      0
```

```
209A 28         LPAR     DB      '('
209B 00                  DB      0
209C 29         RPAR     DB      ')'
209D 00                  DB      0
209E 2C         COMMA    DB      ','
209F 00                  DB      0
20A0 0D         NEWLINE  DB      ASCRET
20A1 00                  DB      0
20A2 2F         CMNT     DB      '/'
20A3 2A         XSTAR    DB      '*'
20A4 00                  DB      0
20A5 3B         SEMI     DB      ';'
20A6 00                  DB      0
20A7 25         XPCNT    DB      '%'
20A8 00                  DB      0
20A9 2F         XSLASH   DB      '/'
20AA 00                  DB      0
20AB 2B         XPLUS    DB      '+'
20AC 00                  DB      0
20AD 2D         XMINUS   DB      '-'
20AE 00                  DB      0
20AF 3C         LT       DB      '<'
20B0 00                  DB      0
20B1 3E         GT       DB      '>'
20B2 00                  DB      0
20B3 21         NOTEQ    DB      '!'
20B4 3D                  DB      '='
20B5 00                  DB      0
20B6 3D         EQEQ     DB      '='
20B7 3D         XEQ      DB      '='
20B8 00                  DB      0
20B9 3E         GE       DB      '>'
20BA 3D                  DB      '='
20BB 00                  DB      0
20BC 3C         LE       DB      '<'
20BD 3D                  DB      '='
20BE 00                  DB      0
20BF 0D         XNL      DB      ASCRET
```

```
20C0 00                 DB      0
                ;EQ performs an assignment of top into top-1. Top-1
                ;  must be an lvalue.
20C1 CD8421     EQ      CALL    TOPTOI  ;value into DE
20C4 D5                 PUSH    D       ;stuff to be assigned
20C5 CDAA21             CALL    POPST   ;where to assign
20C8 B7                 ORA     A
20C9 CACE20             JZ      EQ2     ;if class>0 set size=2
20CC 0E02               MVI     C,2
20CE 78         EQ2     MOV     A,B     ;must be lvalue
20CF FE4C               CPI     'L'
20D1 C2E020             JNZ     EQERR
20D4 EB                 XCHG            ;where -> HL
20D5 D1                 POP     D       ;stuff -> DE
20D6 73                 MOV     M,E     ;assign lo byte
20D7 0D                 DCR     C       ;size--
20D8 CAC921             JZ      PUSHK   ;call/ret, put result on stack
20DB 23                 INX     H
20DC 72                 MOV     M,D     ;hi byte
20DD C3C921             JMP     PUSHK   ;call/ret, put result on stack
20E0 CDF221     EQERR   CALL    ESET
20E3 0E                 DB      LVALERR
20E4 D1                 POP     D
20E5 C3C921             JMP     PUSHK   ;skip the assign part
                ;
                ;-(BC) -> BC
20E8 79         DNEG    MOV     A,C
20E9 2F                 CMA
20EA 4F                 MOV     C,A
20EB 78                 MOV     A,B
20EC 2F                 CMA
20ED 47                 MOV     B,A
20EE 03                 INX     B
20EF C9                 RET
                ;
                ;difference between two top values -> DE, setting Z, CY
20F0 CDA121     TOPDIF  CALL    POPTWO  ;hence fall into DSUB.
                ;
```

```
                        ; (DE) - (BC) -> DE
20F3 7B         DSUB    MOV     A,E
20F4 91                 SUB     C
20F5 5F                 MOV     E,A
20F6 7A                 MOV     A,D
20F7 98                 SBB     B
20F8 57                 MOV     D,A
20F9 B3                 ORA     E       ;Z now set, CY clear
20FA 7A                 MOV     A,D
20FB 07                 RLC             ;sign is now in CY
20FC C9                 RET
                        ;
                        ; (BC) + (DE) -> DE
20FD 79         DADD    MOV     A,C
20FE 83                 ADD     E
20FF 5F                 MOV     E,A
2100 78                 MOV     A,B
2101 8A                 ADC     D
2102 57                 MOV     D,A
2103 B3                 ORA     E       ;Z now set. CY cleared.
2104 7A                 MOV     A,D
2105 07                 RLC             ;Sign is now in CY, Z not hurt.
2106 C9                 RET
                        ;
                        ; (BC) * (DE) -> DE
2107 210000     DMPY    LXI     H,0
210A 79         DM2     MOV     A,C     ;test lo bit of BC
210B 0F                 RRC
210C D21021             JNC     DM3
210F 19                 DAD     D       ;add multiplier
2110 CD2021     DM3     CALL    BCRS    ;shift BC right
2113 C21821             JNZ     DM4     ;return if BC is 0
2116 EB                 XCHG            ;answer -> DE
2117 C9                 RET
2118 CD2921     DM4     CALL    DELS    ;shift multiplier left, return
211B C20A21             JNZ     DM2     ;  if zero.
211E EB                 XCHG
211F C9                 RET
```

```
                ;
                ; shift BC riht, setting Z if 0.
2120 AF         BCRS    XRA     A       ;zero CY flag
2121 78                 MOV     A,B
2122 1F                 RAR
2123 47                 MOV     B,A
2124 79                 MOV     A,C
2125 1F                 RAR             ;picks up carry left by hi byte
2126 4F                 MOV     C,A
2127 B0                 ORA     B
2128 C9                 RET
                ; shift DE left. Sets z iff (DE)==0.
2129 AF         DELS    XRA     A       ;zero CY flag
                ; rotate DE left, CY -> lo bit
212A 7B         RDEL    MOV     A,E     ;lo byte first
212B 17                 RAL
212C 5F                 MOV     E,A
212D 7A                 MOV     A,D
212E 17                 RAL             ;picks up carry left by lo byte
212F 57                 MOV     D,A
2130 B3                 ORA     E
2131 C9                 RET
                ;
                ; (DE) 7 (BC) -> DE, quotient in HL.
2132 7A         DREM    MOV     A,D     ;sign of result -> stack
2133 A8                 XRA     B
2134 F5                 PUSH    PSW
2135 7A                 MOV     A,D     ;make factors positive
2136 B7                 ORA     A
2137 FC7521             CM      DENEG
213A 78                 MOV     A,B
213B B7                 ORA     A
213C FCE820             CM      DNEG
213F 3E10               MVI     A,16    ;shift count -> stack
2141 F5                 PUSH    PSW
2142 EB                 XCHG            ;numerator -> HL
2143 110000             LXI     D,0     ;partial remainder -> DE
2146 CD8221     DR2     CALL    HLLS    ;divide loop.  Long left shift
```

```
2149 CD2A21             CALL    RDEL    ;  DEHL.
214C CA5C21             JZ      DR3
214F CD7D21             CALL    DCMP    ;test BC <= DE
2152 FA5C21             JM      DR3
2155 7D                 MOV     A,L     ;set lo bit of L, and subtract
2156 F601               ORI     1       ;  divisor from partial
2158 6F                 MOV     L,A     ;  remainder
2159 CDF320             CALL    DSUB
215C F1         DR3     POP     PSW     ;decrement shift count
215D 3D                 DCR     A
215E CA6521             JZ      DR4
2161 F5                 PUSH    PSW
2162 C34621             JMP     DR2
2165 F1         DR4     POP     PSW     ;put sign on quotient and rem
2166 F0                 RP
2167 CD7521             CALL    DENEG
216A EB                 XCHG
216B CD7521             CALL    DENEG
216E EB                 XCHG
216F C9                 RET
                ;
                ; (DE) / (BC) -> DE
2170 CD3221     DDIV    CALL    DREM
2173 EB                 XCHG
2174 C9                 RET
                ;
                ; -(DE) -> DE
2175 7A         DENEG   MOV     A,D
2176 2F                 CMA
2177 57                 MOV     D,A
2178 7B                 MOV     A,E
2179 2F                 CMA
217A 5F                 MOV     E,A
217B 13                 INX     D
217C C9                 RET
                ;
                ;double compare (DE) - (BC) changing neither, but
                ;  setting s, cy
```

```
                ;  Note that z is not set reliably.
217D 7B         DCMP    MOV     A,E
217E 91                 SUB     C
217F 7A                 MOV     A,D
2180 98                 SBB     B
2181 C9                 RET
                ;
                ;HL left shift
2182 29         HLLS    DAD     H
2183 C9                 RET
                ;
                ;  stack tools
                ;
                ;TOPTOI pops top of stack into DE, converting lvalue
                ;  to actual if necessary.
2184 CDAA21     TOPTOI  CALL    POPST   ;class in A, lvalue in B,
2187 32A021             STA     TPCLASS ;  size in C, stuff in DE
218A 78                 MOV     A,B
218B FE41               CPI     'A'
218D CA9421             JZ      TT2
2190 EB                 XCHG            ;fetch data
2191 5E                 MOV     E,M
2192 23                 INX     H
2193 56                 MOV     D,M
2194 0D         TT2     DCR     C       ;if size 1 and class 0 return
2195 C0                 RNZ             ;  lo byte, with sign propgated
2196 3AA021             LDA     TPCLASS ;  thru hi byte.
2199 B7                 ORA     A
219A C0                 RNZ
219B 7B                 MOV     A,E
219C 07                 RLC             ;propogate sign into D.
219D 9F                 SBB     A
219E 57                 MOV     D,A
219F C9                 RET
21A0 00         TPCLASS DB      0
                ;
                ;pops two from stack, top -> bc, next -> de.
21A1 CD8421     POPTWO  CALL    TOPTOI
```

```
21A4 D5              PUSH    D
21A5 CD8421          CALL    TOPTOI
21A8 C1              POP     B
21A9 C9              RET
                     ;
                     ;pops the stack into A, B, C, DE. New top in HL.
21AA 2A4E20    POPST LHLD    TOP
21AD 7E              MOV     A,M     ;class
21AE 23              INX     H
21AF 46              MOV     B,M     ;lvalue
21B0 23              INX     H
21B1 4E              MOV     C,M     ;size
21B2 23              INX     H
21B3 5E              MOV     E,M     ;stuff, lo-byte
21B4 23              INX     H
21B5 56              MOV     D,M     ;stuff, hi-byte
21B6 C5              PUSH    B
21B7 01F7FF          LXI     B,-9
21BA 09              DAD     B       ;decrement top by 5.
21BB C1              POP     B
21BC 224E20          SHLD    TOP
21BF C9              RET
                     ;
                     ;pushes constant 1.
21C0 110100    PONE  LXI     D,1
21C3 C3C921          JMP     PUSHK
                     ;pushes constant 0.
21C6 110000    PZERO LXI     D,0
                     ;pushes constant in DE
21C9 AF        PUSHK XRA     A       ;class 0
21CA 0641            MVI     B,'A'   ;actual
21CC 0E02            MVI     C,2     ;2 byte size
                     ;pushes class (A), lvalue (B), size (C), stuff (DE)
                     ;  onto stack.
21CE 2A4E20    PUSHST LHLD   TOP     ;add 5 to top.
21D1 D5              PUSH    D
21D2 110500          LXI     D,5
21D5 19              DAD     D
```

```
21D6 224E20             SHLD    TOP
21D9 EB                 XCHG
21DA 2A3820             LHLD    ESTACK
21DD 19                 DAD     D
21DE EB                 XCHG            ;top -> HL
21DF D1                 POP     D       ;restore stuff
21E0 DAED21             JC      PERR
21E3 77                 MOV     M,A
21E4 23                 INX     H
21E5 70                 MOV     M,B
21E6 23                 INX     H
21E7 71                 MOV     M,C
21E8 23                 INX     H
21E9 73                 MOV     M,E
21EA 23                 INX     H
21EB 72                 MOV     M,D
21EC C9                 RET
21ED CDF221     PERR    CALL    ESET
21F0 10                 DB      PUSHERR
21F1 C9                 RET
                ;
                ;  ESET sets ERR unless one is already set
21F2 3A4820     ESET    LDA     ERR
21F5 E3                 XTHL
21F6 B7                 ORA     A
21F7 CAFD21             JZ      ES2
21FA 23                 INX     H
21FB E3                 XTHL
21FC C9                 RET
21FD 7E         ES2     MOV     A,M
21FE 23                 INX     H
21FF E3                 XTHL
2200 324820             STA     ERR
2203 2A5C20             LHLD    CURSOR
2206 224A20             SHLD    ERRAT
2209 C9                 RET
                ;
                ;store 0's from (DE) thru (HL) inclusive
```

