Apple /// Computer Technical Information

Apple /// Business BASIC 1.3 Source Code Listing

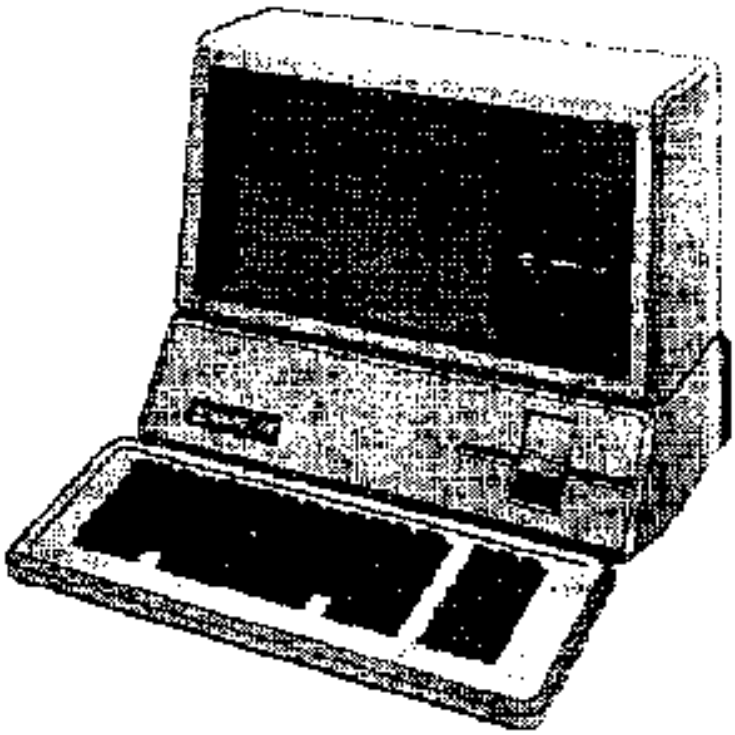

Table of Contents

General Information
Source File Catalog
Build Information
Source Code Listing
DTCAsmReFormat Program Listing

GENERAL INFORMATION

Apple /// Business BASIC was the BASIC interpreter that Apple computer created for its Apple /// Computer which was released in 1980.

This BASIC appears to be based heavily upon Apple's AppleSoft BASIC which was licenced from Microsoft around 1976. Apple appears to have taken the AppleSoft BASIC source and modified it to support new features for Business BASIC. From internal Apple documents relating to the Apple ///'s development, it appears that Apple created in the late 1970's a BASIC called BASIC III which pre-dated the Apple ///'s development. Apple appears to have converted this BASIC III to become Apple /// Business BASIC.

This BASIC is written in 6502 assembly language. It was assembled on the Apple][(or //e) computers using Apple's EDASM 6502 editor and assembler program. It is interesting (to me at least) that the source code for SOS (the Apple ///'s operating system) was also not ported to the Apple ///. It seems that Apple had no need to put the sources for either Business BASIC or SOS on the /// since the][host worked and changing the source from EDASM format to Pascal /// Assembler format would have taken some time.

This BASIC's chief programmer was Donn Denman who also later wrote Apple's Macintosh BASIC (which was never released).

For information about Business BASIC's features see Apple's "Business BASIC Reference Manual" (two volumes). Also available from Apple is a document describing Business BASIC's variable storage and how to write Business BASIC's invokable modules. I think it would be interesting to obtain the Business BASIC ERS (External Reference Specification) which Apple prepared before each hardware and software project and which served as the blueprint for a project.

From a programming perspective Business BASIC's source code is not well commented. Very few routines have a header comment describing what the routine does and what parameters it uses. The source also does not have a modification history unlike other Apple /// software which Apple produced. Also lacking in the source is a general discussion of how Business BASIC works. This discussion would include the details about the tokenization of keywords how this BASIC generally worked (maybe the BB ERS has this information). This source appears in general to have the same minimal commenting style as AppleSoft BASIC.

Business BASIC's source is not without humor. File CATALOG.TEXT contains the following line:

```
      CMP #$0F      ;ROOT DIRECTORY? (WHAT DOES TOM HAVE TO DO WITH IT?)
```

This line refers to Tom Root who was one of the designers behind the Apple ///'s operating system and its file system. FWIW, I searched in the file for other people's names who worked on the /// software but none were found.

File INVOKE1.TEXT also contains a bit of humor:

```
      ;I WISH I WERE A MOTOROLA 68000,
      ;YES THAT IS WHAT I'D TRUELY LIKE TO BE,
      ;CUZ IF I WERE A M. 68000.
      ;EVERYONE WOULD LOVE TO PROGRAM ME!
```

The author of this document (David Craig) has modified the Business BASIC source files to make them more readable. The original files had no line formatting and were very difficult to read. I have made the following changes:

o Renamed all source files to end with a ".TEXT" suffix.

o Reformatted all the files so they look much more readable. I used a Macintosh program that I created (DTCAsmReFormat) for this purpose which lined up all the assembly language elements in a nice fashion.

o Added a header and a footer to each file listing the file's name.

o Merged all the source files into a single file in the order that the files are assembled. This was done so that I would have just a single file instead of the many original files.

I also used another Macintosh utility (DTCStripTabs) which added the line numbers to each source file and put its own header and footer information.

SOURCE FILE CATALOG

This catalog is arranged the order in which the files were assembled. The line and character counts are for the files after I reformatted them.