```
220A 0600       ZERO    MVI     B,0
                ;store (B) from (DE) thru (HL) inclusive
220C 7D         BZAP    MOV     A,L
220D 93                 SUB     E
220E 7C                 MOV     A,H
220F 9A                 SBB     D
2210 D8                 RC
2211 70                 MOV     M,B
2212 2B                 DCX     H
2213 C30C22             JMP     BZAP
                ;
                ;print string starting at (HL), terminated by null byte
2216 7E         PS      MOV     A,M
2217 B7                 ORA     A
2218 C8                 RZ
2219 CD0D20             CALL    OUTCH
221C 23                 INX     H
221D C31622             JMP     PS
                ;
                ;  SCAN TOOLS
                ;
                ;LIT is used to match literals. It advances the cursor
                ;  over blanks, then attempts a match with the literal.
                ;  DE points to the literal, which is terminated by a
                ;  null byte. On match, the cursor is advanced
                ;  beyond the matched text, and NZ is set. On no match
                ;  the cursor is not advanced (except over the initial
                ;  blanks), and Z is set. LIT is called often, so some
                ;  attention to speed is given, mainly by using inline
                ;  code for blanks and string matching.
2220 2A5C20     LIT     LHLD    CURSOR
2223 3E20               MVI     A,' '   ;trim blanks
2225 BE         LIT2    CMP     M
2226 C22D22             JNZ     LIT3
2229 23                 INX     H
222A C32522             JMP     LIT2
222D 225C20     LIT3    SHLD    CURSOR  ;capture cursor, in case no mch
2230 1A         LIT4    LDAX    D       ;char from literal
```

```
2231 B7              ORA     A
2232 CA3E22          JZ      MATCH   ;null signals end of literal
2235 BE              CMP     M       ;char from program
2236 13              INX     D
2237 23              INX     H
2238 CA3022          JZ      LIT4
223B AF              XRA     A       ;no match, return Zero
223C B7              ORA     A
223D C9              RET
223E 225C20  MATCH   SHLD    CURSOR  ;capture new cursor
2241 2F              CMA             ;return Not Zero
2242 B7              ORA     A
2243 C9              RET
                     ;
                     ;advances cursor over blanks. Puts cursor in HL.
2244 2A5C20  BLANKS  LHLD    CURSOR
2247 3E20            MVI     A,' '
2249 BE      LOOP    CMP     M
224A C25122          JNZ     BLOUT
224D 23              INX     H
224E C34922          JMP     LOOP
2251 225C20  BLOUT   SHLD    CURSOR
2254 C9              RET
                     ;
                     ;skips over balanced l-r delimiters, (assuming the
                     ;first 1 delimiter is already matched.) Tests that
                     ;cursor stays within program limits, and sets ERR and
                     ;doesn't advance cursor on violation.
2255 1601    SKIP    MVI     D,1     ;counter
2257 7E      SK2     MOV     A,M
2258 B8              CMP     B
2259 CA6B22          JZ      SKL     ;match left delimiter
225C B9              CMP     C
225D C26C22          JNZ     SKNEXT
2260 15              DCR     D       ;match right delimiter
2261 C26C22          JNZ     SKNEXT
2264 23              INX     H       ;all done, bump over last
2265 225C20          SHLD    CURSOR  ;  matched.
```

```
2268 37              STC
2269 3F              CMC             ;CY off on success
226A C9              RET
226B 14        SKL   INR     D
226C 23        SKNEXT INX    H       ;bump HL, test for overflow
226D EB              XCHG            ;cursor -> DE
226E E5              PUSH    H       ;make H safe
226F 2A6020          LHLD    PROGEND ;stored negative, so add
2272 19              DAD     D
2273 E1              POP     H
2274 EB              XCHG            ;now all reg's restored
2275 D25722          JNC     SK2
2278 CDF221          CALL    ESET
227B 02              DB      CURSERR
227C 37              STC             ;CY set on error
227D C9              RET
                ;
                ;tests if (A) is alphanumeric. Plus on yes.
227E FE30      ALNUM CPI     '0'
2280 F8              RM
2281 FE3A            CPI     '9'+1
2283 FA9922          JM      YESA
                ;tests if (A) is alpha. Plus on yes.
2286 FE41      ALPHA CPI     'A'
2288 F8              RM              ;not alpha
2289 FE5B            CPI     'Z'+1
228B FA9922          JM      YESA
228E FE61            CPI     'a'
2290 F8              RM
2291 FE7B            CPI     'z'+1
2293 FA9922          JM      YESA
2296 2F              CMA             ;not alpha, this sets Minus.
2297 B7              ORA     A
2298 C9              RET
2299 AF        YESA  XRA     A       ;set Plus.
229A C9              RET
                ;
                ;matches a variable or function name. Sets FNAME,
```

```
                ;  LNAME to first and last chars of the name. Returns
                ;  Not Zero on match, Zero on no match.
229B CD4422     SYMNAME CALL    BLANKS
229E 225620             SHLD    FNAME
22A1 7E                 MOV     A,M
22A2 CD8622             CALL    ALPHA
22A5 FAB822             JM      SY3
22A8 23         SY2     INX     H       ;is a symbol, find its end.
22A9 7E                 MOV     A,M
22AA CD7E22             CALL    ALNUM
22AD F2A822             JP      SY2
22B0 225C20             SHLD    CURSOR  ;just beyond symbol
22B3 2B                 DCX     H
22B4 225820             SHLD    LNAME   ;symbol end
22B7 C9                 RET
22B8 AF         SY3     XRA     A       ;no symbol, return Z
22B9 C9                 RET
                ;
                ;matches 3 kinds of constants, setting FNAME, LNAME as
                ;  in SYMNAME. Sets A to 0 on no match, 1,2,or 3 on mch
22BA CD4422     CONST   CALL    BLANKS
22BD 7E                 MOV     A,M     ;first char
22BE FE2B               CPI     '+'     ;test for number
22C0 CAD222             JZ      CN2
22C3 FE2D               CPI     '-'
22C5 CAD222             JZ      CN2
22C8 FE30               CPI     '0'
22CA FAEB22             JM      CN3
22CD FE3A               CPI     '9'+1
22CF F2EB22             JP      CN3
22D2 225620     CN2     SHLD    FNAME   ;number, cursor to fname
22D5 23         CN4     INX     H       ;find end
22D6 7E                 MOV     A,M
22D7 FE30               CPI     '0'
22D9 FAE122             JM      CN5
22DC FE3A               CPI     '9'+1
22DE FAD522             JM      CN4     ;is a digit, keep going
22E1 225C20     CN5     SHLD    CURSOR  ;not a digit
```

```
22E4 2B              DCX    H
22E5 225820          SHLD   LNAME
22E8 3E01            MVI    A,1      ;type 1 constant (integer)
22EA C9              RET
22EB FE22      CN3   CPI    '"'      ;test for quoted string
22ED C21923          JNZ    CN6
22F0 23              INX    H        ;quote found
22F1 225620          SHLD   FNAME    ;first char of string (quote
22F4 7E        CN7   MOV    A,M      ;  excluded
22F5 B7              ORA    A        ;ended by either null or "
22F6 CA0B23          JZ     CN8
22F9 DE22            SBI    '"'
22FB CA0B23          JZ     CN8
22FE 23              INX    H
22FF EB              XCHG            ;cursor check
2300 2A6020          LHLD   PROGEND
2303 19              DAD    D
2304 EB              XCHG
2305 D2F422          JNC    CN7
2308 C33F23          JMP    CNERR    ;cursor overflow
230B 77        CN8   MOV    M,A      ;end quote found, replace with
230C 2B              DCX    H        ;  a null.
230D 225820          SHLD   LNAME    ;last char of string
2310 3E02            MVI    A,2      ;constant of type 2 (char str)
2312 B7              ORA    A
2313 23              INX    H
2314 23              INX    H
2315 225C20          SHLD   CURSOR
2318 C9              RET
2319 FE27      CN6   CPI    27H      ;test for prime
231B C23D23          JNZ    CN9
231E 23              INX    H
231F 225620          SHLD   FNAME
2322 7E        CN12  MOV    A,M      ;scan for matching prime
2323 FE27            CPI    27H
2325 CA3523          JZ     CN11
2328 23              INX    H
2329 EB              XCHG            ;cursor check
```

```
232A 2A6020            LHLD    PROGEND
232D 19                DAD     D
232E EB                XCHG
232F D22223            JNC     CN12
2332 C33F23            JMP     CNERR
2335 3E03       CN11   MVI     A,3     ;found matching prime
2337 B7                ORA     A
2338 23                INX     H
2339 225C20            SHLD    CURSOR
233C C9                RET
233D AF         CN9    XRA     A       ;no match
233E C9                RET
233F CDF221     CNERR  CALL    ESET
2342 02                DB      CURSERR
2343 C9                RET
                ;
                ;skips over remarks and/or end-of-lines in any order.
2344 11A020     REM    LXI     D,NEWLINE
2347 CD2022            CALL    LIT
234A CA5A23            JZ      RE2
234D 7E         RE3    MOV     A,M     ;skip linefeeds
234E FE0A              CPI     0AH
2350 C24423            JNZ     REM
2353 23                INX     H
2354 225C20            SHLD    CURSOR
2357 C34423            JMP     REM
235A 11A220     RE2    LXI     D,CMNT
235D CD2022            CALL    LIT
2360 C8                RZ
2361 0601              MVI     B,1     ;comment found, skip its text
2363 0E0D              MVI     C,ASCRET
2365 CD5522            CALL    SKIP
2368 D8                RC              ;error check
2369 C34D23            JMP     RE3
                ;
                ;HL points to start of digit string. Converts to intger
                ;  leaving result in DE. Uses all digits, even if DE
                ;  overflows. First nondigit stops scan.
```

```
236C EB         ATON    XCHG            ;pointer into DE
236D 210000             LXI     H,0     ;answer developed here
2370 1A         AN2     LDAX    D       ;next ascii
2371 D630               SUI     48
2373 DA8923             JC      AN3     ;test for digit
2376 FE0A               CPI     10
2378 D28923             JNC     AN3
237B 44                 MOV     B,H     ;digit, set HL=10*HL+A
237C 4D                 MOV     C,L
237D 29                 DAD     H
237E 29                 DAD     H
237F 09                 DAD     B
2380 29                 DAD     H
2381 4F                 MOV     C,A
2382 0600               MVI     B,0
2384 09                 DAD     B
2385 13                 INX     D       ;bump pointer
2386 C37023             JMP     AN2
2389 EB         AN3     XCHG            ;answer -> DE
238A C9                 RET
                ;
                ;HL points to beginning of ascii integer, possibly
                ;  signed. Converts to integer and leaves value in DE.
238B 00         AISGN   DB      0       ;nonzero for -
238C AF         ATOI    XRA     A
238D 328B23             STA     AISGN
2390 7E         AI6     MOV     A,M     ;skip blanks
2391 FE20               CPI     ' '
2393 C29A23             JNZ     AI2
2396 23                 INX     H
2397 C39023             JMP     AI6
239A FE2D       AI2     CPI     '-'     ;test sign
239C C2A323             JNZ     AI3
239F 328B23             STA     AISGN   ;is -
23A2 23                 INX     H
23A3 FE2B       AI3     CPI     '+'
23A5 C2A923             JNZ     AI4
23A8 23                 INX     H
```

```
23A9 7E        AI4     MOV     A,M     ;skip more blanks
23AA FE20              CPI     ' '
23AC C2B323            JNZ     AI5
23AF 23                INX     H
23B0 C3A923            JMP     AI4
23B3 CD6C23    AI5     CALL    ATON    ;does the digits
23B6 3A8B23            LDA     AISGN   ;magnitude in DE
23B9 B7                ORA     A
23BA C8                RZ
23BB C37521            JMP     DENEG   ;computes negative and returns
               ;
               ;  SYMBOL TOOLS
               ;
               ;allocate reference in FUNB for variables of a function
23BE 2A5220    NEWFUN  LHLD    CURFUN
23C1 110600            LXI     D,6     ;bump CURFUN by 6
23C4 19                DAD     D
23C5 225220            SHLD    CURFUN
23C8 EB                XCHG            ;test too many active functions
23C9 2A3C20            LHLD    EFUN
23CC 19                DAD     D
23CD EB                XCHG
23CE D2D623            JNC     NF2
23D1 CDF221            CALL    ESET
23D4 11                DB      TMFUERR
23D5 C9                RET
23D6 3A5020    NF2     LDA     NXTVAR  ;init first and last var
23D9 77                MOV     M,A     ;fv lo byte
23DA D60E              SUI     6+VLEN
23DC 4F                MOV     C,A     ;lv lo byte -> C for now
23DD 3A5120            LDA     NXTVAR+1
23E0 23                INX     H
23E1 77                MOV     M,A     ;fv hi byte
23E2 DE00              SBI     0       ;picks up possible carry
23E4 23                INX     H
23E5 71                MOV     M,C     ;lv lo byte
23E6 23                INX     H
23E7 77                MOV     M,A     ;lv hi byte
```

```
23E8 3A5E20             LDA     PRUSED  ;now set up backup pointer
23EB 23                 INX     H
23EC 77                 MOV     M,A     ;bu lo byte
23ED 3A5F20             LDA     PRUSED+1
23F0 23                 INX     H
23F1 77                 MOV     M,A     ;bu hi bytv
23F2 C9                 RET             ;all done
                ;
                ;deallocate variables of last function.
23F3 2A5220     FUNDONE LHLD    CURFUN
23F6 7E                 MOV     A,M
23F7 325020             STA     NXTVAR  ;lo byte
23FA 23                 INX     H
23FB 7E                 MOV     A,M
23FC 325120             STA     NXTVAR+1
23FF 23                 INX     H
2400 23                 INX     H
2401 23                 INX     H
2402 7E                 MOV     A,M
2403 325E20             STA     PRUSED
2406 23                 INX     H
2407 7E                 MOV     A,M
2408 325F20             STA     PRUSED+1
240B 11F5FF             LXI     D,-11
240E 19                 DAD     D       ;subtract 5 for above INX's,
240F 225220             SHLD    CURFUN  ;  plus 5 more to pop FUNB.
2412 C9                 RET
                ;
                ;allocate a variable. Class in A, size in B, len in DE,
                ;  passed value in HL.
2413 00         CLASS   DB      0       ;temps used by newvar
2414 00         OBSIZE  DB      0
2415 0000       PASSED  DW      0
2417 0000       LEN     DW      0
2419 0000       FVAL    DW      0
241B 0000       KF      DW      0
                ;
241D 321324     NEWVAR  STA     CLASS
```

```
2420 78              MOV     A,B
2421 321424          STA     OBSIZE
2424 221524          SHLD    PASSED
2427 EB              XCHG
2428 221724          SHLD    LEN
242B 2A5020          LHLD    NXTVAR
242E CD7925          CALL    CANON   ;put canonical form of name
                ;                     into (NXTVAR). Leaves HL
                ;    pointing to last byte of NAME o VARB.
2431 23              INX     H       ;-> CLASS in VARB.
2432 3A1324          LDA     CLASS
2435 77              MOV     M,A
2436 23              INX     H       ;-> OBJSIZE in VARB.
2437 3A1424          LDA     OBSIZE
243A 77              MOV     M,A
243B 23              INX     H       ;-> LEN in VARB (2 bytes).
243C 3A1724          LDA     LEN
243F 77              MOV     M,A
2440 23              INX     H
2441 3A1824          LDA     LEN+1
2444 77              MOV     M,A
2445 23              INX     H
2446 221924          SHLD    FVAL    ;address where fval will be put
2449 3A1324          LDA     CLASS
244C B7              ORA     A       ;if class is 0, or not a passed
244D CA5824          JZ      NR2     ;  arg, then get value space.
2450 2A1524          LHLD    PASSED
2453 7D              MOV     A,L
2454 B4              ORA     H
2455 C29424          JNZ     NR3
2458 2A5E20    NR2   LHLD    PRUSED  ;get value space
245B 23              INX     H       ;  starting at PRUSED + 1
245C 221B24          SHLD    KF      ;Put in KF for later use.
245F EB              XCHG
2460 2A1924          LHLD    FVAL
2463 73              MOV     M,E
2464 23              INX     H
2465 72              MOV     M,D     ;fval part of varb set to
```

```
2466 2A1724          LHLD    LEN     ;  prused+1. Now bump prused
2469 EB              XCHG            ;  by obsize*len.
246A 2A5E20          LHLD    PRUSED
246D 3A1424          LDA     OBSIZE
2470 19              DAD     D
2471 3D              DCR     A
2472 CA7624          JZ      NR7
2475 19              DAD     D
2476 225E20    NR7   SHLD    PRUSED
2479 EB              XCHG            ;test if allocation exceeds
247A 2A4420          LHLD    EPR     ;  limits of prog space.
247D 19              DAD     D
247E EB              XCHG
247F D28724          JNC     NR4
2482 CDF221          CALL    ESET    ;RAM exceeded
2485 13              DB      TMVLERR
2486 C9              RET
2487 2A1B24    NR4   LHLD    KF      ;zero the allocated space
248A EB              XCHG
248B 2A5E20          LHLD    PRUSED
248E CD0A22          CALL    ZERO
2491 C3A324          JMP     NR5     ;end of space allocation
2494 2A1924    NR3   LHLD    FVAL    ;Value is passed and is a
2497 3A1524          LDA     PASSED  ;  class > 0. Put value in fval
249A 77              MOV     M,A     ;  part of VARB. Dont allocate
249B 23              INX     H       ;  space.
249C 3A1624          LDA     PASSED+1
249F 77              MOV     M,A
24A0 C3B924          JMP     NR6
24A3 3A1324    NR5   LDA     CLASS   ;if passed & class is 0 move
24A6 B7              ORA     A       ;  the passed value into the
24A7 C2B924          JNZ     NR6     ;  allocated space.
24AA 2A1524          LHLD    PASSED
24AD 7C              MOV     A,H
24AE B5              ORA     L
24AF CAB924          JZ      NR6
24B2 EB              XCHG            ;passed -> DE
24B3 2A1B24          LHLD    KF
```

```
24B6 73                 MOV     M,E     ;lo byte of passed value
24B7 23                 INX     H
24B8 72                 MOV     M,D     ;hi byte, or junk if only one
                ;                          byte passed. Who cares.
24B9 2A5220     NR6     LHLD    CURFUN  ;in FUNB set lvar part to this
24BC 23                 INX     H       ;  variable.
24BD 23                 INX     H
24BE 3A5020             LDA     NXTVAR
24C1 77                 MOV     M,A
24C2 23                 INX     H
24C3 3A5120             LDA     NXTVAR+1
24C6 77                 MOV     M,A
24C7 2A5020             LHLD    NXTVAR  ;increment NXTVAR
24CA 110E00             LXI     D,6+VLEN        ;  by 6 + vlen
24CD 19                 DAD     D
24CE 225020             SHLD    NXTVAR
24D1 EB                 XCHG            ;test if too many variables
24D2 2A4020             LHLD    EVAR
24D5 19                 DAD     D
24D6 EB                 XCHG
24D7 2A1924             LHLD    FVAL
24DA D0                 RNC             ;normal return, FVAL in HL.
24DB CDF221             CALL    ESET    ;VARB exceeded.
24DE 12                 DB      TMVRERR
24DF C9                 RET
                ;
                ;ADDRVAL looks up a symbol pointed to by FNAME,LNAME.
                ;  Returns address in HL, class in A, size in B, and
                ;  length in DE. Sets err if symbol cannot be found.
                ;  Searches 3 areas:
                ;       area    0       locals
                ;               1       globals
                ;               2       library symbols
24E0            NAME    DS      VLEN    ;holds canonical form of name
24E8 0000       PVAR    DW      0
24EA 00         AREA    DB      0
24EB 0000       SFUN    DW      0
24ED 0000       LAST    DW      0
```

```
                ;
24EF 2A5220     ADDRVAL LHLD    CURFUN
24F2 22EB24             SHLD    SFUN    ;search locals first
24F5 21E024             LXI     H,NAME
24F8 CD7925             CALL    CANON
24FB AF                 XRA     A
24FC 32EA24             STA     AREA    ;area 0
24FF 2AEB24     AD8     LHLD    SFUN    ;variable search area
2502 5E                 MOV     E,M
2503 23                 INX     H
2504 56                 MOV     D,M     ;fvar of search area -> DE
2505 23                 INX     H
2506 4E                 MOV     C,M
2507 23                 INX     H
2508 46                 MOV     B,M     ;lvar -> BC
2509 EB                 XCHG
250A 22E824             SHLD    PVAR    ;currently searched variable
250D 60                 MOV     H,B
250E 69                 MOV     L,C
250F 22ED24             SHLD    LAST    ;last to search in this area
2512 2AE824             LHLD    PVAR    ;begin search loop
2515 3AED24     AD2     LDA     LAST    ;test for end of loop
2518 95                 SUB     L
2519 3AEE24             LDA     LAST+1
251C 9C                 SBB     H
251D DA5325             JC      AD3
2520 0E08               MVI     C,VLEN  ;number of chars to match
2522 11E024             LXI     D,NAME  ;match string address
2525 1A         AD4     LDAX    D       ;(HL already as table entry)
2526 BE                 CMP     M
2527 C24625             JNZ     AD5     ;no match
252A 0D                 DCR     C
252B 13                 INX     D
252C 23                 INX     H
252D C22525             JNZ     AD4     ;next char
2530 7E                 MOV     A,M     ;MATCH. HL points to class.
2531 23                 INX     H
2532 46                 MOV     B,M     ;obsize
```

```
2533 23              INX     H
2534 5E              MOV     E,M
2535 23              INX     H
2536 56              MOV     D,M     ;length
2537 23              INX     H
2538 B7              ORA     A       ;if class > 0 & class < 'E'
2539 CA3F25          JZ      AD9     ;  then return address of fval
253C FE45            CPI     'E'     ;  part of VARB, which is alrdy
253E C0              RNZ             ;  in HL.
253F D5        AD9   PUSH    D       ;otherwise return contents of
2540 5E              MOV     E,M     ;  fval part of VARB.
2541 23              INX     H
2542 56              MOV     D,M
2543 EB              XCHG
2544 D1              POP     D
2545 C9              RET
2546 2AE824    AD5   LHLD    PVAR    ;go to next variable
2549 110E00          LXI     D,VLEN+6
254C 19              DAD     D
254D 22E824          SHLD    PVAR
2550 C31525          JMP     AD2
2553 3AEA24    AD3   LDA     AREA    ;go to next area
2556 B7              ORA     A
2557 C26725          JNZ     AD6
255A 2A5420          LHLD    CURGLBL ;second search area, globals
255D 22EB24    AD7   SHLD    SFUN
2560 3C              INR     A
2561 32EA24          STA     AREA
2564 C3FF24          JMP     AD8
2567 FE02      AD6   CPI     2
2569 F27225          JP      ADERR
256C 2A3A20          LHLD    BFUN    ;third area is library, which
256F C35D25          JMP     AD7     ;  is at beginning of FUNB.
2572 CDF221    ADERR CALL    ESET
2575 1A              DB      SYMERRA
2576 C9              RET
                ;
                ;canonicalizes symbol from FNAME to LNAME inclusive,
```

```
                ;  putting form with VLEN chars in (HL).
2577 0000       OUTNAME DW      0
2579 227725     CANON   SHLD    OUTNAME
257C 3E08               MVI     A,VLEN  ;zero output field
257E 0600               MVI     B,0
2580 48                 MOV     C,B     ;zero C for later
2581 70         CA2     MOV     M,B
2582 3D                 DCR     A
2583 CA8A25             JZ      CA3
2586 23                 INX     H
2587 C38125             JMP     CA2
258A E5         CA3     PUSH    H       ;save pointer to last byte
258B 2A5620             LHLD    FNAME   ;compute symbols actual length
258E 3A5820             LDA     LNAME
2591 95                 SUB     L
2592 3C                 INR     A
2593 FE08               CPI     VLEN
2595 FA9B25             JM      CA6
2598 3E08               MVI     A,VLEN  ;A now has number of chars to
259A 4F                 MOV     C,A     ;  be moved, and C is nonzero
259B EB         CA6     XCHG            ;  iff act len > VLEN.
259C 47                 MOV     B,A
259D 2A7725             LHLD    OUTNAME ;FNAME -> DE, OUTNAME -> HL
25A0 1A         CA4     LDAX    D       ;copy loop
25A1 77                 MOV     M,A
25A2 05                 DCR     B
25A3 CAAB25             JZ      CA5
25A6 13                 INX     D
25A7 23                 INX     H
25A8 C3A025             JMP     CA4
25AB E1         CA5     POP     H       ;pointer to last byte
25AC AF                 XRA     A
25AD B1                 ORA     C       ;test if short name
25AE C8                 RZ
25AF EB                 XCHG            ;long name, put last char in
25B0 2A5820             LHLD    LNAME   ;  the canon form.
25B3 7E                 MOV     A,M     ;last char of name
25B4 EB                 XCHG
```

```
25B5 77                 MOV     M,A     ;into last pos of outname
25B6 C9                 RET
                ;ASGN is the expression evaluator,so called because
                ;  the highest form of an expression is an assignment.
                ;  An asgn is a reln or an lvalue = asgn. Note that
                ;  reln can match an lvalue.
                ;Returns non-zero if valid expression, 0 if invalid.
25B7 CDD825     ASGN    CALL    RELN    ;stacked as lvalue if that's
                ;                         what it is.
25BA 11B720             LXI     D,XEQ   ; test for =
25BD CD2022             CALL    LIT
25C0 CACD25             JZ      A2
25C3 CDB725             CALL    ASGN
25C6 3A4820             LDA     ERR     ;check for error
25C9 B7                 ORA     A
25CA CCC120             CZ      EQ      ;perform assignment
25CD 3A4820     A2      LDA     ERR     ;return 0 (i.e. no match) if
25D0 B7                 ORA     A       ;  there was an error
25D1 CAD625             JZ      A3
25D4 AF                 XRA     A
25D5 C9                 RET
25D6 3D         A3      DCR     A       ;no error so return non-zero A
25D7 C9                 RET
                ;
                ;a RELN is an expr or a comparison of exprs
25D8 CD6026     RELN    CALL    EXPR
25DB 11BC20             LXI     D,LE    ; <=
25DE CD2022             CALL    LIT
25E1 CAF325             JZ      R2
25E4 CD6026             CALL    EXPR    ;right side
25E7 CDF020             CALL    TOPDIF  ;sets Z,C flags. C set as
25EA CAC021             JZ      PONE    ;  though it were S. Must be
25ED DAC021             JC      PONE    ;  zero or negative for true.
25F0 C3C621             JMP     PZERO   ;These jumps all call/rets.
25F3 11B920     R2      LXI     D,GE    ; >=
25F6 CD2022             CALL    LIT
25F9 CA0B26             JZ      R3
25FC CD6026             CALL    EXPR
```