File Name	Lines	Chars
BASIC.TEXT	5	93
BASIC.RT.TEXT	5	93
BASIC.D.TEXT	5	93
BASIC.INCL.TEXT	58	2247
B3INVOK.TEXT	170	7866
ZPG.EQUS.TEXT	555	27069
B3RESVB.TEXT	394	14978
INITIAL.TEXT	570	26080
B3MAINC.TEXT	564	27430
EXTRAS.TEXT	312	13576
SOSSTUF.TEXT	351	15697
B3LISTD.TEXT	563	26574
B3GOTOE.TEXT	543	26737
B3INPUF.TEXT	599	28226
B3EVALG.TEXT	399	18518
STRUTILS.TEXT	461	19661
B3MATHK.TEXT	279	12528
B3MATHL.TEXT	517	22356
B3FINPM.TEXT	448	21543
B3EXPON.TEXT	431	18931
B3FREER.TEXT	231	11268
LONGINT.TEXT	647	28719
B3DMPYT.TEXT	228	10986
B3DIMNH.TEXT	517	25863
B3UDEFI.TEXT	378	17495
STRNGSTUF.TEXT	654	29891
INVOKE.TEXT	667	29985
INVOKE1.TEXT	346	15660
B3PRU1.TEXT	723	36115
B3PRU2.TEXT	578	26292
DISKSTUF1.TEXT	366	16105
DISKSTUF2.TEXT	303	13293
DISCMDS.TEXT	598	26914
FILESTUF.TEXT	486	22774
CATALOG.TEXT	495	20529
BASICEND.TEXT	77	3720
Total	14523	665905

```
000072                  STY       FRETOP
000073                  STA       FRETOP+1                ;TOP OF STRINGS TOO.
000074                  STX       FRETOPB
000075                  LDX       RAMLOC
000076                  LDY       RAMLOC+1                ;LOAD THE ADDRESS
000077                  LDA       RAMLOCB                 ;  (AND BANK)
000078                  STX       TXTTAB                  ;  OF LOW END OF USER RAM SEGMENT
000079                  STY       TXTTAB+1                ;  & SAVE IT IN TXTTAB
000080                  STA       TXTTABB
000081                  LDA       $FFD9                   ;RANDOM COUNTER HERE.
000082                  AND       #$7F
000083                  STA       RNDX+4
000084                  LDA       $FFE9                   ;ANOTHER COUNTER.
000085                  STA       RNDX+5                  ;SEED RANDOM # GENERATOR.
000086                  JSR       P1INIT                  ;FIRST PART OF INIT
000087                  JSR       INITCNS                 ;INIT THE CONSOLE.
000088                  JSR       SCRTCH                  ;SET UP EVERYTHING ELSE.
000089                  JSR       FILSOS                  ;Put SOS prefix in SOSPATH
000090                  LDA       #>SWCHGO
000091                  STA       DISPATCH+1
000092                  LDA       #<SWCHGO
000093                  STA       DISPATCH+2
000094                  LDA       SELECTOR
000095                  BEQ       BOOTRUN
000096 *
000097 *  This entry routine if Program Selector
000098 *
000099                  LDA       PROGPATH                ;Get the length of the program path
000100                  beq       nopgm
000101                  LDA       #>PROGPATH              ;Set up pathname pointer to point
000102                  STA       HELF+1                  ;  to length byte of PROGPATH
000103                  LDA       #<PROGPATH
000104                  STA       HELF+2
000105                  BRK                               ;Here for the SELECTOR
000106                  DFB       SGFI
000107                  DW        HELF
000108                  BEQ       SELLO                   ;Go & RUN the Designated Program
000109 nopgm1           EQU       *
000110                  LDX       #$FF
000111                  STX       CURLIN+1
000112                  JMP       SERROR                  ;If file not found, err msg and return
000113 SELLO            STA       LINNUM                  ;Now RUN Selected Program
000114                  STA       LINNUM+1                ;  starting at line 0
000115                  STA       TXTPTRB
000116                  DEC       CURLIN+1
000117                  LDA       #>PROGPATH+1            ;POINT AT NAME
000118                  STA       TXTPTR
000119                  LDA       #<PROGPATH+1
000120                  STA       TXTPTR+1
000121                  JSR       CHRGOT                  ;Set STATUS on first character
000122                  JMP       RUN
000123                  DO        RUNTIME
000124 nopgm            LDA       #SEFNF
000125                  BNE       nopgm1
000126                  ELSE
000127 nopgm            EQU       *
000128                  FIN
000129 *
000130 *  This entry routine if NO Program Selector or if NO program
000131 *    is specified in the P/S Development environment.
000132 *
000133 BOOTRUN          EQU       *
000134                  JSR       COPYSOS                 ;Put SOS prefix into PROGPATH
000135                  BRK
000136                  DFB       SGFI
000137                  DW        HELF
000138                  BEQ       *+9                     ;Go & RUN the "HELLO" Program
000139                  DO        RUNTIME
000140                  LDX       #$FF
000141                  STX       CURLIN+1
000142                  JMP       SERROR                  ;If RUNTIME, & no HELLO, back to COLD START
000143                  ELSE
000144                  LDX       CURLIN+1
000145                  STX       CURLIN+1
000146                  JMP       MAIN                    ;IGNORE THE FILE NOT FOUND ERROR, GO INTO BASIC
000147                  FIN
000148 *   NOW RUN "HELLO"
000149                  STA       LINNUM                  ;AT LINE 0
000150                  STA       LINNUM+1
000151                  STA       TXTPTRB
```

```
000152                  DEC       CURLIN+1
000153                  LDA       #>HELCN                   ;POINT AT NAME
000154                  STA       TXTPTR
000155                  LDA       #<HELCN
000156                  STA       TXTPTR+1
000157                  JSR       CHRGOT                    ;SET STATUS ON 'H'
000158                  JMP       RUN
000159 *
000160 *   From here on, the code MUST be preserved!
000161 *
000162 *=========================================================================
000163 * HERE IS ROUTINE TO INITIALIZE THE CONSOLE, STUFF
000164 *
000165 INITCNS:         BRK
000166                  DFB       SOPN
000167                  DW        OPNCNST
000168                  BNE       FUK2                      ;ERROR...
000169                  LDA       CONSRFN                   ;REF NUM FOR CONSOLE
000170                  STA       SCHRTB+1
000171                  STA       SLINTB+1
000172                  STA       SINIT+1
000173                  STA       ISNLTB+1
000174                  STA       RDCTC+1                   ;READ FOR CNTROL-C.
000175                  BRK                                 ;GET DEFAULTS ON .CONS
000176                  DFB       SDGDN                     ;SOS CALL - GET DEV NUM
000177                  DW        DGCN
000178                  BNE       FUK2
000179                  LDA       OTDN
000180                  STA       INDN
000181                  STA       INDN2
000182                  STA       INDN3
000183                  STA       INDN4
000184                  STA       INDN5
000185                  STA       INDN6
000186                  BRK                                 ;SOS CALL - D-CONTROL
000187                  DFB       SDCNT                     ;RESET DEVICE
000188                  DW        DINIT
000189                  BNE       FUK2
000190                  BRK
000191                  DFB       SDCNT                     ;SET CNTRL-C SNIFFING ON.
000192                  DW        DCNTR
000193                  BNE       FUK2
000194                  BRK                                 ;SET NEWLINE=TRUE, ON CR
000195                  DFB       SNWL
000196                  DW        ISNLTB
000197                  BNE       FUK2
000198                  BRK                                 ;NOW INITIALIZE OUTPUT
000199                  DFB       SWRT
000200                  DW        SINIT
000201                  BNE       FUK2
000202                  RTS
000203 FUK2             JMP       SERROR
000204 * CHARS TO INIT THE VIDEO--
000205 SICHRS:          DFB       $10,2,$15,$D
000206                  DFB       1                         ;RESET VIEWPORT.
000207                  DFB       $1C                       ;CLEAR SCREEN
000208                  DFB       6                         ;CURSOR OFF
000209                  DFB       7
000210                  DFB       $0D                       ;CARRIAGE RETURN
000211                  DO        RUNTIME
000212                  ASC       '    ' ;4 spaces'
000213                  ELSE
000214                  ASC       '       ' ;7 spaces '
000215                  FIN
000216                  ASC       'Apple Business BASIC ' '
000217                  DO        RUNTIME
000218                  ASC       'Run-Time ' '
000219                  FIN
000220                  ASC       'v1.3' '
000221                  DO        DEBUG
000222                  ASC       'D' ;Put in the 'D' if Debugger Version '
000223                  FIN
000224                  ASC       ' - Copyright Apple Computer, 1980-83'
000225                  DO        RUNTIME
000226                  ELSE
000227                  ASC       '       ' ;7 spaces '
000228                  FIN
000229                  DFB       0,0,0
000230                  DFB       $0D,$0A,$0A               ;CR,LF.
000231 SICLEN           EQU       *-SICHRS
```

```
000232                  DFB     $DA,$19,$83
000233 P1INIT:          LDX     #255                    ;MAKE IT LOOK DIRECT IN CASE OF
000234                  STX     CURLIN+1                ;ERROR MESSAGE.
000235                  STX     FILNO
000236                  STX     FILNO+1
000237                  LDA     #$80
000238                  STA     OLDTXTB
000239                  LDA     #$4C                    ;A JMP opcode to set up the jumps to
000240                  STA     DISPATCH                ; DISPATCH and JMPER.  (In both cases,