```
25FF CDF020             CALL    TOPDIF
2602 CAC021             JZ      PONE
2605 D2C021             JNC     PONE
2608 C3C621             JMP     PZERO
260B 11B620     R3      LXI     D,EQEQ  ; ==
260E CD2022             CALL    LIT
2611 CA2026             JZ      R4
2614 CD6026             CALL    EXPR
2617 CDF020             CALL    TOPDIF
261A CAC021             JZ      PONE
261D C3C621             JMP     PZERO
2620 11B320     R4      LXI     D,NOTEQ
2623 CD2022             CALL    LIT
2626 CA3526             JZ      R5
2629 CD6026             CALL    EXPR
262C CDF020             CALL    TOPDIF
262F C2C021             JNZ     PONE
2632 C3C621             JMP     PZERO
2635 11B120     R5      LXI     D,GT    ; >
2638 CD2022             CALL    LIT
263B CA4D26             JZ      R6
263E CD6026             CALL    EXPR
2641 CDF020             CALL    TOPDIF
2644 CAC621             JZ      PZERO
2647 DAC621             JC      PZERO
264A C3C021             JMP     PONE
264D 11AF20     R6      LXI     D,LT    ; <
2650 CD2022             CALL    LIT
2653 C8                 RZ              ; no relational operator
2654 CD6026             CALL    EXPR
2657 CDF020             CALL    TOPDIF
265A DAC021             JC      PONE
265D C3C621             JMP     PZERO
                ;
                ;an EXPR is a term or sum (diff) of terms.
2660 11AD20     EXPR    LXI     D,XMINUS        ; unary -
2663 CD2022             CALL    LIT
2666 CA7C26             JZ      EX2
```

```
2669 CDB826             CALL    TERM
266C CD8421             CALL    TOPTOI  ;push negative of top back onto
266F 7B                 MOV     A,E
2670 2F                 CMA
2671 5F                 MOV     E,A
2672 7A                 MOV     A,D
2673 2F                 CMA
2674 57                 MOV     D,A
2675 13                 INX     D
2676 CDC921             CALL    PUSHK
2679 C38526             JMP     EX3
267C 11AB20     EX2     LXI     D,XPLUS ;optional unary +
267F CD2022             CALL    LIT
2682 CDB826             CALL    TERM
                ;first term is now stacked. Check for error so far.
2685 3A4820     EX3     LDA     ERR
2688 B7                 ORA     A
2689 C0                 RNZ
268A 11AB20             LXI     D,XPLUS ; +
268D CD2022             CALL    LIT
2690 CAA226             JZ      EX4
2693 CDB826             CALL    TERM
2696 CDA121             CALL    POPTWO  ;top two values on stack are
                ;                       actualized and put into
                ;                       (BC) and (DE).
2699 CDFD20             CALL    DADD    ; (BC)+(DE)->(DE)
269C CDC921             CALL    PUSHK   ; sum onto stack.
269F C38526             JMP     EX3     ;back for more terms
26A2 11AD20     EX4     LXI     D,XMINUS ; -
26A5 CD2022             CALL    LIT
26A8 C8                 RZ              ;no more terms
26A9 CDB826             CALL    TERM
26AC CDA121             CALL    POPTWO
26AF CDF320             CALL    DSUB
26B2 CDC921             CALL    PUSHK
26B5 C38526             JMP     EX3     ;back for more terms.
                ;
                ;a term is a factor or a product of factors.
```

```
26B8 CD0927    TERM    CALL    FACTOR
26BB 3A4820    TE2     LDA     ERR     ;check for error so far
26BE B7                ORA     A
26BF C0                RNZ
26C0 11A320            LXI     D,XSTAR ; *
26C3 CD2022            CALL    LIT
26C6 CAD826            JZ      TE3
26C9 CD0927            CALL    FACTOR
26CC CDA121            CALL    POPTWO
26CF CD0721            CALL    DMPY
26D2 CDC921            CALL    PUSHK
26D5 C3BB26            JMP     TE2     ;back for more factors.
26D8 CD4423    TE3     CALL    REM     ;make sure no /*
26DB 11A920            LXI     D,XSLASH        ; /
26DE CD2022            CALL    LIT
26E1 CAF326            JZ      TE4
26E4 CD0927            CALL    FACTOR
26E7 CDA121            CALL    POPTWO
26EA CD7021            CALL    DDIV
26ED CDC921            CALL    PUSHK
26F0 C3BB26            JMP     TE2
26F3 11A720    TE4     LXI     D,XPCNT ; 30
26F6 CD2022            CALL    LIT
26F9 C8                RZ              ;no more factors.
26FA CD0927            CALL    FACTOR
26FD CDA121            CALL    POPTWO
2700 CD3221            CALL    DREM
2703 CDC921            CALL    PUSHK
2706 C3BB26            JMP     TE2
                ;
                ;a FACTOR is a ( asgn ), or a constant, or a variable
                ;  reference, or a function reference.
2709 119A20    FACTOR  LXI     D,LPAR  ; (
270C CD2022            CALL    LIT
270F CA2127            JZ      FA2
2712 CDB725            CALL    ASGN
2715 119C20            LXI     D,RPAR  ; )
2718 CD2022            CALL    LIT
```

```
271B C0              RNZ
271C CDF221          CALL    ESET    ;right paren error
271F 05              DB      RPARERR
2720 C9              RET
2721 CDBA22    FA2   CALL    CONST   ;recognizes 3 types of constant
2724 CA5327          JZ      FA5     ;  setting A accordingly.
2727 FE01            CPI     1
2729 C23527          JNZ     FA3
272C 2A5620          LHLD    FNAME   ;type 1: integer. FNAME points
272F CD8C23          CALL    ATOI    ;  to beginning. ATOI converts
2732 C3C921          JMP     PUSHK   ;  it, leaving value in (DE).
2735 FE02      FA3   CPI     2
2737 C24727          JNZ     FA4
273A 3E01            MVI     A,1     ;type 2: char string. Push
273C 0641            MVI     B,'A'   ;  class=1, lval='A', size=1,
273E 0E01            MVI     C,1     ;  and stuff=address of
2740 2A5620          LHLD    FNAME   ;  beginning of string.
2743 EB              XCHG
2744 C3CE21          JMP     PUSHST
2747 AF        FA4   XRA     A       ;type 3: char constant. Push
2748 0641            MVI     B,'A'   ;  class=0, lval='A', size=1,
274A 0E01            MVI     C,1     ; and stuff=actual character.
274C 2A5620          LHLD    FNAME
274F 5E              MOV     E,M
2750 C3CE21          JMP     PUSHST
2753 CD9B22    FA5   CALL    SYMNAME ;not a constant, try symbol.
2756 CA3128          JZ      FA6
2759 2A5620          LHLD    FNAME   ;symbol. Test for special
275C 23              INX     H       ;  symbol MC. First is symbol
275D 3A5820          LDA     LNAME   ;  length exactly 2.
2760 BD              CMP     L
2761 C27E27          JNZ     FA7
2764 3A5920          LDA     LNAME+1
2767 BC              CMP     H
2768 C27E27          JNZ     FA7
276B 7E              MOV     A,M     ;length is 2, and (HL)=FNAME.
276C FE43            CPI     'C'
276E C27E27          JNZ     FA7
```

```
2771 2B              DCX    H
2772 7E              MOV    A,M
2773 FE4D            CPI    'M'
2775 C27E27          JNZ    FA7
2778 210000          LXI    H,0
277B C3A32A          JMP    ENTER    ;causes machine call.
277E CDEF24    FA7   CALL   ADDRVAL  ;not MC, look up symbol.
2781 223628          SHLD   FWHERE
2784 321324          STA    CLASS
2787 78              MOV    A,B      ;save results of lookup.
2788 321424          STA    OBSIZE
278B EB              XCHG
278C 221724          SHLD   LEN
278F 7A              MOV    A,D      ;where is now in DE
2790 B3              ORA    E
2791 CA2C28          JZ     FA8
2794 3A1324          LDA    CLASS
2797 FE45            CPI    'E'      ;class E => function entry
2799 CA2628          JZ     FA9
279C 119A20          LXI    D,LPAR   ;variable. Test for subscript.
279F CD2022          CALL   LIT
27A2 CA1628          JZ     FA10
27A5 3A1324          LDA    CLASS    ;subscripted, class must be > 0
27A8 3D              DCR    A
27A9 321324          STA    CLASS    ;class of element is one less
27AC F2B427          JP     FA11     ;  than class of array.
27AF CDF221          CALL   ESET
27B2 07              DB     CLASERR
27B3 C9              RET
27B4 2A3628    FA11  LHLD   FWHERE   ;replace where by two bytes
27B7 5E              MOV    E,M      ;  referenced by where.
27B8 23              INX    H
27B9 56              MOV    D,M
27BA D5              PUSH   D        ;save where, len, class,
27BB 2A1724          LHLD   LEN      ;  obsize.
27BE E5              PUSH   H
27BF 2A1324          LHLD   CLASS    ;(also gets obsize)
27C2 E5              PUSH   H
```

```
27C3 CDB725            CALL    ASGN    ;evaluate subscript
27C6 E1                POP     H
27C7 221324            SHLD    CLASS   ;restore everything
27CA E1                POP     H
27CB 221724            SHLD    LEN
27CE E1                POP     H
27CF 223628            SHLD    FWHERE
27D2 C8                RZ              ;assign error
27D3 119C20            LXI     D,RPAR  ;skip )
27D6 CD2022            CALL    LIT
27D9 CD8421            CALL    TOPTOI  ;subscript value -> DE
27DC EB                XCHG
27DD 223828            SHLD    SUBSCR
27E0 EB                XCHG
27E1 2A1724            LHLD    LEN
27E4 7D                MOV     A,L
27E5 3D                DCR     A
27E6 B4                ORA     H       ;for LEN = 1 skip subscript
27E7 CA0128            JZ      FA12    ;  check.
27EA 3A1324            LDA     CLASS
27ED B7                ORA     A
27EE C20128            JNZ     FA12    ;skip for pointers, too.
27F1 B2                ORA     D
27F2 FAFD27            JM      SUBERR  ;cant be negative
27F5 44                MOV     B,H     ;len -> BC
27F6 4D                MOV     C,L
27F7 CDF320            CALL    DSUB
27FA DA0128            JC      FA12    ;subscr-len must be negative
27FD CDF221    SUBERR  CALL    ESET
2800 06                DB      RANGERR
2801 2A3828    FA12    LHLD    SUBSCR
2804 EB                XCHG            ;where =+ subscr * obsize
2805 2A3628            LHLD    FWHERE
2808 3A1424            LDA     OBSIZE
280B 3D        FA13    DCR     A
280C FA1328            JM      FA14
280F 19                DAD     D
2810 C30B28            JMP     FA13
```

```
2813 223628    FA14    SHLD    FWHERE
2816 3A1424    FA10    LDA     OBSIZE  ;push class, 'L', obsize,
2819 4F                MOV     C,A     ;  stuff=where.
281A 3A1324            LDA     CLASS
281D 064C              MVI     B,'L'
281F 2A3628            LHLD    FWHERE
2822 EB                XCHG
2823 C3CE21            JMP     PUSHST  ;call/ret
2826 2A3628    FA9     LHLD    FWHERE
2829 C3A32A            JMP     ENTER   ;call/ret
282C CDF221    FA8     CALL    ESET    ;symbol error
282F 03                DB      SYMERR
2830 C9                RET
2831 CDF221    FA6     CALL    ESET    ;cannot recognize factor
2834 09                DB      SYNXERR
2835 C9                RET
               ;
               ;locals used by ASGN, etc.
2836 0000      FWHERE  DW      0
2838 0000      SUBSCR  DW      0
               ;SKIPST skips over a (possibly compound) statement,
               ;  including whole nested sets of if-then-elses.
               ;  Assumes balanced (), even within comments.
283A CD4423    SKIPST  CALL    REM
283D 119620            LXI     D,LB    ;test for [
2840 CD2022            CALL    LIT
2843 CA5028            JZ      SS2
2846 065B              MVI     B,'['
2848 0E5D              MVI     C,']'
284A CD5522            CALL    SKIP
284D C34423            JMP     REM     ;and done
2850 116320    SS2     LXI     D,XIF   ;test for if or while
2853 CD2022            CALL    LIT
2856 C26228            JNZ     SS6
2859 117420            LXI     D,XWHI
285C CD2022            CALL    LIT
285F CA7E28            JZ      SS3
2862 119A20    SS6     LXI     D,LPAR
```

```
2865 CD2022             CALL    LIT
2868 0628               MVI     B,'('
286A 0E29               MVI     C,')'
286C CD5522             CALL    SKIP    ;skip over (condition) part
286F CD3A28             CALL    SKIPST  ;skip then part
2872 116620             LXI     D,XELS  ;test for ELSE
2875 CD2022             CALL    LIT
2878 C43A28             CNZ     SKIPST  ;skip else part
287B C34423             JMP     REM     ;and done
287E 2A5C20     SS3     LHLD    CURSOR  ;simple statement, move cursor
2881 7E         SS4     MOV     A,M     ;  past next ; or to return or ).
2882 FE0D               CPI     ASCRET
2884 CA9F28             JZ      SS8
2887 FE5D               CPI     ']'
2889 CA9F28             JZ      SS8
288C FE3B               CPI     ';'
288E CA9E28             JZ      SS5
2891 23                 INX     H
2892 EB                 XCHG            ;test cursor overflow
2893 2A6020             LHLD    PROGEND
2896 19                 DAD     D
2897 EB                 XCHG
2898 D28128             JNC     SS4
289B C34423             JMP     REM     ;and done
289E 23         SS5     INX     H
289F 225C20     SS8     SHLD    CURSOR
28A2 C34423             JMP     REM     ;and done
                ;
                ;VALLOC parses one variable behind INT or CHAR and
                ;  makes allocation and symbol entry.
28A5 00         TYPE    DB      0       ;'C' or 'I'
28A6 0000       VPASSED DW      0       ;0 for global or local, two
                ;                        byte value if param to fnction
                ;       It turns out a 0 valued parameter gets the same
                ;       treatment as a local.
28A8 00         VCLASS  DB      0       ;defined in globals section.
28A9 0000       ALEN    DW      0       ;elements in an array.
                ;
```

```
28AB 32A528    VALLOC  STA     TYPE
28AE 22A628            SHLD    VPASSED
28B1 CD9B22            CALL    SYMNAME ;sets FNAME, LNAME around symbl
28B4 CA1429            JZ      V2      ;error if no symbol.
28B7 AF                XRA     A
28B8 32A828            STA     VCLASS  ;assume class 0 (not an array)
28BB 119A20            LXI     D,LPAR
28BE CD2022            CALL    LIT
28C1 CAF628            JZ      V3
28C4 2A5620            LHLD    FNAME   ;array, evaluate subscript
28C7 E5                PUSH    H       ;  expression. Must push FNAME,
28C8 2A5820            LHLD    LNAME   ;  LNAME, and class, because
28CB E5                PUSH    H       ;  subscripts may invoke
28CC 3AA828            LDA     VCLASS  ;  functions which themselves
28CF 3C                INR     A       ;  allocate variables.
28D0 F5                PUSH    PSW
28D1 CDB725            CALL    ASGN
28D4 F1                POP     PSW     ;restore pushed stuff.
28D5 32A828            STA     VCLASS
28D8 E1                POP     H
28D9 225820            SHLD    LNAME
28DC E1                POP     H
28DD 225620            SHLD    FNAME
28E0 3A4820            LDA     ERR     ;test for error in ASGN
28E3 B7                ORA     A
28E4 C0                RNZ
28E5 119C20            LXI     D,RPAR
28E8 CD2022            CALL    LIT     ;skip )
28EB CD8421            CALL    TOPTOI  ;value of subscript + 1 into
28EE 13                INX     D       ;  LEN
28EF EB                XCHG
28F0 22A928            SHLD    ALEN
28F3 C3FC28            JMP     V5
28F6 210100    V3      LXI     H,1     ;non-subscripted variable
28F9 22A928            SHLD    ALEN    ;  has ALEN 1.
28FC 3AA528    V5      LDA     TYPE    ;object size is 1 of 'C', 2 for
28FF 0601              MVI     B,1     ;  'I'
2901 FE43              CPI     'C'
```

```
2903 CA0729             JZ      V7
2906 04                 INR     B       ;obsize in B
2907 3AA828     V7      LDA     VCLASS  ;class in A
290A 2AA928             LHLD    ALEN    ;len in DE.
290D EB                 XCHG
290E 2AA628             LHLD    VPASSED ;passed in HL
2911 C31D24             JMP     NEWVAR  ;call/ret, NEWVAR allocates the
                ;                           variable
2914 CDF221     V2      CALL    ESET
2917 03                 DB      SYMERR
2918 C9                 RET
                ;
                ; tiny - c interpreter
                ;
                ;ST interprets a possibly compound statement
                ;
2919 CD872A     ST      CALL    QUIT    ;test if program should quit.
291C 3A4820             LDA     ERR
291F B7                 ORA     A
2920 C0                 RNZ
2921 CD4423             CALL    REM     ;pass over remarks and/or
                ;                       end of line
2924 CD2520             CALL    STBEGIN ;bugout for blips, statistics,
                ;                       ;  etc, user provided.
2927 2A5C20     ST2     LHLD    CURSOR  ;capture cursor
292A 225A20             SHLD    STCURS
292D CD482A             CALL    DECL    ;test for declaration
2930 C24423             JNZ     REM
2933 119620             LXI     D,LB    ;test for left bracket
2936 CD2022             CALL    LIT
2939 CA5C29             JZ      TIF
293C CD4423             CALL    REM
293F 3A4820     CMPND   LDA     ERR     ;compound statement. Execute
2942 47                 MOV     B,A     ;  each of its inner stmnts.
2943 3A4C20             LDA     LEAVE   ;  Exit on error, leave, break,
2946 B0                 ORA     B       ;  or ] literal.
2947 47                 MOV     B,A
2948 3A4D20             LDA     BRAKE
```

```
294B B0              ORA     B
294C C0              RNZ
294D 119820          LXI     D,RB    ; ]
2950 CD2022          CALL    LIT
2953 C24423          JNZ     REM     ;and done
2956 CD1929          CALL    ST      ;recursive call to ST
2959 C33F29          JMP     CMPND   ;then do next statement.
295C 116320    TIF   LXI     D,XIF   ;test for IF
295F CD2022          CALL    LIT
2962 CA9729          JZ      TWHI
2965 119A20          LXI     D,LPAR  ;skip (
2968 CD2022          CALL    LIT
296B CDB725          CALL    ASGN    ;evaluate condition
296E C8              RZ              ;return on error
296F 119C20          LXI     D,RPAR  ;skip )
2972 CD2022          CALL    LIT
2975 CD8421          CALL    TOPTOI  ;condition value
2978 7A              MOV     A,D
2979 B3              ORA     E
297A CA8A29          JZ      IF2
297D CD1929          CALL    ST      ;true, execute conditional
2980 116620          LXI     D,XELS  ;skip else clause if there
2983 CD2022          CALL    LIT
2986 C43A28          CNZ     SKIPST
2989 C9              RET
298A CD3A28    IF2   CALL    SKIPST  ;false, skip conditional
298D 116620          LXI     D,XELS  ;execute else clause if there
2990 CD2022          CALL    LIT
2993 C41929          CNZ     ST
2996 C9              RET
2997 117420    TWHI  LXI     D,XWHI  ;test for WHILE
299A CD2022          CALL    LIT
299D CAEB29          JZ      TSEM
29A0 119A20          LXI     D,LPAR  ;skip (
29A3 CD2022          CALL    LIT
29A6 CDB725          CALL    ASGN    ;condition
29A9 C8              RZ              ;return on error
29AA 119C20          LXI     D,RPAR  ;skip )
```

```
29AD CD2022             CALL    LIT
29B0 CD8421             CALL    TOPTOI  ;condition value
29B3 7A                 MOV     A,D
29B4 B3                 ORA     E
29B5 CAE729             JZ      WH2
29B8 2A5A20             LHLD    STCURS  ;true, save STCURS and CURSOR
29BB E5                 PUSH    H
29BC 2A5C20             LHLD    CURSOR
29BF E5                 PUSH    H
29C0 CD1929             CALL    ST      ;execute object of while
29C3 E1                 POP     H       ;saved cursor into OBJT
29C4 22442A             SHLD    OBJT
29C7 E1                 POP     H       ;  and stcurs into AGIN
29C8 22462A             SHLD    AGIN
29CB 3A4D20             LDA     BRAKE   ;if a BREAK statement caused
29CE B7                 ORA     A       ;  this return, then set CURSOR
29CF CAE029             JZ      WH3     ;  to object of the while and
29D2 2A442A             LHLD    OBJT    ;  skip over it, and restore
29D5 225C20             SHLD    CURSOR  ;  break. The WHILE is allll
29D8 CD3A28             CALL    SKIPST  ;  done.
29DB AF                 XRA     A
29DC 324D20             STA     BRAKE
29DF C9                 RET
29E0 2A462A     WH3     LHLD    AGIN    ;Otherwise, set cursor back to
29E3 225C20             SHLD    CURSOR  ;  beginning of while statement
29E6 C9                 RET             ;  and return, causing WHILE to
                ;                       ;  to be done again.
29E7 CD3A28     WH2     CALL    SKIPST  ;If condition is false, skip
29EA C9                 RET             ;  the object, and done.
29EB 11A520     TSEM    LXI     D,SEMI  ;test for null statement
29EE CD2022             CALL    LIT
29F1 C24423             JNZ     REM     ;and done
29F4 117A20     TRET    LXI     D,XRET  ;test for RETURN statement
29F7 CD2022             CALL    LIT
29FA CA1E2A             JZ      TBRK
29FD 11A520             LXI     D,SEMI  ;if ; or remark push a 0.
2A00 CD2022             CALL    LIT
2A03 C2152A             JNZ     TR2
```

```
2A06 11BF20           LXI     D,XNL
2A09 CD2022           CALL    LIT
2A0C C2152A           JNZ     TR2
2A0F CDB725           CALL    ASGN    ;otherwise push return value
2A12 C3182A           JMP     TR4
2A15 CDC621   TR2     CALL    PZERO
2A18 3E01     TR4     MVI     A,1     ;set leave flag
2A1A 324C20           STA     LEAVE
2A1D C9               RET
2A1E 118120   TBRK    LXI     D,XBRK  ;test for BREAK
2A21 CD2022           CALL    LIT
2A24 CA2D2A           JZ      TASG
2A27 3E01             MVI     A,1     ;set break flag
2A29 324D20           STA     BRAKE
2A2C C9               RET
2A2D CDB725   TASG    CALL    ASGN    ;if none of above, must be an
2A30 CA3F2A           JZ      STER    ;  expression, or an error.
2A33 CD8421           CALL    TOPTOI  ;if an expression, discard its
                      ;                 value.
2A36 11A520           LXI     D,SEMI  ;skip optional ;
2A39 CD2022           CALL    LIT
2A3C C34423           JMP     REM     ;and done
2A3F CDF221   STER    CALL    ESET
2A42 01               DB      STATERR ;statement error
2A43 C9               RET
2A44 0000     OBJT    DW      0       ;points to object of while
2A46 0000     AGIN    DW      0       ;points to beginning of while
                      ;
                      ;DECL tests for and interprets declarations
2A48 116F20   DECL    LXI     D,XCHAR
2A4B CD2022           CALL    LIT     ;test for CHAR
2A4E CA6C2A           JZ      TINT
2A51 3E43     CH2     MVI     A,'C'
2A53 210000           LXI     H,0
2A56 CDAB28           CALL    VALLOC
2A59 119E20           LXI     D,COMMA
2A5C CD2022           CALL    LIT
2A5F C2512A           JNZ     CH2     ;get all vars
```

```
2A62 11A520     CH3     LXI     D,SEMI  ;skip optional ;
2A65 CD2022             CALL    LIT
2A68 3E7F               MVI     A,07FH  ;set flag to Not Zero
2A6A B7                 ORA     A
2A6B C9                 RET
2A6C 116B20     TINT    LXI     D,XINT
2A6F CD2022             CALL    LIT
2A72 C8                 RZ              ;flag is zero
2A73 3E49       IN2     MVI     A,'I'
2A75 210000             LXI     H,0
2A78 CDAB28             CALL    VALLOC
2A7B 119E20             LXI     D,COMMA
2A7E CD2022             CALL    LIT
2A81 C2732A             JNZ     IN2
2A84 C3622A             JMP     CH3
                ;
                ;catches interrupts (ESC key) at appl level.
2A87 3A6220     QUIT    LDA     APPLVL
2A8A B7                 ORA     A
2A8B C8                 RZ
2A8C CD1020             CALL    CHRDY
2A8F C8                 RZ
2A90 47                 MOV     B,A     ;char keyed in -> B
2A91 3A3520             LDA     ESCAPE
2A94 B8                 CMP     B
2A95 C0                 RNZ
2A96 CD0A20             CALL    INCH    ;discard the ESC
2A99 CDF221             CALL    ESET    ;signal the escape
2A9C 63                 DB      KILL
2A9D C9                 RET
                ;
                ;evaluates arguments of a function. Sets cursor to
                ;  beginning of function's text. Parses its argument
                ;  declarations, giving them values of the parameters.
                ;  executes the function. Determines cause of exit, and
                ;  pushes default 0 return value if needed. Restores
                ;  cursor.
2A9E 00         NARGS   DB      0       ;number of args
```