000241                  STA     JMPER                   ;   bytes +1 & +2 are filled w/address)
000242                  LDX     #0
000243                  STX     LASTPT+1
000244                  STX     ERRFLG                  ;NO ERROR OR ON KBD
000245                  LDA     #STRSIZ
000246                  STA     FOUR6
000247                  LDX     #TEMPST
000248                  STX     TEMPPT                  ;SET UP STRING TEMPORARIES.
000249                  LDA     #80
000250                  STA     OUTREC                  ;DEFAULT RIGHT HAND MARGIN FOR LIST.
000251                  LDA     #2
000252                  STA     INDENT                  ;DEFAULT # OF SPACES TO INDENT FOR-NEXT LISTINGS.
000253                  LDA     #0                      ;Expand to all possible memory
000254                  JSR     EXPAND
000255                  LDY     #0
000256                  STY     RNFLG                   ;RUN FLAG (0 until a pgm runs)
000257                  TYA
000258                  STY     ERRFLG                  ; MISC INITIALIZATION
000259                  DEY                             ;STORE $FF
000260                  STY     FILNO                   ;  IN OUTPUT FILENUM
000261                  STY     FILNO+1                 ;  & OUTPUT#
000262                  LDY     #FCBLEN*10              ;CLEAR OUT FILE FCBS
000263                  STA     FCB-1,Y
000264                  DEY
000265                  BNE     *-4
000266                  STA     (TXTTAB),Y              ;SET UP TEXT TABLE.
000267                  INY
000268                  STA     (TXTTAB),Y
000269                  INY                             ;TXTTAB ALWAYS STARTS ON PAGE BOUNDARY.
000270                  STA     (TXTTAB),Y
000271                  LDA     TXTTAB
000272                  CLC
000273                  ADC     #3
000274                  STA     ARYTAB
000275                  LDA     TXTTAB+1
000276                  ADC     #0
000277                  LDY     TXTTABB
000278                  JSR     FIXADC
000279                  STA     ARYTAB+1
000280                  TYA
000281                  ADC     #0
000282                  STA     ARYTABB
000283                  RTS
000284                  SBTL    "GENERAL STORAGE MANAGEMENT ROUTINES."
000285 ; Find a FOR entry on the Stack via VARPNT.
000286 FORSIZ           EQU     $14
000287 FNDFOR:          TSX                             ;Load X register with Stack Pointer
000288                  INX
000289                  INX
000290                  INX
000291                  INX                             ;IGNORE ADR(NEWSTT) AND RTS ADDR.
000292 FFLOOP:          LDA     257,X                   ;GET STACK ENTRY.
000293                  CMP     #FORTK                  ;Is it a FOR token?
000294                  BNE     FFRTS                   ;No, no FOR loops with this Pntr.
000295                  LDA     FORPNT+1                ;GET HIGH.
000296                  ORA     FORPNT                  ;IS IT ZERO?
000297                  BNE     CMPFOR
000298                  LDA     259,X                   ;PNTR IS ZERO, SO ASSUME THIS ONE.
000299                  STA     FORPNT
000300                  LDA     260,X
000301                  STA     FORPNT+1
000302                  LDA     258,X                   ;FAKE ARRAY AND INT FLAGS TOO.
000303                  STA     TEMPFOR
000304                  STA     INTFLG
000305                  ROR     A
000306                  ROR     ISARA
000307 CMPFOR:          LDA     FORPNT+1
000308                  CMP     260,X
000309                  BNE     ADDFRS                  ;NOT THIS ONE.
000310                  LDA     FORPNT                  ;GET D_WN.
000311                  CMP     259,X
```

```
000312                BNE       ADDFRS
000313                LDA       TEMPFOR
000314                CMP       258,X
000315                BEQ       FFRTS                   ;WE GOT IT! WE GOT IT!
000316 ADDFRS:        TXA
000317                CLC                               ;ADD 16 TO X.
000318                ADC       #FORSIZ
000319                TAX                               ;RESULT BACK INTO X.
000320                BNE       FFLOOP
000321 FFRTS:         RTS                               ;RETURN TO CALLER.
000322 ;
000323 * Here is the Block Transfer Up routine. (HIGHDS)<(LOWTR).(HIGHTR)M
000324 * ON ENTRY:
000325 *  HIGHDS is the Destination of the highest byte transferred.
000326 *  LOWTR is the lowest byte to be transferred.
000327 *  HIGHTR is the highest byte to be transferred.
000328 * ON EXIT:
000329 *  LOWTR is unchanged, HIGHTR is somewhere within a page of LOWTR,
000330 *  HIGHDS is lowest address transferred into.
000331 BLTU:           JSR       REASON
000332                STA       STREND                  ;THIS IS WHAT EVERYBODY CALLS
000333                STY       STREND+1
000334                STX       STRENDB
000335 BLTUC          LDY       HIGHTR                  ;SUBTRACT LOWTR FROM HIGHTR
000336                CPY       LOWTR                   ;TO FIND OUT HOW MUCH LEFT TO MOVE.
000337                LDA       HIGHTR+1                ;AND MOVE PAGES IF MORE THAN A PAGE.
000338                SBC       LOWTR+1
000339                LDY       HIGHTRB
000340                LDX       LOWTRB
000341                JSR       FIXAYX
000342                CPY       #0                      ;MORE THAN A BANK!  MOVE PAGES.
000343                BNE       MV256
000344                CMP       #1                      ;MORE THAN A PAGE?
000345                BCC       MVEND                   ;IF NOT, FINISH UP.
000346 MV256:         LDX       #HIGHDS                 ;NOW MOVE ONE PAGE OF DATA "UP" IN MEM
000347                JSR       SUB256
000348                LDX       #HIGHTR
000349                JSR       SUB256                  ;ADJUST DOWN A PAGE...
000350                LDY       #$FF                    ;MOVE IT NOW..
000351 BLK2:          LDA       (HIGHTR),Y
000352                STA       (HIGHDS),Y
000353                DEY
000354                BNE       BLK2
000355                LDA       (HIGHTR),Y              ;MOVE ONE MORE BYTE
000356                STA       (HIGHDS),Y
000357                JMP       BLTUC                   ;AND LOOP...
000358 MVEND:         LDA       HIGHTR                  ;CARRY IS SET
000359                SEC
000360                SBC       LOWTR
000361                BEQ       MVDONE                  ;ALL DONE, HOW  CONVENIENT
000362                PHA                               ;SAVE DIFFERENCE (HIGHTR-LOWTR) (IS #BYTES TO MOVE  MOD 256)
000363                LDY       HIGHDSB                 ;ADJUST HIGHDS TO FINAL MOVE LOCATION
000364                STA       HIGHDSB                 ;CHEAP-SHIT TEMPORARY
000365                LDA       HIGHDS
000366                SEC
000367                SBC       HIGHDSB                 ;HIGHDS-DIFFERENCE(HIGHTR-LOWTR)
000368                STA       HIGHDS
000369                LDA       HIGHDS+1
000370                SBC       #0
000371                CPY       #$80                    ;ARE WE MOVING IN THE STACK?
000372                BCC       *+5
000373                JSR       FIXSB2
000374                STA       HIGHDS+1
000375                STY       HIGHDSB
000376                PLA
000377                TAY                               ;GET INDEX OF # BYTES TO MOVE
000378                DEY
000379                LDA       (LOWTR),Y
000380                STA       (HIGHDS),Y
000381                DEY
000382                CPY       #$FF
000383                BNE       *-7
000384 MVDONE:        RTS
000385 SUB256:        LDA       1,X                     ;LOWER PTR BY ONE PAGE
000386                LDY       SYSPAG,X                ;GET BANK
000387                SEC
000388                SBC       #1
000389                JSR       FIXSBC
000390                STA       1,X
000391                TYA                               ;THE !#$%&'() 6502 DOESN'T HAVE STY ABS,X
```

```
000392                STA       SYSPAG,X
000393                RTS
000394 * MOVE MEMORY DOWN ROUTINE: MOVE (LOWTR)<(INDEX1).(STREND)
000395 * PRESERVES LOWTR. ADJUSTS ALL VARIABLE TABLE POINTERS, DOS BUFFERS,
000396 * ETC.
000397 MVDWN:          LDA       LOWTR                   ;SAVE LOWTR ON THE STACK
000398                PHA
000399                LDA       LOWTR+1
000400                PHA
000401                LDA       LOWTRB
000402                PHA
000403                LDY       #0
000404 MVDWN0         LDA       INDEX1                  ;IF NO MEMORY TO MOVE,
000405                CMP       STREND                  ;DON'T MOVE ANY!
000406                BNE       MVDWN1
000407                LDA       INDEX1+1
000408                CMP       STREND+1
000409                BNE       MVDWN1
000410                LDA       INDEX1B
000411                CMP       STRENDB
000412                BEQ       MVDWN3
000413 MVDWN1         LDA       (INDEX),Y               ;MOVE A BYTE
000414                STA       (LOWTR),Y
000415                INC       INDEX1                  ;NEXT BYTE TO MOVE
000416                BNE       MVDWN4
000417                LDX       INDEX1+1
000418                INX                               ;INC INDEX1+1
000419                CPX       #MAXPG
000420                BCC       *+7
000421                LDX       #MINPG
000422                INC       INDEX1B
000423                STX       INDEX1+1
000424 MVDWN4         INC       LOWTR
000425                BNE       MVDWN0
000426                LDX       LOWTR+1                 ;INC LOWTR+1
000427                INX
000428                CPX       #MAXPG
000429                BCC       *+7
000430                LDX       #MINPG
000431                INC       LOWTRB
000432                STX       LOWTR+1
000433                BNE       MVDWN0                  ;ALWAYS
000434 MVDWN3:        PLA                               ;RESTORE LOWTR
000435                STA       LOWTRB
000436                PLA
000437                STA       LOWTR+1
000438                PLA
000439                STA       LOWTR
000440 * HEY MAN, LIKE WOW, THE STUFF IS MOVED. (WHAT STUFF OCCIFER?)
000441                LDX       #5                      ;FINISH THE MOVE OPERATION.
000442 MVDWN2         CLC                               ;ADJUST THE POINTERS
000443                LDA       VARTAB-1,X
000444                ADC       DELTA
000445                STA       VARTAB-1,X
000446                LDA       VARTAB,X
000447                ADC       DELTA+1
000448                LDY       VARTABB-1,X
000449                JSR       FIXADC
000450                STA       VARTAB,X
000451                TYA
000452                ADC       DELTAB
000453                STA       VARTABB-1,X
000454                DEX
000455                DEX
000456                BPL       MVDWN2                  ;DO THE ZERO PAGE POINTERS FOR BASIC3
000457                RTS
000458 * MOVE UP ROUTINE. (STREND+DELTA)<(LOWTR).(STREND)
000459 * PRESERVES LOWTR
000460 MVUP:          LDA       STREND+1                ;THIS REALLY JUST DOES A 'JSR BLTU'
000461                STA       HIGHTR+1                ;BUT IT'S MORE CONVENIENT THIS WAY
000462                LDA       STRENDB
000463                STA       HIGHTRB
000464                LDA       STREND
000465                STA       HIGHTR                  ;HIGHEST LOC TO MOVE=HIGHTR
000466                CLC
000467                ADC       DELTA                   ;CALCULATE THE DESTINATION OF THE MOVE
000468                STA       HIGHDS
000469                LDA       STREND+1
000470                ADC       DELTA+1
000471                LDY       STRENDB
```

```
000232                LDA       #RESLSTB
000233                STA       FACB
000234                LDA       BUF,X                ;GET THE NEXT CHAR OF INPUT
000235                BEQ       GCRNCHED             ;IF AN EOL, GO HOME!
000236                AND       #$7F
000237                CMP       #$3A                 ;COLON?
000238                BEQ       ONCRNC               ;YES, TURN CRUNCHING BACK ON
000239                CMP       #$2C                 ;COMMA?
000240                BNE       CL1
000241                BIT       INTFLG
000242                BPL       *+6
000243                ROR       DORES
000244                BMI       CL1
000245                BIT       DORES                ;IF A COMMA, AND A DISK COMMAND, RE-ALLOW CRUNCHING
000246                BVS       CL1                  ;NOT A DISK COM
000247 ONCRNC:        STA       DORES                ;THIS WILL RE-ALLOW CRUNCHING
000248 CL1:           BIT       DORES                ;CRUNCH THIS CHAR?
000249                BMI       STUFIT               ;NO, STICK IT
000250                CMP       #$21                 ;IGNORE SPACES
000251                BCS       *+5
000252                JMP       NXCHR
000253                CMP       #'!'                 ;REM TOKEN?
000254                BCC       STUFIT
000255                BNE       SHORT1
000256                LDA       #REMTK
000257                BCS       STUFIT
000258 SHORT1         CMP       #'?'                 ;PRINT TOKEN?
000259                BCC       STUFIT
000260                BNE       NPRNT
000261                LDA       #PRINTK
000262                BCS       STUFIT
000263 GNMTCH         JMP       NOMTCH
000264 NPRNT          STX       TEMP                 ;SAVE FOR WHEN LOOPING BACK IN
000265 NPR2           LDX       TEMP                 ;LOOP BACK TO HERE
000266 NPR3           LDA       BUF,X
000267                BEQ       NSPC
000268                AND       #$7F
000269                CMP       #$21                 ;SPACES?
000270                BCS       NSPC                 ;NO
000271                INX                            ;YES, SKIP IT
000272                BNE       NPR3
000273 NSPC           CMP       #'A'+$20
000274                BCC       NSP2
000275                CMP       #'Z'+$21
000276                BCS       NSP2
000277                AND       #$DF                 ;KILL $20 BIT SO UPPER=LOWER CASE.
000278 NSP2           EOR       (FAC),Y
000279                INY
000280                INX
000281                ASL       A                    ;LEAST SIGN. 7 BITS MUST MATCH
000282                BNE       GNMTCH               ;NAW, DAMN HIM!
000283                BCC       NPR3                 ;SURE DOES SO FAR (NOT END OF RESERVED WORD)
000284 * WE FOUND A RESERVED WORD!(I THINK- SEE IF IMBEDDED IN NON-ALPHA,
000285 * NON-DIGIT DELIMS)
000286                LDA       BUF,X                ;CHECK CHAR AFTER THE WORD
000287                AND       #$7F
000288                JSR       CKSEP                ;IS IT A SEPERATOR?
000289                BCC       YMTCH                ;YUP, IT MATCHED
000290                DEY                            ;DID THE WORD END ON A SEPARATOR?
                                                                 I.E., LOMEM: OR DEC(?
000291                LDA       (FAC),Y              ;CHECK LAST CHAR IN WORD
000292                AND       #$7F                 ;WIPE HIGH BIT
000293                JSR       CKSEP
000294                BCC       YMTCH                ;YES IT MATCHED
000295                LDA       COUNT                ;WAS IT A FN TOKEN?
000296                CMP       #EXFNSTK-1
000297                BEQ       YMTCH
000298                CMP       #EXFNSTK
000299                BEQ       YMTCH
000300                CMP       #FNTK
000301                BNE       GNMTCH               ;NO.
000302                BIT       COUNT+1              ;MUST BE AN ESCAPE TOKEN
000303                BPL       GNMTCH
000304 YMTCH          TXA                            ;SAVE X-REG.
000305                PHA                            ;ON THE STACK
000306                LDX       TEMP                 ;WAS THE CHARACTER BEFORE IT A SEPERATOR?
000307                LDA       BUF-1,X
000308                JSR       CKSEP
000309                PLA                            ;RESTORE X-REG
000310                TAX
```

```
000311                BCS       GNMTCH
000312                DEX                           ;DON'T GO BEYOND RESERVED WORD IN PROGRAM
000313                LDA       COUNT+1             ;IS IT AN ESCAPE TOKEN?
000314                BMI       STUFIT              ;YES, STUFF THE $FF
000315                LDA       COUNT               ;NO, JUST STUFF A NORMAL TOKEN
000316 STUFIT         LDY       BUFPTR              ;GET INDEX WHERE TO PUT THIS BYTE
000317                STA       BUF-3,Y
000318                INY                           ;ADVANCE POINTER
000319                CMP       #$FF                ;ESCAPE TOKEN?
000320                BNE       NESC                ;  THERE IS NO ESCAPE!!!!!!
000321                LDA       COUNT               ;IF SO, STUFF BOTH BYTES
000322                STA       BUF-3,Y
000323                INY
000324                LDA       #0                  ;DON'T WANT TO MATCH REMTK OR STUFF
000325 NESC           STY       BUFPTR              ;SAVE BACK THE POINTER
000326 NXCHR          INX                           ;GET NEXT CHAR IN THE LINE
000327                CMP       #REMTK              ;DID WE STUFF A REMTK?
000328                BEQ       REMIT               ;IF SO, REM-ARKABLE
000329                CMP       #DATATK
000330                BEQ       ITDIR15
000331                CMP       #$22
000332                BEQ       DOQUOT
000333                CMP       #IMAGETK+2
000334                BCS       GCLOOP
000335                LDY       LOWTR+1             ;ARE WE IN IMMEDIATE MODE?
000336                INY
000337                BEQ       ITDIR               ;YES, DON'T CRUNCH
000338                CMP       #DSKCOMS+1          ;IF IN DEFERRED MODE...
000339                BCC       GCLOOP              ;CRUNCH NORMAL
000340 ITDIR          CMP       #OPENTK             ;OPEN DOES IT BACKWARDS.
000341                BNE       ITDIR1
000342                ROR       INTFLG              ;SET HIGH BIT
000343                BMI       GCLOOP              ;ALWAYS
000344 ITDIR1         CMP       #INVOKTK
000345                BEQ       ITDIR15
000346                CMP       #LDTKN
000347                BCC       GCLOOP
000348                CMP       #RENMTK             ;IS IT RENAME?
000349                BNE       ITDIR2              ;NO
000350 ITDIR15        SEC
000351                DFB       44
000352 ITDIR2         CMP       #DSKCOMS+1          ;DORES: BIT 7 ON- DONT CRUNCH UNTIL A :
000353                ROR       DORES               ;SET BIT 7 OF DORES
000354                SEC
000355                ROR       DORES               ;DORES: BIT 6 OFF-- DON'T CRUNCH UNTIL A COMMA, ELSE UNTIL ':'
000356 GCLOOP         JMP       CLOOP
000357 REMIT          LDA       #0                  ;A 'REM' ENDS AT THE END OF THE LINE ONLY
000358 DOQUOT         DEX                           ;START WITH CORRECT CHAR FROM INPUT LINE
000359                STA       ENDCHR              ;THIS IS WHAT THE UN-CRUNCHED AREA MAY END ON
000360                LDY       BUFPTR              ;GET WHERE TO STUFF CHARS
000361                DEY
000362 DOQ2           INY
000363                INX