```
2A9F 0000      WHERE   DW      0       ;0 for MC, otherwise address of
                       ;                function.
2AA1 0000      ARG     DW      0       ;pointer into stack to first
                       ;                arg.
2AA3 229F2A    ENTER   SHLD    WHERE
2AA6 AF                XRA     A
2AA7 329E2A            STA     NARGS
2AAA 2A4E20            LHLD    TOP
2AAD 110500            LXI     D,5
2AB0 19                DAD     D
2AB1 22A12A            SHLD    ARG
2AB4 119A20            LXI     D,LPAR  ;skip optional (
2AB7 CD2022            CALL    LIT
2ABA 119C20            LXI     D,RPAR  ;test for no args, several ways
2ABD CD2022            CALL    LIT
2AC0 C20C2B            JNZ     ARGSDNE
2AC3 2A5C20            LHLD    CURSOR
2AC6 7E                MOV     A,M
2AC7 FE5D              CPI     ']'
2AC9 CA0C2B            JZ      ARGSDNE
2ACC FE3B              CPI     ';'
2ACE CA0C2B            JZ      ARGSDNE
2AD1 FE0D              CPI     ASCRET
2AD3 CA0C2B            JZ      ARGSDNE
2AD6 FE2F              CPI     '/'
2AD8 CA0C2B            JZ      ARGSDNE
2ADB 3A4820    EN2     LDA     ERR     ;eval args, first test for err
2ADE B7                ORA     A
2ADF C0                RNZ
2AE0 2AA12A            LHLD    ARG     ;save locals
2AE3 E5                PUSH    H
2AE4 2A9F2A            LHLD    WHERE
2AE7 E5                PUSH    H
2AE8 2A9E2A            LHLD    NARGS
2AEB E5                PUSH    H
2AEC CDB725            CALL    ASGN    ;evaluate
2AEF E1                POP     H       ;restore locals
2AF0 7D                MOV     A,L
```

```
2AF1 E1              POP     H
2AF2 229F2A          SHLD    WHERE
2AF5 E1              POP     H
2AF6 22A12A          SHLD    ARG
2AF9 3C              INR     A       ;increment NARGS
2AFA 329E2A          STA     NARGS
2AFD 119E20          LXI     D,COMMA
2B00 CD2022          CALL    LIT     ;comma means more args
2B03 C2DB2A          JNZ     EN2
2B06 119C20          LXI     D,RPAR  ;optional )
2B09 CD2022          CALL    LIT
2B0C 3A4820    ARGSDNE LDA   ERR
2B0F B7              ORA     A
2B10 C0              RNZ
2B11 2A9F2A          LHLD    WHERE   ;test for MC
2B14 7C              MOV     A,H
2B15 B5              ORA     L
2B16 C2202B          JNZ     EN3
2B19 3A9E2A          LDA     NARGS
2B1C CDF72E          CALL    MC
2B1F C9              RET
2B20 2A5C20    EN3   LHLD    CURSOR  ;save current cursor
2B23 E5              PUSH    H
2B24 2A5A20          LHLD    STCURS
2B27 E5              PUSH    H
2B28 2A9F2A          LHLD    WHERE   ;set cursor to start of fctn
2B2B 225C20          SHLD    CURSOR
2B2E CDBE23          CALL    NEWFUN  ;new layer of value space
2B31 CD4423    EN4   CALL    REM     ;parse arg decls and pass value
2B34 116B20          LXI     D,XINT  ;works just like DECL, except
2B37 CD2022          CALL    LIT     ;  uses SETARG instead of
2B3A CA612B          JZ      EN5     ;  VALLOC.
2B3D 2AA12A    EN6   LHLD    ARG
2B40 0649            MVI     B,'I'
2B42 CDD22B          CALL    SETARG
2B45 2AA12A          LHLD    ARG     ;bump ARG pointer to next
2B48 110500          LXI     D,5     ;  stack layer
2B4B 19              DAD     D
```

```
2B4C 22A12A          SHLD    ARG
2B4F 119E20          LXI     D,COMMA
2B52 CD2022          CALL    LIT
2B55 C23D2B          JNZ     EN6
2B58 11A520          LXI     D,SEMI
2B5B CD2022          CALL    LIT
2B5E C3312B          JMP     EN4
2B61 116F20    EN5   LXI     D,XCHAR
2B64 CD2022          CALL    LIT
2B67 CA8E2B          JZ      EN7
2B6A 2AA12A    EN8   LHLD    ARG
2B6D 0643            MVI     B,'C'
2B6F CDD22B          CALL    SETARG
2B72 2AA12A          LHLD    ARG
2B75 110500          LXI     D,5
2B78 19              DAD     D
2B79 22A12A          SHLD    ARG
2B7C 119E20          LXI     D,COMMA
2B7F CD2022          CALL    LIT
2B82 C26A2B          JNZ     EN8
2B85 11A520          LXI     D,SEMI
2B88 CD2022          CALL    LIT
2B8B C3312B          JMP     EN4
2B8E 2A4E20    EN7   LHLD    TOP     ;test correct number of args
2B91 110500          LXI     D,5
2B94 19              DAD     D
2B95 3AA12A          LDA     ARG     ;should be TOP+5
2B98 BD              CMP     L
2B99 CAA72B          JZ      EN9
2B9C D1              POP     D       ;set up old cursor for
2B9D E1              POP     H       ;  the error call
2B9E 225C20          SHLD    CURSOR
2BA1 E5              PUSH    H
2BA2 D5              PUSH    D
2BA3 CDF221          CALL    ESET
2BA6 15              DB      ARGSERR
2BA7 219E2A    EN9   LXI     H,NARGS ;pop all args off stack
2BAA 35              DCR     M
```

```
2BAB FAB42B             JM      EN11
2BAE CDAA21             CALL    POPST
2BB1 C3A72B             JMP     EN9
2BB4 3A4820     EN11    LDA     ERR     ;if no errors, execute function
2BB7 B7                 ORA     A
2BB8 CC1929             CZ      ST
2BBB 3A4C20             LDA     LEAVE   ;push 0 if default leave
2BBE B7                 ORA     A
2BBF CCC621             CZ      PZERO
2BC2 AF                 XRA     A       ;zero LEAVE
2BC3 324C20             STA     LEAVE
2BC6 E1                 POP     H       ;restore cvrsor
2BC7 225A20             SHLD    STCURS
2BCA E1                 POP     H
2BCB 225C20             SHLD    CURSOR
2BCE CDF323             CALL    FUNDONE ;pop layer of value space
2BD1 C9                 RET
                ;
                ;HL points into stack to an arg. B (used by VALLOC) is
                ;  type. SETARG gets actual value of arg, calls VALLOC
                ;  to allocate local space, which also puts arg value
                ;  into allocated space.
2BD2 C5         SETARG  PUSH    B
2BD3 46                 MOV     B,M     ;class
2BD4 23                 INX     H
2BD5 7E                 MOV     A,M     ;lvalue
2BD6 23                 INX     H
2BD7 4E                 MOV     C,M     ;size
2BD8 23                 INX     H
2BD9 5E                 MOV     E,M     ;stuff
2BDA 23                 INX     H
2BDB 56                 MOV     D,M
2BDC FE41               CPI     'A'     ;test for actual
2BDE CAE52B             JZ      SE2
2BE1 EB                 XCHG            ;address of datum -> HL
2BE2 5E                 MOV     E,M
2BE3 23                 INX     H
2BE4 56                 MOV     D,M
```

```
2BE5 79         SE2     MOV     A,C     ;if size==1 & class==0
2BE6 3D                 DCR     A
2BE7 B0                 ORA     B
2BE8 C2EF2B             JNZ     SE3
2BEB 7B                 MOV     A,E     ;  then propogate sign
2BEC 07                 RLC
2BED 9F                 SBB     A
2BEE 57                 MOV     D,A
2BEF C1         SE3     POP     B       ;type -> A
2BF0 78                 MOV     A,B
2BF1 EB                 XCHG            ;passed value -> HL
2BF2 C3AB28             JMP     VALLOC  ;call/ret, valloc does the rest
                ;
                ;scans program and allocates all externals in next fctn
                ;  layer. An "endlibrary" line causes a new fctn layer
                ;  to be opened.
2BF5 CDBE23     LINK    CALL    NEWFUN
2BF8 3A4820     LI2     LDA     ERR     ;check no error
2BFB B7                 ORA     A
2BFC C0                 RNZ
2BFD 2A5C20             LHLD    CURSOR
2C00 23                 INX     H
2C01 23                 INX     H
2C02 EB                 XCHG
2C03 2A6020             LHLD    PROGEND
2C06 19                 DAD     D
2C07 EB                 XCHG
2C08 D8                 RC
2C09 CD4423             CALL    REM     ;more text to process, skip
2C0C 119620             LXI     D,LB    ;  remarks.
2C0F CD2022             CALL    LIT     ;test for compound statement.
2C12 CA1F2C             JZ      LIDCL
2C15 065B               MVI     B,'['   ;skip compound st.
2C17 0E5D               MVI     C,']'
2C19 CD5522             CALL    SKIP
2C1C C3F82B             JMP     LI2
2C1F CD482A     LIDCL   CALL    DECL    ;test for declaration, and
2C22 C2F82B             JNZ     LI2     ;  allocate it
```

```
2C25 118720            LXI     D,XENDL ;test for endlibrary statement.
2C28 CD2022            CALL    LIT
2C2B CA342C            JZ      LISYM
2C2E CDBE23            CALL    NEWFUN
2C31 C3F82B            JMP     LI2
2C34 CD9B22    LISYM   CALL    SYMNAME ;test for symbol
2C37 CA692C            JZ      LIERR
2C3A 3E45              MVI     A,'E'   ;allocate a variable with
2C3C 0602              MVI     B,2     ;  class E, size 2, len 1,
2C3E 1E01              MVI     E,1     ;  passed value = cursor. (This
2C40 1600              MVI     D,0     ;  is a function entry.)
2C42 2A5C20            LHLD    CURSOR
2C45 CD1D24            CALL    NEWVAR
2C48 2A5C20            LHLD    CURSOR  ;advance cursor to beginning of
2C4B 3E5B              MVI     A,'['   ;  program body.
2C4D BE        LI3     CMP     M
2C4E CA602C            JZ      LI4
2C51 23                INX     H
2C52 EB                XCHG
2C53 2A6020            LHLD    PROGEND
2C56 19                DAD     D
2C57 EB                XCHG
2C58 D24D2C            JNC     LI3
2C5B CDF221            CALL    ESET
2C5E 16                DB      LBRCERR
2C5F C9                RET
2C60 225C20    LI4     SHLD    CURSOR  ;skip body
2C63 CD3A28            CALL    SKIPST
2C66 C3F82B            JMP     LI2
2C69 CDF221    LIERR   CALL    ESET
2C6C 14                DB      LINKERR
2C6D C9                RET
               ;
               ;move -(bc) bytes from (hl) to (de)
2C6E 7E        MOVE    MOV     A,M
2C6F 12                STAX    D
2C70 13                INX     D
2C71 23                INX     H
```

```
2C72 0C                 INR     C
2C73 C26E2C             JNZ     MOVE
2C76 04                 INR     B
2C77 C26E2C             JNZ     MOVE
2C7A C9                 RET
                ;It all starts here!!!!!
                ;cold start erases system level tc programs, and enters
                ;  the loader. Used to load a tailered or different
                ;  system program.
                ;warm start does not erase sys level progs, but enters
                ;  the loader so more can be loaded.
                ;hot start assumes all the loading is done, and immed
                ;  starts up the loaded sys level tc prog.
                ;Unfortunately, there is no hot start that preserves
                ;  application programs.
2C7B 2A4620     COLD    LHLD    MSTACK  ;initialize 8080 stack, if need
2C7E 7C                 MOV     A,H
2C7F B5                 ORA     L
2C80 CA842C             JZ      CL2
2C83 F9                 SPHL
2C84 01F6FF     CL2     LXI     B,-10   ;copy initial statement
2C87 2A4220             LHLD    BPR     ;  PR
2C8A EB                 XCHG
2C8B 212B2D             LXI     H,INST  ;  into PR
2C8E CD6E2C             CALL    MOVE
2C91 2A4220             LHLD    BPR
2C94 110900             LXI     D,9
2C97 19                 DAD     D
2C98 CDF42D             CALL    HLNEG
2C9B 226020             SHLD    PROGEND
2C9E CD382E             CALL    LOGO
2CA1 CD352D     WARM    CALL    LOADER
2CA4 CD382E     HOT     CALL    LOGO
2CA7 2A6020             LHLD    PROGEND
2CAA CDF42D             CALL    HLNEG
2CAD 225E20             SHLD    PRUSED
2CB0 2A4220             LHLD    BPR
2CB3 225C20             SHLD    CURSOR
```

```
2CB6 2A3A20             LHLD    BFUN
2CB9 110600             LXI     D,6
2CBC 19                 DAD     D
2CBD 225420             SHLD    CURGLBL
2CC0 11F4FF             LXI     D,-12
2CC3 19                 DAD     D
2CC4 225220             SHLD    CURFUN
2CC7 2A3E20             LHLD    BVAR
2CCA 225020             SHLD    NXTVAR
2CCD 2A3620             LHLD    BSTACK
2CD0 11FBFF             LXI     D,-5
2CD3 19                 DAD     D
2CD4 224E20             SHLD    TOP
2CD7 AF                 XRA     A
2CD8 67                 MOV     H,A
2CD9 6F                 MOV     L,A
2CDA 324820             STA     ERR
2CDD 224A20             SHLD    ERRAT
2CE0 324C20             STA     LEAVE
2CE3 324D20             STA     BRAKE
2CE6 CDF52B             CALL    LINK
2CE9 CDBE23             CALL    NEWFUN
2CEC 2A4220             LHLD    BPR
2CEF 225C20             SHLD    CURSOR
2CF2 CD2220             CALL    PRBEGIN
2CF5 CD1929             CALL    ST      ;this executes the system progm
2CF8 CD2820             CALL    PRDONE
2CFB 21232D             LXI     H,DONEMSG
2CFE CD1622             CALL    PS
2D01 3A4820             LDA     ERR
2D04 B7                 ORA     A
2D05 CA1B2D             JZ      NOERR
2D08 2A4820             LHLD    ERR
2D0B EB                 XCHG
2D0C CDFC2D             CALL    PN
2D0F 3E20               MVI     A,' '   ;  and a space,
2D11 CD0D20             CALL    OUTCH
2D14 2A4A20             LHLD    ERRAT
```

```
2D17 EB                 XCHG
2D18 CDFC2D             CALL    PN
2D1B 3E0D       NOERR   MVI     A,0DH
2D1D CD0D20             CALL    OUTCH
2D20 C3A12C             JMP     WARM
2D23 0D0D444F4EDONEMSG  DB      0DH,0DH,'DONE ',0
2D2B 5B6D61696EINST     DB      '(main();)',0
                ;
2D35 21CC2D     LOADER  LXI     H,BUFF
2D38 3E3E               MVI     A,'>'
2D3A CD0D20             CALL    OUTCH
2D3D CD0D20             CALL    OUTCH
2D40 CD0D20             CALL    OUTCH
2D43 CD0A20     D2      CALL    INCH
2D46 47                 MOV     B,A
2D47 3A0920             LDA     ECHO
2D4A B7                 ORA     A
2D4B 78                 MOV     A,B
2D4C C40D20             CNZ     OUTCH
2D4F 77                 MOV     M,A
2D50 FE7F               CPI     7FH     ;delete char
2D52 CA5E2D             JZ      D3
2D55 FE0D               CPI     0DH     ;return
2D57 CA6B2D             JZ      DOIT
2D5A 23                 INX     H
2D5B C3432D             JMP     D2
2D5E 1133D2     D3      LXI     D,-BUFF-1
2D61 E5                 PUSH    H
2D62 19                 DAD     D
2D63 E1                 POP     H
2D64 D2432D             JNC     D2
2D67 2B                 DCX     H
2D68 C3432D             JMP     D2
2D6B 3600       DOIT    MVI     M,0     ;null at command's end
2D6D 3ACD2D             LDA     BUFF+1  ;ignore period in buff.
2D70 47                 MOV     B,A
2D71 3A9220             LDA     XR      ;the letter r
2D74 B8                 CMP     B
```

```
2D75 CA972D             JZ      LOAD
2D78 3A9420             LDA     XX      ;the letter x
2D7B B8                 CMP     B
2D7C CA0000             JZ      TCEXIT
2D7F 3A9320             LDA     XG      ;the letter g
2D82 B8                 CMP     B
2D83 C8                 RZ              ;leaves editor
2D84 3E3F               MVI     A,'?'   ;unrecognized command
2D86 CD0D20             CALL    OUTCH
2D89 CD0D20             CALL    OUTCH
2D8C CD0D20             CALL    OUTCH
2D8F 3E0D               MVI     A,0DH
2D91 CD0D20             CALL    OUTCH
2D94 C3352D             JMP     LOADER
2D97 21CF2D     LOAD    LXI     H,BUFF+3        ;file name
2D9A 110100             LXI     D,1     ;read option
2D9D 010100             LXI     B,1     ;unit
2DA0 3E01               MVI     A,1     ;open to read
2DA2 CD1320             CALL    FOPEN
2DA5 C2352D             JNZ     LOADER
2DA8 2A6020             LHLD    PROGEND ;where to load (stored neg)
2DAB CDF42D     L2      CALL    HLNEG
2DAE 010100             LXI     B,1     ;unit
2DB1 CD1620             CALL    FREAD   ;read one block
2DB4 C2C32D             JNZ     L5      ;err or end of file
2DB7 19                 DAD     D       ;??bytes read in DE
2DB8 3600               MVI     M,0     ;just beyond last byte read
2DBA CDF42D             CALL    HLNEG
2DBD 226020             SHLD    PROGEND ;points to null byte at end
2DC0 C3AB2D             JMP     L2
2DC3 010100     L5      LXI     B,1     ;close unit 1
2DC6 CD1C20             CALL    FCLOSE
2DC9 C3352D             JMP     LOADER
2DCC            BUFF    DS      40
                ;
                ;Negate HL
2DF4 7C         HLNEG   MOV     A,H
2DF5 2F                 CMA
```

```
2DF6 67                 MOV     H,A
2DF7 7D                 MOV     A,L
2DF8 2F                 CMA
2DF9 6F                 MOV     L,A
2DFA 23                 INX     H
2DFB C9                 RET
                ;
                ;print (DE) as signed integer
2DFC 21CC2D     PN      LXI     H,BUFF
2DFF CD0A2E             CALL    ITOA
2E02 3600               MVI     M,0     ;put null at end
2E04 21CC2D             LXI     H,BUFF
2E07 C31622             JMP     PS      ;and done
                ;
                ;convert (DE) to ascii signed integer
2E0A 7A         ITOA    MOV     A,D     ;test for minus
2E0B B7                 ORA     A
2E0C F2152E             JP      NTOA
2E0F CD7521             CALL    DENEG   ;make positive
2E12 362D               MVI     M,'-'   ;output minus
2E14 23                 INX     H       ;now fall into NTOA
                ;convert (DE) to ascii unsigned integer
2E15 7A         NTOA    MOV     A,D
2E16 B3                 ORA     E       ;must be at least one digit, so
2E17 C21E2E             JNZ     NT2     ;  test for 0.
2E1A 3630               MVI     M,'0'
2E1C 23                 INX     H
2E1D C9                 RET
2E1E AF         NT2     XRA     A       ;put mark on stack
2E1F F5                 PUSH    PSW
2E20 010A00     NT3     LXI     B,10
2E23 E5                 PUSH    H
2E24 CD7021             CALL    DDIV
2E27 7D                 MOV     A,L     ;remainder -> A
2E28 E1                 POP     H
2E29 C630               ADI     '0'
2E2B F5                 PUSH    PSW     ;ascii digit -> stack
2E2C 7A                 MOV     A,D     ;done if quotient is zero
```

```
2E2D B3                 ORA     E
2E2E C2202E             JNZ     NT3
2E31 F1         NT4     POP     PSW     ;top of stack is digit or mark.
2E32 C8                 RZ              ;done if mark.
2E33 77                 MOV     M,A     ;otherwise digit -> buffer.
2E34 23                 INX     H
2E35 C3312E             JMP     NT4
                ;
                ;prints the copyright message on the terminal.
2E38 213E2E     LOGO    LXI     H,CPMSG
2E3B C31622             JMP     PS
2E3E 0D2A2A2A20CPMSG    DB      0DH,'***  TINY-C   VERSION 80-01-02  ***'
2E62 0D434F5059         DB      0DH,'COPYRIGHT 1979, T. A. GIBSON',0DH,0
                ;
                ;move the block (DE)...(HL) inclusive (BC) bytes. If
                ;  (BC) is positive, the block is moved up in RAM,
                ;  highest byte first, lowest byte last. If (BC) is
                ;  negative, the block is moved down in RAM, lowest
                ;  byte first. Thus large blocks can be safely moved
                ;  up or down short distances.
2E81 78         MOVEBL  MOV     A,B
2E82 B7                 ORA     A
2E83 FAA92E             JM      MOVEDN
2E86 B1                 ORA     C
2E87 C8                 RZ
2E88 22C02E     MOVEUP  SHLD    FROMPTR ;hi end of block is fromptr
2E8B 09                 DAD     B       ;to pointer -> DE
2E8C EB                 XCHG
2E8D 3AC02E             LDA     FROMPTR ; - length -> BC
2E90 2F                 CMA
2E91 85                 ADD     L       ; - length =
2E92 4F                 MOV     C,A     ;       current HL - fromptr +1
2E93 3AC12E             LDA     FROMPTR+1
2E96 2F                 CMA
2E97 8C                 ADC     H
2E98 47                 MOV     B,A
2E99 2AC02E             LHLD    FROMPTR
2E9C 7E         MU2     MOV     A,M
```

```
2E9D 12              STAX    D
2E9E 2B              DCX     H
2E9F 1B              DCX     D
2EA0 0C              INR     C
2EA1 C29C2E          JNZ     MU2
2EA4 04              INR     B
2EA5 C29C2E          JNZ     MU2
2EA8 C9              RET
2EA9 EB        MOVEDN XCHG             ;lo end of block is from ptr
2EAA 22C02E          SHLD    FROMPTR
2EAD 09              DAD     B          ;to pointer -> HL
2EAE 3AC02E          LDA     FROMPTR ; - length -> BC
2EB1 93              SUB     E
2EB2 4F              MOV     C,A
2EB3 3AC12E          LDA     FROMPTR+1
2EB6 9A              SBB     D
2EB7 47              MOV     B,A
2EB8 0B              DCX     B
2EB9 EB              XCHG            ;to ptr -> DE
2EBA 2AC02E          LHLD    FROMPTR ;from ptr -> HL
2EBD C36E2C          JMP     MOVE
2EC0 0000      FROMPTR DW    0
               ;
               ;scan for the Nth occurance of a character in a block,
               ;  or the end of the block, whichever comes first. The
               ;  block is (DE)..(HL) inclusive. N is (BC) and can be
               ;  0 to 65k. (A) is the character. On completion, (DE)
               ;  points to the Nth occurance, or to the last byte of
               ;  the block. (BC) is N minus the number of (A) found,
               ;  e.g. 0 if N (A)'s were found. HL is undisturbed.
2EC2 F5        SCANN   PUSH    PSW     ;ch -> stack
2EC3 EB                XCHG            ;reverse first and last
2EC4 79        SC2     MOV     A,C
2EC5 B0                ORA     B       ;test if done
2EC6 CADB2E            JZ      SC9
2EC9 7B                MOV     A,E
2ECA 95                SUB     L
2ECB 7A                MOV     A,D
```

```
2ECC 9C                 SBB     H
2ECD DADB2E             JC      SC9
2ED0 F1                 POP     PSW
2ED1 F5                 PUSH    PSW
2ED2 BE                 CMP     M
2ED3 C2D72E             JNZ     SC3
2ED6 0B                 DCX     B
2ED7 23         SC3     INX     H
2ED8 C3C42E             JMP     SC2
2EDB 2B         SC9     DCX     H
2EDC EB                 XCHG
2EDD F1                 POP     PSW
2EDE C9                 RET
                ;
                ;count the occurances of a character in a block. (A) is
                ;  the character. The block is (DE)..(HL) inclusive.
                ;  The count is returned in (BC). (A) and (DE) are
                ;  unchanged. (HL) is clobbered.
2EDF 010000     COUNTCH LXI     B,0
2EE2 F5                 PUSH    PSW     ;ch -> stack
2EE3 7D         CC2     MOV     A,L     ;test for end
2EE4 93                 SUB     E
2EE5 7C                 MOV     A,H
2EE6 9A                 SBB     D
2EE7 DAF52E             JC      CC9
2EEA F1                 POP     PSW
2EEB F5                 PUSH    PSW
2EEC BE                 CMP     M
2EED 2B                 DCX     H
2EEE C2E32E             JNZ     CC2
2EF1 03                 INX     B       ;count this one
2EF2 C3E32E             JMP     CC2
2EF5 F1         CC9     POP     PSW
2EF6 C9                 RET
                ;Machine Call routine to interface to 8080 coded
                ;  routines.  Standard routines used by the system
                ;  are coded here, numbers 1 to 11. 12 to 999 are
                ;  reserved. 1000 and up are available to users.
```

```
2EF7 323420    MC      STA     MCARGS  ;for checking,
2EFA CD8421            CALL    TOPTOI  ;  for MC's that need it.
2EFD 2118FC            LXI     H,-1000 ;test for user MC
2F00 19                DAD     D
2F01 DA1F20            JC      USERMC
2F04 7B                MOV     A,E     ;fctn num -> A
2F05 FE01              CPI     1
2F07 CA502F            JZ      MC1
2F0A FE02              CPI     2
2F0C CA5A2F            JZ      MC2
2F0F FE03              CPI     3
2F11 CA7E2F            JZ      MC3
2F14 FE04              CPI     4
2F16 CA9C2F            JZ      MC4
2F19 FE05              CPI     5
2F1B CAB22F            JZ      MC5
2F1E FE06              CPI     6
2F20 CACA2F            JZ      MC6
2F23 FE07              CPI     7
2F25 CAD52F            JZ      MC7
2F28 FE08              CPI     8
2F2A CAE82F            JZ      MC8
2F2D FE09              CPI     9
2F2F CAFE2F            JZ      MC9
2F32 FE0A              CPI     10
2F34 CA2830            JZ      MC10
2F37 FE0B              CPI     11
2F39 CA2A30            JZ      MC11
2F3C FE0C              CPI     12
2F3E CABB30            JZ      MC12
2F41 FE0D              CPI     13
2F43 CAC430            JZ      MC13
2F46 FE0E              CPI     14
2F48 CAE230            JZ      MC14
2F4B CDF221    MCESET  CALL    ESET
2F4E 18                DB      MCERR
2F4F C9                RET
               ;
```