000364                LDA       BUF,X
000365                STA       BUF-3,Y             ;MOVE CHAR
000366                STY       BUFPTR
000367                BEQ       CRNCHED             ;END OF THE LINE
000368                CMP       ENDCHR              ;END CHAR REACHED?
000369                BNE       DOQ2                ;NO, LOOP
000370                INY
000371                STY       BUFPTR              ;SAVE BACK
000372                INX
000373                JMP       GCLOOP
000374 * THIS WORD DIDN'T MATCH. TRY THE NEXT ONE
000375 NOMTCH         DEY
000376                BEQ       FNDNXT
000377                DEY
000378 FNDNXT         INY                           ;FIND THE NEXT RESERVED WORD
000379                LDA       (FAC),Y
000380                BPL       FNDNXT              ;WORD ENDS ON A NEGATIVE CHARACTER
000381                SEC
000382                TYA                           ;ADD LENGTH OF THIS WORD TO (FAC)
000383                ADC       FAC
000384                STA       FAC
000385                BCC       *+4
000386                INC       FAC+1               ;NEVER CROSSES BANK BOUNDARY.
000387                LDY       #0
000388                LDX       COUNT
000389                INX                           ;ADVANCE TOKEN COUNT
000390                CPX       #SCRATK+1           ;PAST THE STATEMENTS?
```

```
000391                BNE       FND2
000392                LDX       #$FF                    ;SET ESCAPE MODE
000393                STX       COUNT+1
000394                LDX       #$80
000395 FND2           STX       COUNT
000396                LDA       (FAC),Y                 ;ARE WE AT THE END OF THE LIST?
000397                BNE       GNPR2                   ;NO, KEEP GOING
000398                LDX       TEMP                    ;MAKE SURE TO GET THE RIGHT CHAR
000399                LDA       BUF,X                   ;YES, STUFF THIS CHARACTER
000400                AND       #$7F
000401                JMP       STUFIT
000402 * LINE IS FINISHED CRUNCHING
000403 CRNCHED        LDY       BUFPTR
000404                STA       BUF-3,Y
000405                STA       BUF-2,Y                 ;I DON'T KNOW WHY, BUT IT NEEDS THIS
000406                LDA       #0
000407                STA       TXTPTRB
000408                LDA       #<BUF-1
000409                STA       TXTPTR+1
000410                LDA       #>BUF-1
000411                STA       TXTPTR
000412                INY
000413                RTS
000414 GNPR2           JMP       NPR2
000415 CKSEP:          JSR       ISLETC                  ;IS IT A LETTER IN BETWEEN
000416                BCS       CKRTS
000417                CMP       #'9'+1                  ;IF NOT A DIGIT OR A SPECIAL, ITS A SEPERATOR.
000418                BCS       CKRT1
000419                CMP       #'0'                    ;A DIGIT?
000420                BCS       CKRTS
000421                CMP       #'.'
000422                BEQ       CKRTS
000423 CKRT1           CLC
000424 CKRTS           RTS
000425 ; FNDLIN searches the program text for the line whose number is passed
000426 ;   in LINNUM.  There are only two possible returns:
000427 ; 1) Carry Set.
000428 ;    LOWTR points to the link byte in the line that was searched for.
000429 ; 2) Carry Clear.
000430 ;    Line not found.  LOWTR points to the first line in the program with
000431 ;    a line number greater than the one sought after.
000432 FNDLIN:         LDA       TXTTAB
000433                LDX       TXTTAB+1                ;LOAD X,A WITH TXTTAB
000434                LDY       TXTTABB                 ;Y WITH BANK #.
000435 FNDLNC0         STY       LOWTRB
000436 FNDLNC1         STX       LOWTR+1
000437 FNDLNC          STA       LOWTR
000438                LDY       #0                      ;POINT TO LINK.
000439                LDA       (LOWTR),Y               ;SEE IF LINK IS 0
000440                BEQ       FLINRT
000441                LDY       #2
000442                TAX
000443                LDA       LINNUM+1                ;COMP HIGH ORDERS OF LINE NUMBERS.
000444                CMP       (LOWTR),Y
000445                BCC       FLNRTS                  ;NO SUCH LINE NUMBER.
000446                BNE       AFFRTS                  ;CHECK NEXT LINE.
000447                LDA       LINNUM
000448                DEY
000449                CMP       (LOWTR),Y               ;COMPARE LOW ORDERS.
000450                BCC       FLNRTS                  ;NO SUCH NUMBER.
000451                BEQ       FLNRTS                  ;RETURN WITH CARRY SET.
000452 AFFRTS:         TXA
000453                CLC
000454                ADC       LOWTR                   ;COMPUTE NEXT RELATIVE LINE POSITION
000455                BCC       FNDLNC                  ;BRANCH IF DONE
000456                LDX       LOWTR+1                 ;INC LOWTR+1
000457                INX
000458                CPX       #MAXPG
000459                BCC       FNDLNC1
000460                PHA
000461                TXA
000462                SBC       #MAXPG-MINPG
000463                TAX
000464                PLA
000465                INC       LOWTRB
000466                BNE       FNDLNC1                 ;ALWAYS
000467                BCS       FNDLNC                  ;ALWAYS BRANCHES.
000468 FLINRT:         CLC                               ;C MAY BE HIGH.
000469 FLNRTS:         RTS                               ;RETURN TO CALLER.
000470 ;
```

```
000153                  BMI       PRIT25                    ;YES, GO TOSS IT UP
000154 RESCR1:          JSR       GETNXTW
000155                  BPL       RESCR1                    ;NO, CONTINUE PASSING.
000156                  BMI       RESRCH
000157 PRIT25:          EQU       *
000158                  LDA       LSTPNT                    ;SOMETIMES WE DON'T EVEN WANT THEM FOR
000159                  CMP       #3                        ; SPECIAL FUNCTIONS (LIKE AT THE
                                                                        BEGINNING OF A LINE)
000160                  BEQ       PRIT3
000161                  TYA                                 ;SAVE INDEX TO TOKEN'S TEXT
000162                  PHA
000163                  LDY       LSTPNT                    ;DON'T WANT EXTRA SPACES WHEN
                                                                        THE LAST THING WAS A TOKEN
000164                  DEY
000165                  BIT       INPFLG                    ;AN ESCAPE TOKEN?
000166                  BMI       PRIT26
000167                  DEY
000168                  LDA       TKNSAV                    ;CHEK FOR VALID ESC TOKEN
000169                  CMP       #ONEFUN-$80               ;A SPECIAL WORD?
000170                  BCS       PRIT27
000171 PRIT26           LDA       (VARNAM),Y                ;SO CHECK FOR THAT CASE...
000172                  CMP       #':'
000173                  BEQ       PRIT27
000174                  ASL       A                         ;HI BIT TELLS ALL. SHIFT IT TO CARRY
000175 PRIT27           PLA                                 ;FIRST MUST RESTORE Y
000176                  TAY
000177                  BCS       PRIT3                     ;NO SPACE IF LAST WAS TOKEN
000178                  JSR       ROUTSPC
000179 PRIT3:           JSR       GETNXTW                   ;PRINT THE RESERVED WORD
000180                  BMI       PRFINIS
000181                  JSR       ROUTDO                    ;PRINT THE CHARACTER
000182                  JMP       PRIT3                     ;ALL OF THE WORD
000183 PRFINIS:         AND       #$7F
000184                  JSR       ROUTDO                    ;PRINT LAST CHAR OF THE WORD
000185                  LDY       LSTPNT                    ;IS THE WORD FOLLOWED BY A SEPERATOR?
000186                  INY
000187                  CMP       #'A'                      ;DON'T OUTPUT EXTRA SPACES FOR TOKENS WITH
000188                  BCC       JPL2                      ; NON ALPHA ENDINGS SUCH AS PR#, CHR$(, ETC...
000189                  LDA       (VARNAM),Y                ;IF A TOKEN ALSO, PRINT A SPACE
000190                  BMI       PRF2
000191                  JSR       CKSEP
000192                  BCC       JPL2                      ;BRANCH IF A SEPERATOR
000193 PRF2:            JSR       ROUTSPC
000194 JPL2:            DEY
000195                  JMP       PLOOP2
000196                  SBTL      "RELATIVE FORPNT CREATE."
000197 PNTREL           STX       KIMY
000198                  LDX       #FORPNT
000199 PNTRL1           LDA       SMVARS                    ;MAKE THE VARIABLE POINTER RELATIVE
000200                  SEC                                 ;TO SMVARS. THIS IS IN CASE
000201                  SBC       0,X                       ;THE SIMPLE VARIABLE TABLE CHANGES DURING
000202                  STA       0,X                       ;LOOP STATEMENTS.
000203                  LDA       SMVARS+1
000204                  SBC       1,X
000205                  LDY       SMVARSB
000206                  JSR       FIXSBC
000207                  STA       1,X
000208                  TYA
000209                  SBC       SYSPAG,X
000210                  STA       SYSPAG,X
000211                  LDX       KIMY
000212                  RTS
000213                  SBTL      "THE 'FOR' STATEMENT."
000214 FOR:             JSR       LET
000215                  LDA       VALTYP
000216                  BEQ       *+5
000217                  JMP       TMERR                     ;ONLY INTEGER AND FLOATING.
000218                  LDA       ISARA
000219                  ROL       A
000220                  LDA       INTFLG                    ;WAS IT AN INT?
000221                  ADC       #0                        ;IN TO LOW BIT.
000222                  STA       TEMPFOR                   ;HIGH BIT IF INTEGER. LOW BIT IF ARRAY.
000223 ;READ THE VARIABLE AND ASSIGN IT
000224 ;THE CORRECT INITIAL VALUE AND STORE
000225 ;A POINTER TO THE VARIABLE IN VARPNT.
000226                  JSR       RELPTR                    ;MAKE FORPNT TO START OF VARS TABLE.
000227                  LDA       FORPNT+1
000228                  LDY       FORPNTB
000229                  JSR       FIXAY
000230                  STA       FORPNT+1
```

```
000231                CPY       #$FE
000232                BCS       *+5
000233                JMP       OVERR
000234                JSR       FNDFOR                    ;PNTR IS IN VARPNT, AND FORPNT.
000235                BNE       NOTOL
000236                LDA       #1
000237                STA       HIGHDSB
000238                STA       HIGHTRB
000239                STA       LOWTRB
000240                STA       HIGHDS+1
000241                STA       HIGHTR+1
000242                STA       LOWTR+1
000243                INX
000244                STX       HIGHTR
000245                TXA
000246                CLC
000247                ADC       #FORSIZ
000248                STA       HIGHDS
000249                TSX
000250                STX       LOWTR
000251                JSR       BLTUC
000252                TSX
000253                TXA
000254                CLC
000255                ADC       #FORSIZ
000256                TAX
000257                TXS
000258 NOTOL:         PLA                                 ;GET RID OF NEWSTT RETURN ADDRESS
000259                PLA                                 ;IN CASE THIS IS A TOTALLY NEW ENTRY.
000260                LDA       #10
000261                JSR       GETSTK                    ;MAKE SURE 20 BYTES ARE AVAILABLE.
000262                JSR       SVTXT                     ;SAVE THE TEXT.
000263                JSR       DATA                      ;MOVE TXTPTR TO END OF FOR STATEMENT.
000264                JSR       PSHTXT3                   ;PUT FOR ENTRY ON THE STACK.
000265                PLA                                 ;GET RID OF THE 'GOSUB' TOKEN
000266                JSR       RSTTXT                    ;RESTORE TXTPTR TO PREVIOUS VALUE
000267                LDA       #TOTK
000268                JSR       MSTESC                    ;'TO' IS NECESSARY.
000269                JSR       CHKNUM                    ;VALUE MUST BE A NUMBER.
000270                JSR       FRMNUM                    ;GET UPPER VALUE INTO FAC.
000271                LDA       FACSGN                    ;PACK FAC.
000272                ORA       #127
000273                AND       FACHO
000274                STA       FACHO                     ;SET PACKED SIGN BIT.
000275                LDA       #>LDFONE
000276                LDY       #<LDFONE
000277                STA       INDEX1
000278                STY       INDEX1+1
000279                JMP       FORPSH                    ;PUT FAC ONTO STACK, PACKED.
000280 LDFONE:        LDA       #>FONE
000281                LDY       #<FONE                    ;PUT 1.0 INTO FAC.
000282                LDX       #0
000283                JSR       MOVFM
000284                LDA       #STEPTK
000285                JSR       TRYESC                    ;A STEP IS GIVEN?
000286                BNE       ONEON                     ;NO. AUME 1.0.
000287                JSR       CHRGET                    ;YES. ADVANCE POINTER.
000288                JSR       FRMNUM                    ;READ THE STEP.
000289 ONEON          JSR       SIGN                      ;GET SIGN IN ACCA.
000290                JSR       PUSHF                     ;PUSH FAC ONTO STACK (THRU A).
000291                LDA       FORPNT+1
000292                PHA
000293                LDA       FORPNT
000294                PHA                                 ;PUT PNTR TO VARIABLE ON STACK.
000295                LDA       TEMPFOR
000296                PHA
000297                LDA       #FORTK                    ;PUT A FORTK ONTO STACK.
000298                PHA
000299 ;FALL INTO NEWSTT
000300                SBTL      "NEW STATEMENT FETCHER."
000301 ; Back here for new statement.  Char pointed to by TXTPTR is a : or
000302 ;   the End-of-line terminator.  The address of its location is left
000303 ;   on the stack when a statement is executed, so that it can merely
000304 ;   do a RTS when it is done.
000305 NEWSTT:        EQU       *
000306                LDA       #0
000307                STA       CMDFLG                    ;Reset CMDFLG to 0
000308                JSR       SETSOS                    ;Set prefix to SOS PREFIX...
000309                TSX                                 ;IN CASE OF ERROR.
000310                STX       REMSTK
```

```
000153                JSR       PSHTXT                    ;DO A GOSUB
000154                JSR       RSTTXT                    ;RESTORE THE TEXTPTR.
000155                JSR       DOONGT
000156                RTS
000157 DOONGT          JSR       CHRGET                    ;EAT THE COMMA.
000158                JSR       LINGET                    ;FIND THE LINE
000159                JMP       GOTOB                     ;AND POSITION TO IT
000160 ONGLP1:         JSR       CHRGET                    ;ADVANCE AND SET CODES.
000161                JSR       LINGET
000162                CMP       #44                       ;IS IT A COMMA?
000163                BEQ       ONGLOP
000164                PLA                                 ;REMOVE STACK ENTRY (TOKEN).
000165 ONGRTS:         RTS                                 ;EITHER END-OF-LINE OR SYNTAX ERROR.
000166 GOERR           JMP       ONERR
000167 GOKBD           JMP       ONKBD
000168                PAGE
000169                SBTL      "LINGET -- READ LINE # INTO LINNUM"
000170 ; 'LINGET' reads a line number from the Current Text position
000171 ;   Line numbers range from 0 to 64000-1.
000172 ;   The answer is returned in 'LINNUM'.
000173 ; 'TXTPTR' is updated to point to the termination character,
000174 ;   A = the termination character with condition codes set up
000175 ;   to reflect its value.
000176 LINGET:         LDX       #0
000177                STX       LINNUM                    ;INITIALIZE LINE NUMBER TO ZERO.
000178                STX       LINNUM+1
000179 MORLIN:         BCS       ONGRTS                    ;IT IS NOT A DIGIT.
000180                SBC       #'0'-1                    ;-1 SINCE C=0.
000181                STA       CHARAC                    ;SAVE CHARACTER.
000182                LDA       LINNUM+1
000183                STA       INDEX
000184                CMP       #25                       ;LINE NUMBER WILL BE .LT. 64000?
000185                BCS       SNERR3
000186                LDA       LINNUM
000187                ASL       A                         ;MULTIPLY BY 10.
000188                ROL       INDEX
000189                ASL       A
000190                ROL       INDEX
000191                ADC       LINNUM
000192                STA       LINNUM
000193                LDA       INDEX
000194                ADC       LINNUM+1
000195                STA       LINNUM+1
000196                ASL       LINNUM
000197                ROL       LINNUM+1
000198                LDA       LINNUM
000199                ADC       CHARAC                    ;ADD IN DIGIT.
000200                STA       LINNUM
000201                BCC       NXTLGC
000202                INC       LINNUM+1
000203 NXTLGC:         JSR       CHRGET
000204                JMP       MORLIN
000205                PAGE
000206                SBTL      "'LET' CODE."
000207 LET:            JSR       MYPTRGET                  ;GET PTR TO VAR INTO VARPNT, FORPNT
000208                JSR       CHKEQL                    ;'=' IS NECESSARY
000209                LDA       INTFLG                    ;SAVE FOR LATER.
000210                PHA
000211                LDA       ISARA                     ;GET WHEATHER ARRAY OR NOT.
000212                PHA
000213                JSR       FRMEVL                    ;Get value of formula into 'FAC'.
000214                JMP       LETP3
000215 LETP2:          LDA       INTFLG                    ;DOS INTERFACE ENTERS HERE
000216                PHA
000217                LDA       ISARA                     ;THIS LINE MAY NOT BE NEEDED
000218                PHA
000219 LETP3           PLA                                 ;GET BACK ISARA.
000220                STA       ISARA
000221                BIT       VALTYP                    ;MAKE SURE 'VALTYP' SPECIFIES NUMERIC.
000222                BMI       COPSTR                    ;IF NUMERIC, COPY IT.
000223                BVS       BMOVVF
000224                PLA                                 ;GET NUMBER TYPE.
000225                STA       INTFLG                    ;FOR "FOR".
000226 QINTGR:         BPL       COPFLT                    ;STORE A FLTING NUMR.
000227                JSR       ROUND                     ;ROUND INTEGER.
000228                JSR       AYINT                     ;MAKE 2-BYTE NUMBER.
000229                LDY       #0
000230                LDA       FACMO                     ;GET HIGH.
000231                STA       (FORPNT),Y                ;STORE IT.
000232                INY
```

```
000233                  LDA       FACLO                   ;GET LOW.
000234                  STA       (FORPNT),Y
000235                  RTS
000236 COPFLT:          JMP       MOVVF                   ;PUT NUMBER AT FORPNT.
000237 BMOVVF           PLA                               ;POP OFF VALTYP, WE DON'T NEED IT.