```
                ;put a character to screen
2F50 CD8421     MC1     CALL    TOPTOI  ;char -> A
2F53 CDC921             CALL    PUSHK   ;push it back
2F56 7B                 MOV     A,E
2F57 C30D20             JMP     OUTCH
                ;
                ;get a char from keyboard
2F5A CD0A20     MC2     CALL    INCH    ;char -> DE
2F5D 47                 MOV     B,A     ;test for ESC in appl level
2F5E 3A6220             LDA     APPLVL
2F61 B7                 ORA     A
2F62 CA702F             JZ      USEIT
2F65 3A3520             LDA     ESCAPE
2F68 B8                 CMP     B
2F69 C2702F             JNZ     USEIT
2F6C CDF221             CALL    ESET
2F6F 63                 DB      KILL
2F70 3A0920     USEIT   LDA     ECHO    ;test if echo required
2F73 B7                 ORA     A
2F74 78                 MOV     A,B
2F75 C40D20             CNZ     OUTCH
2F78 5F                 MOV     E,A
2F79 AF                 XRA     A
2F7A 57                 MOV     D,A
2F7B C3C921             JMP     PUSHK   ;put char onto stack
                ;
                ;file open (r/w, name, fsize, unit)
2F7E CD8421     MC3     CALL    TOPTOI
2F81 D5                 PUSH    D
2F82 CD8421             CALL    TOPTOI
2F85 D5                 PUSH    D
2F86 CD8421             CALL    TOPTOI
2F89 D5                 PUSH    D
2F8A CD8421             CALL    TOPTOI  ;r/w -> A
2F8D 7B                 MOV     A,E
2F8E B2                 ORA     D
2F8F E1                 POP     H       ;name pointer -> HL
2F90 D1                 POP     D       ;file size -> DE
```

```
2F91 C1              POP     B       ;unit -> BC
2F92 CD1320          CALL    FOPEN
2F95 110000          LXI     D,0
2F98 5F              MOV     E,A     ;push result code
2F99 C3C921          JMP     PUSHK
                     ;
                     ; read block( where, unit)
2F9C CD8421    MC4   CALL    TOPTOI
2F9F D5              PUSH    D
2FA0 CD8421          CALL    TOPTOI
2FA3 EB              XCHG            ;where -> HL
2FA4 C1              POP     B       ;unit -> BC
2FA5 CD1620          CALL    FREAD
2FA8 CAAF2F          JZ      MC4P    ;if result code is 0 DE has
2FAB 11FFFF          LXI     D,-1    ;  byte count to be pushed.
2FAE 5F              MOV     E,A     ;  Otherwise A is an err or eof
2FAF C3C921    MC4P  JMP     PUSHK   ;  code to be returned negative
                     ;
                     ;write block ( first byte, last byte, unit). Block may
                     ;  be any size from 1 to 256.
2FB2 CD8421    MC5   CALL    TOPTOI
2FB5 D5              PUSH    D
2FB6 CD8421          CALL    TOPTOI
2FB9 D5              PUSH    D
2FBA CD8421          CALL    TOPTOI
2FBD EB              XCHG            ;first -> HL
2FBE D1              POP     D       ;last -> DE
2FBF C1              POP     B       ;unit -> BC
2FC0 CD1920          CALL    FWRITE
2FC3 110000          LXI     D,0     ;push result code
2FC6 5F              MOV     E,A
2FC7 C3C921          JMP     PUSHK
                     ;
                     ;close file ( unit )
2FCA CD8421    MC6   CALL    TOPTOI
2FCD 4B              MOV     C,E     ;unit -> BC
2FCE 42              MOV     B,D
2FCF CD1C20          CALL    FCLOSE
```

```
2FD2 C3C621             JMP     PZERO   ;return a 0
                ;
                ;move a block up or down. Args are first,last,K. If K
                ;  negative, block is moved down |k| bytes, if positive
                ;  then up K bytes.
2FD5 CD8421     MC7     CALL    TOPTOI
2FD8 D5                 PUSH    D
2FD9 CD8421             CALL    TOPTOI
2FDC D5                 PUSH    D
2FDD CD8421             CALL    TOPTOI  ;first -> DE
2FE0 E1                 POP     H       ;last
2FE1 C1                 POP     B       ;K
2FE2 CD812E             CALL    MOVEBL
2FE5 C3C621             JMP     PZERO   ;return a 0
                ;
                ;count ??instances of character CH in a block. Args are
                ; first,last,CH.
2FE8 CD8421     MC8     CALL    TOPTOI
2FEB D5                 PUSH    D
2FEC CD8421             CALL    TOPTOI
2FEF D5                 PUSH    D
2FF0 CD8421             CALL    TOPTOI  ;first -> DE
2FF3 E1                 POP     H       ;last
2FF4 C1                 POP     B       ;ch -> A
2FF5 79                 MOV     A,C
2FF6 CDDF2E             CALL    COUNTCH
2FF9 59                 MOV     E,C     ;count -> DE
2FFA 50                 MOV     D,B
2FFB C3C921             JMP     PUSHK
                ;
                ;scan for nth occurance of CH in a block. Args are
                ;  first,last,CH,cnt address. Return pointer to nth
                ;  occurance,if it exists, otherwise to last. Also
                ;  cnt is reduced by one for every CH found.
2FFE CD8421     MC9     CALL    TOPTOI
3001 D5                 PUSH    D
3002 CD8421             CALL    TOPTOI
3005 D5                 PUSH    D
```

```
3006 CD8421            CALL    TOPTOI
3009 D5                PUSH    D
300A CD8421            CALL    TOPTOI  ;first -> DE
300D E1                POP     H       ;last
300E C1                POP     B       ;ch -> A
300F 79                MOV     A,C
3010 E3                XTHL
3011 4E                MOV     C,M     ;cnt -> BC
3012 23                INX     H
3013 46                MOV     B,M
3014 2B                DCX     H
3015 E3                XTHL            ;addr of cnt still on stack
3016 D5                PUSH    D       ;first on stack, too
3017 CDC22E            CALL    SCANN
301A E1                POP     H       ;make ptr (DE) relative to
301B 7B                MOV     A,E     ;  first
301C 95                SUB     L
301D 5F                MOV     E,A
301E 7A                MOV     A,D
301F 9C                SBB     H
3020 57                MOV     D,A
3021 E1                POP     H       ;BC -> cnt
3022 71                MOV     M,C
3023 23                INX     H
3024 70                MOV     M,B
3025 C3C921            JMP     PUSHK   ;return pointer to last byte
                ;                      ;  examined.
                ;
                ;trap to moniter 4.0 for debugging.
3028 FF         MC10    DB      0FFH    ;RST 7
3029 C9                 RET
                ;
                ;enters an application program, setting up a new
                ; globals variable level, redefining progend, links
                ; the program, executes if no error occured, upon
                ; completion captures a few facts (err, and either
                ; cursor or errat) and restores old globals level,
                ; progend, zeros err, pushes a zero as the value of
```

```
                ; this function, and resumes the calling program.
302A 2A5C20     MC11    LHLD    CURSOR
302D E5                 PUSH    H
302E 2A6020             LHLD    PROGEND
3031 E5                 PUSH    H
3032 2A5E20             LHLD    PRUSED
3035 E5                 PUSH    H
3036 2A5420             LHLD    CURGLBL
3039 E5                 PUSH    H
303A CD8421             CALL    TOPTOI  ;appl pr address
303D EB                 XCHG
303E E5                 PUSH    H
303F 225C20             SHLD    CURSOR
3042 CD8421             CALL    TOPTOI  ;end of appl addr
3045 EB                 XCHG
3046 225E20             SHLD    PRUSED
3049 CDF42D             CALL    HLNEG
304C 226020             SHLD    PROGEND
304F CDF52B             CALL    LINK
3052 2A5220             LHLD    CURFUN
3055 225420             SHLD    CURGLBL
3058 CD8421             CALL    TOPTOI  ;start statement address
305B EB                 XCHG
305C 225C20             SHLD    CURSOR
305F CDBE23             CALL    NEWFUN
3062 CD8421             CALL    TOPTOI  ;facts address
3065 D5                 PUSH    D
3066 216220             LXI     H,APPLVL        ;increment appl level
3069 34                 INR     M
306A E5                 PUSH    H
306B 3A4820             LDA     ERR     ;if no err so far, do it!!
306E B7                 ORA     A
306F C27B30             JNZ     DONE
3072 CD2220             CALL    PRBEGIN
3075 CD1929             CALL    ST
3078 CD2820             CALL    PRDONE
307B E1         DONE    POP     H       ;its done, decrement appl level
307C 35                 DCR     M
```

```
307D CDF323            CALL    FUNDONE ;discard appl locals
3080 CDF323            CALL    FUNDONE ;  and globals
3083 2A5C20            LHLD    CURSOR  ;set up facts
3086 3A4820            LDA     ERR
3089 B7                ORA     A
308A CA9030            JZ      MCEN2
308D 2A4A20            LHLD    ERRAT
3090 EB        MCEN2   XCHG            ;returned currsor -> DE
3091 E1                POP     H       ;facts -> HL
3092 C1                POP     B       ;appl pr address -> BC
3093 7B                MOV     A,E     ;make returned cursor relative
3094 91                SUB     C       ;  to appl address
3095 5F                MOV     E,A
3096 7A                MOV     A,D
3097 98                SBB     B
3098 57                MOV     D,A
3099 3A4820            LDA     ERR
309C 77                MOV     M,A     ;err -> facts
309D AF                XRA     A
309E 23                INX     H
309F 77                MOV     M,A     ;err hi byte -> facts
30A0 23                INX     H
30A1 73                MOV     M,E     ;cursor -> facts
30A2 23                INX     H
30A3 72                MOV     M,D
30A4 E1                POP     H       ;curglobal
30A5 225420            SHLD    CURGLBL
30A8 E1                POP     H
30A9 225E20            SHLD    PRUSED
30AC E1                POP     H       ;progend
30AD 226020            SHLD    PROGEND
30B0 E1                POP     H       ;cursor
30B1 225C20            SHLD    CURSOR
30B4 AF                XRA     A       ;zero the error
30B5 324820            STA     ERR
30B8 C3C621            JMP     PZERO   ;value of MC11
                ;
                ;test if keyboard char ready, return copy if so,else 0.
```

```
30BB CD1020    MC12    CALL    CHRDY
30BE 1600              MVI     D,0
30C0 5F                MOV     E,A
30C1 C3C921            JMP     PUSHK
                ;
                ;print RAM, from and to addresses are given
                ;  nulls are mapped to quotes
30C4 CD8421    MC13    CALL    TOPTOI
30C7 D5                PUSH    D
30C8 CD8421            CALL    TOPTOI
30CB EB                XCHG            ;from -> HL
30CC D1                POP     D       ;to -> DE
30CD 7B        LOOP13  MOV     A,E     ;test if done
30CE 95                SUB     L
30CF 7A                MOV     A,D
30D0 9C                SBB     H
30D1 DAC621            JC      PZERO   ;done
30D4 7E                MOV     A,M
30D5 B7                ORA     A
30D6 C2DB30            JNZ     EC13
30D9 3E22              MVI     A,'"'
30DB CD0D20    EC13    CALL    OUTCH
30DE 23                INX     H
30DF C3CD30            JMP     LOOP13
                ;
                ;print a signed integer
30E2 CD8421    MC14    CALL    TOPTOI
30E5 D5                PUSH    D
30E6 CDFC2D            CALL    PN
30E9 D1                POP     D
30EA C3C921            JMP     PUSHK
                ;
                ;end of the standard interpreter
30ED =         TCEND   EQU     $
3100 =         BFREE   EQU     (TCEND+0FFH)/100H*100H  ;next free page
2C00 =         SPACE   EQU     EFREE-BFREE
```

RELOCATION

```
;To relocate tiny-c Version 80-01-02, use the
;  program in Figure 6-1 of Chapter VI with the
;  following TCADDS table substituted for lines
;  0090 through 0430.
;
TCADDS  DW      2000H
        DW      2008H
        DW      202BH
        DW      2033H
        DW      20C1H
        DW      23D3H
        DW      23D5H
        DW      2412H
        DW      241DH
        DW      24DFH
        DW      24EFH
        DW      2576H
        DW      2579H
        DW      27FFH
        DW      2801H
        DW      2835H
        DW      283AH
        DW      28A4H
        DW      28ABH
        DW      2A41H
        DW      2A48H
        DW      2A9DH
        DW      2AA3H
        DW      2C5DH
        DW      2C5FH
        DW      2D22H
        DW      2D35H
        DW      2DCBH
        DW      2DF4H
        DW      2E3DH
        DW      2E81H
        DW      2EBFH
        DW      2EC2H
        DW      30ECH
        DW      0
```