000238 BMOVF1           LDX       FORPNTB
000239                  LDA       FORPNT                  ;LOW BYTE OF PLACE TO MOVE TO.
000240                  LDY       FORPNT+1
000241                  JMP       STFACT
000242 COPSTR:          EQU       *
000243                  PLA                               ;IF STRING, NO INTFLG.
000244 INPCOM:          EQU       *
000245 ; ADD IN DIGIT TO FAC.
000246                  LDY       FACMOB                  ;WAS THIS A TEMP, OR VARIABLE?
000247                  BNE       COPY                    ;IT IS A VARIABLE, MAKE A COPY.
000248                  LDA       FACMO                   ; else here for DNTCPY:
000249                  LDY       FACMO+1
000250                  LDX       FACMOB
000251                  JMP       COPY.C
000252 COPY:            LDY       #0
000253                  LDA       (FACMO),Y
000254                  JSR       STRINI                  ;GET ROOM TO COPY STRING INTO.
000255 COPY.M           LDA       DSCPNT
000256                  LDY       DSCPNT+1                ;GET POINTER TO OLD DESCRIPTOR, SO
000257                  LDX       DSCPNTB
000258                  STA       STRNG1
000259                  STY       STRNG1+1                ;MOVINS CAN FINSTRING.
000260                  STX       STRNG1B
000261                  JSR       MOVINS                  ;COPY IT.
000262                  JSR       PUTNEW
000263                  LDA       FACMO
000264                  LDY       FACMO+1
000265                  LDX       FACMOB
000266 COPY.C:          STA       DSCPNT
000267                  STX       DSCPNTB
000268                  STY       DSCPNT+1                ;REMEMBER POINTER TO DESCRIPTOR.
000269                  JSR       FRETMS                  ;FREE UP THE TEMPORARY WITHOUT
000270 ;   FREEING UP ANY STRING SPACE.
000271                  LDA       FORPNT
000272                  LDY       FORPNT+1
000273                  LDX       FORPNTB
000274                  JSR       NOTNW2                  ;PUT THE POINTER TO THE STRING IN INDEX.
000275                  JSR       FRESPA                  ;FREE THE BUGGER IF POSSIBLE
000276                  LDY       #$2                     ;RESET Y.
000277 COPY.S           LDA       (DSCPNT),Y
000278                  STA       (FORPNT),Y
000279                  DEY
000280                  BPL       COPY.S
000281                  TAX                               ;SET Z FLAG.
000282                  BEQ       LETRTS                  ;DON'T BUILD BACKPOINTER FOR NULL STRINGS
000283 FIXBAK           JSR       RELPTR
000284                  LDA       FORPNT+1
000285                  LDY       FORPNTB
000286                  JSR       FIXAY                   ;PACK A WITH LOW BIT FROM Y.
000287                  CPY       #$FF                    ;CHECK IF <64K!
000288                  BCC       VAERR                   ;VARIABLE ERROR!
000289                  LDY       #2                      ;FOR (),Y.
000290                  STA       (HIGHDS),Y
000291                  DEY
000292                  LDA       FORPNT
000293                  STA       (HIGHDS),Y
000294                  TXA
000295                  BMI       *+5
000296                  LDA       #SIMTYP                 ;$41
000297                  DFB       44                      ;SKIP 2
000298                  LDA       #ARYTYP                 ;$81
000299                  DEY
000300                  STA       (HIGHDS),Y              ;STORE TYPE BYTE.
000301 LETRTS           RTS
000302 RELPTR           LDX       ISARA                   ;IS THIS AN ARRAY?
000303                  BPL       ISSIMP                  ;NO, SIMPLE SIMON
000304 RELPTR2          LDA       ARYTAB
000305                  SEC
000306                  SBC       FORPNT                  ;FORPNT=RELATIVE POINTER TO VARIABLE.
000307                  STA       FORPNT
000308                  LDA       ARYTAB+1
000309                  SBC       FORPNT+1
000310                  LDY       ARYTABB
000311                  JSR       FIXSBC
000312                  STA       FORPNT+1
```

```
000313                TYA
000314                SBC       FORPNTB
000315                STA       FORPNTB
000316                BCC       GOTRELA                 ;ALWAYS
000317                JMP       OMERR                   ;MORE THAN 64K OF ARRAYS GIVES "OUT OF MEMORY".
000318 ISSIMP         JSR       PNTREL
000319 GOTRELA        RTS
000320 VAERR          LDX       #ERRVA
000321                JMP       ERROR
000322                SBTL      "PRINT CODE."
000323 STRDON:        JSR       STRPRT
000324 NEWCHR:        JSR       CHRGOT                  ;REGET LAST CHARACTER.
000325 PRINT:         BEQ       CRDO                    ;TERMINATOR SO TYPE CRLF.
000326 PRINTC:        BEQ       PRTRTS                  ;HERE AFTER SEEING TAB(X) OR , OR ;
000327 ;  IN WHICH CASE A TERMINATOR DOES NOT
000328 ;  MEAN TYPE A CRLF BUT JUST RTS.
000329                LDA       #TABTK                  ;TAB FUNCTION?
000330                JSR       TRYESC
000331                BEQ       TABER                   ;YES.
000332                LDA       #SPCTK                  ;SPACE FUNCTION?
000333                JSR       TRYESC
000334                CLC                               ;REMEMBER IF IT IS.
000335                BEQ       TABER
000336                CMP       #44                     ;A COMMA?
000337                BEQ       COMPRT                  ;YES.
000338                CMP       #59                     ;A SEMICOLON?
000339                BNE       *+5                     ;NO.
000340                JMP       NOTABR                  ;YES.
000341                LDA       #$20                    ;WE CAN GET ANYTHING FROM FRMEVL
000342                STA       VALTYP
000343                JSR       FRMEVL                  ;EVALUATE THE FORMULA.
000344                BIT       VALTYP                  ;A STRING?
000345                BMI       STRDON                  ;YES.
000346                BVC       ISBIN
000347                JSR       LOUT
000348                LDA       #>NUMSTR
000349                STA       INDEX
000350                LDA       #<NUMSTR
000351                STA       INDEX+1
000352                LDA       #NUMSTRB
000353                STA       INDEXB
000354                LDX       LENUM
000355                JSR       STRPR3
000356                BEQ       NEWCHR                  ;ALWAYS TAKEN.
000357 ISBIN          JSR       FOUT
000358                JSR       STRLIT
000359                JMP       STRDON
000360 CRDO:          EQU       *
000361                LDA       #13                     ;MAKE TRMPOS LESS THAN LINE LENGTH.
000362                JSR       OUTDO
000363                STX       TEMP
000364                STY       KIMY
000365                LDX       FILNO                   ;IF OUTPUT TO A DEVICE, GIVE LF
000366                BMI       DOLF                    ;LINE FEED TO CONSOLE FOR SURE
000367                JSR       GTFLNO1
000368                LDA       FCB,Y                   ;IS IT A DEVICE (NOT A DISK FILE?)
000369                BPL       DOLFRT                  ;BRANCH IF A DISK FILE
000370 DOLF           LDA       #10                     ;OUTPUT THE LF
000371                JSR       OUTDO
000372 DOLFRT         LDX       #0
000373                STX       TRMPOS
000374                LDX       TEMP
000375                LDY       KIMY
000376 PRTRTS:        RTS
000377 PRTRTS1        BIT       VALTYP                  ;IS THIS REALY A STRING TYPE?
000378                BPL       PRTRTS                  ;ONLY STRINGS FALL THROUGH SO THEY
000379                JMP       FRECNOW                 ;FREE THE USED STRING.
000380 COMPRT         LDA       TRMPOS
000381                ADC       #$0F                    ;16 POSITIONS PER COLUMN
000382                AND       #$F0                    ;ROUND DOWN.
000383 *  CMP WNDWDTH ;ARE WE OUT OF THE WINDOW?
000384                SEC
000385                SBC       TRMPOS
000386                TAX
000387                JMP       XSPAC1                  ;PUT OUT THAT MANY SPACES.
000388 TABER:         PHP                               ;REMEMBER IF SPC OR TAB FUNCTION.
000389                JSR       GTBYTC                  ;GET VALUE INTO ACCX.
000390                STX       TEMP
000391                JSR       CHKCLS                  ;MAKE SURE A CLOSING PAREN
000392                JSR       DECTPT                  ;DON'T IGNORE NEXT THING IN LIST
```

```
000153                LDA       VALTYP
000154                PHA
000155                LDY       RNDGOT
000156                JSR       STRCP
000157                PLA
000158                STA       VALTYP
000159                JSR       CHANGTP                ;CHANGE TYPE TO WHAT IT SHOULD BE.
000160                JSR       LETP2                  ;DO THE ASSIGNMENT.
000161                JMP       STRDN2                 ;LOOP.
000162 QDATA:         BMI       DTLP1                  ;SEARCH FOR ANOTHER DATA STATEMENT.
000163                BIT       FILNO                  ;IS THIS A FILE INPUT?
000164                BMI       QD11
000165                JSR       DSKLIN                 ;INPUT A LINE FROM THE DISK
000166                JMP       DATBK
000167 QD11           JSR       OUTQST
000168                JSR       QINLIN                 ;GET ANOTHER LINE.
000169 DATBK:         STX       TXTPTR
000170                LDA       INPPTRB
000171                STA       TXTPTRB
000172                STY       TXTPTR+1               ;SET FOR 'CHRGET'.
000173 DATBK1:        LDY       #1
000174                JSR       ADDON
000175                BIT       VALTYP                 ;GET VALUE TYPE.
000176                BPL       NUMINS                 ;INPUT A NUMBER IF NUMERIC.
000177                BIT       INPFLG                 ;GET?
000178                BVC       SETQUT                 ;NO, GO SET QUOTE.
000179                INX
000180                STX       TXTPTR
000181 NOTERMS        LDA       #0                     ;ZERO TERMINATORS.
000182                STA       CHARAC
000183                BEQ       RESETC
000184 DTLP1          JMP       DATLOP
000185 SETQUT         LDY       #0
000186                LDA       (TXTPTR),Y
000187                STA       CHARAC                 ;ASSUME QUOTED STRING
000188                BIT       QUOTE                  ;DO WE TAKE ANYTHING?
000189                BMI       NOTERMS                ;IF SO, NO TERMINATORS!
000190                CMP       #34                    ;TERMINATORS OK?
000191                BEQ       NOWGET                 ;YES.
000192                LDA       #44                    ;COMMA.
000193                STA       CHARAC                 ;ONLY STOP ON COMMAS
000194 RESETC:        CLC
000195 NOWGET:        STA       ENDCHR
000196                LDA       TXTPTR
000197                LDX       TXTPTRB
000198                LDY       TXTPTR+1
000199                ADC       #0                     ;C IS SET PROPERLY ABOVE.
000200                BCC       NOWGE1
000201                INY
000202                CPY       #MAXPG
000203                BCC       *+5
000204                INX
000205                LDY       #MINPG
000206 NOWGE1:        JSR       STRLT2                 ;MAKE A STRING DESCRIPTOR FOR VALUE
000207                JSR       ST2TXT                 ;SET TEXT POINTER.
000208                JSR       INPCOM                 ;DO ASSIGNMENT.
000209                JMP       STRDN2
000210 NUMINS:        PHA
000211                LDA       BUF                    ;BLANK INPUT?
000212                BEQ       MAYBAD
000213 NUMINS2:       PLA
000214                BVS       NUMBCD
000215                JSR       FIN                    ;GET VALUE.
000216                BIT       INPFLG                 ;WATCH FOR GET'S!
000217                BVS       NUMINS3
000218                BMI       NUMINS3
000219                PHA
000220                LDA       ANYNUM
000221                BMI       MAYBAD
000222                PLA
000223 NUMINS3        LDA       INTFLG                 ;SET CODES ON FLAG.
000224                JSR       QINTGR                 ;GO DECIDE ON FLOAT.
000225 STRDN2:        JSR       CHRGOT                 ;READ LAST CHARACTER.
000226                BEQ       TRMOK                  ;':' OR EOL IS OK.
000227                CMP       #44                    ;A COMMA?
000228                BEQ       *+5
000229                JMP       TRMNOK
000230 TRMOK:         LDA       TXTPTR
000231                LDX       TXTPTRB
000232                LDY       TXTPTR+1
```

```
+------------------------------------------------------------------------------
|
|  File    : "B3EVALG.TEXT.PRETTY"
|  Created : Tuesday, December 30, 1997               5:14:25 PM
|  Modified: Wednesday, December 31, 1997             4:37:02 PM
|
+------------------------------------------------------------------------------

000001 ; ###########################################################################################
000002 ; #   PROJECT  :  Apple /// Business BASIC 1.3 (6502 Assembly Source Code)
000003 ; #   FILE NAME:  B3EVALG.TEXT
000004 ; ###########################################################################################
000005
000006                 SEC
000007                 SBC       #6
000008                 CMP       #1
000009                 BPL       *+4
000010                 LDA       #1
000011                 ASL       A
000012                 TAX
000013                 BIT       VALTYP                    ;BUT IF NOT REAL LOOK OUT!
000014                 BMI       ASTRNG                    ;IT'S A STRING.
000015                 CLC
000016                 BVC       NOTDBLP                   ;IT'S NOT DBL PRECISION EITHER
000017                 ADC       #NUMDSP                   ;MAP UP INTO DBL
000018                 TAX
000019                 BCC       RDY2GO
000020 NOTDBLP:        EQU       *
000021                 LDA       ARGSGN
000022                 EOR       FACSGN
000023                 STA       ARISGN
000024 RDY2GO:         LDA       OPDSPT-2,X
000025                 STA       JMPER+1
000026                 LDA       OPDSPT-1,X
000027                 STA       JMPER+2
000028                 LDA       FACEXP
000029                 JSR       JMPER                     ;DO OPERATION...
000030 FRMEV3:         PLA                                 ;GET CURRENT OP BACK AND LOOP.
000031                 JMP       FRMEVLZ
000032 ASTRNG:         CMP       #NUMDSP                   ;AN ADD ON STRINGS
000033                 BEQ       PLUSOK                    ;IS OK.
000034                 CMP       #2*RELNUM                 ;IS THIS A RELATIONAL?
000035                 BEQ       NOTDBLP                   ;YES, GO DO IT. NOTE: CARRY SET INDICATES STRING.
000036 EVALER:         JMP       SNERR
000037 PLUSOK          JSR       CONCAT                    ;GO CONCATENATE THE STRINGS
000038                 JMP       FRMEV3                    ;GET BACK OPERATOR AND LOOP.
000039 ;  HERE IS THE EVAL ROUTINE - EVALUATE A SINGLE
000040 ;   VALUE.  IF A FORMULA IN PARENTHESES, CALL
000041 ;   FRMEVL RIGHT BACK!
000042 EVAL            JSR       CHRGET                    ;GET FIRST CHAR OF VALUE.
000043                 BCS       EVAL1                     ;IF A DIGIT, MUST BE A CONSTANT...
000044 ACONSTNT:       LDA       VALTYP                    ;CHECK DATA TYPE.
000045                 CMP       #$21
000046                 BCC       RLBNY                     ;VALTYP 0 MEANS GET FLOATING POINT.
000047                 JMP       LINP                      ;LONG INTEGER INPUT INTO FAC.
000048 RLBNY           LDA       #0
000049                 STA       VALTYP
000050                 JSR       FIN                       ;BINAY FLOATING POINT INPUT.
000051                 BIT       ANYNUM
000052                 BMI       EVALER
000053                 RTS
000054 EVAL1           CMP       #$FF                      ;HANDLE ESCAPE TOKENS
000055                 BEQ       PARC22                    ;(FUNCTIONS OR RESERVED WORDS)
000056                 CMP       #$80                      ;A RES. WORD?
000057                 BCS       PARC23
000058                 JSR       ISLETC                    ;A LETTER? IF SO, A VARIABLE.
000059                 BCC       NOVAR                     ;NO.
000060                 JMP       ISVAR
000061 NOVAR           CMP       #'.'                      ;LEADING CHARACTER OF CONSTANT?
000062                 BEQ       ACONSTNT
000063                 CMP       #'+'
000064                 BEQ       EVAL
000065                 CMP       #34                       ;A QUOTE? A STRING?
000066                 BNE       EVAL3
000067 STRTXT:         LDA       TXTPTR
000068                 LDX       TXTPTRB
000069                 LDY       TXTPTR+1
000070                 ADC       #0                        ;TO INC, ADD C=1.
000071                 BCC       STRTX2
000072                 INY
```

```
000073                CPY       #MAXPG
000074                BCC       *+5
000075                INX
000076                LDY       #MINPG
000077 STRTX2:        JSR       STRLIT                ;YES. PROCESS IT.
000078                JMP       ST2TXT
000079 NOTDO          JSR       DECTPT                ;DECRIMENT TXTPTR.
000080                JSR       EVAL                  ;CALL MYSELF!
000081                BIT       VALTYP                ;REAL OR DOUBLE PRECISION?
000082                BVC       *+5
000083                JMP       BCDTSTR
000084                LDA       FACEXP
000085                BEQ       RLONE
000086                LDY       #0
000087                DFB       44
000088 RLONE:         LDY       #1
000089                JMP       SNGFLT
000090 EVAL3:         CMP       #'-'
000091                BEQ       DOMIN
000092                JMP       PARCHK
000093 PARC22         JSR       CHRGET                ;EAT THE ESCAPE TOKEN
000094                TAX                             ;KEEP THIS BYTE.
000095                JSR       CHRGET                ;POINT PAST IT.
000096                TXA                             ;GOT THE BYTE BACK.
000097                JMP       ISFUN
000098 PARC23         CMP       #FRETK
000099                BCC       SNERR6
000100                CMP       #POPTKN
000101                BCS       SNERR6
000102                ASL       A
000103                TAX
000104                JSR       CHRGET
000105                LDA       RESTBL-FRETK-FRETK+256,X
000106                STA       JMPER+1
000107                LDA       RESTBL-FRETK-FRETK+257,X
000108                STA       JMPER+2
000109                JMP       JMPER
000110 VPOS           BRK
000111                DFB       SDSTAT                ;CALL SOS FOR CONSOLE STATUS
000112                DW        REDCUR                ;SPECIFYING A READ OF THE CURSOR POSITION.
000113                LDY       CURY
000114                RTS
000115 DOERRLN        LDA       ERRLIN+1
000116                LDY       ERRLIN
000117                STA       FACHO
000118                STY       FACHO+1
000119                LDX       #$90
000120                SEC
000121                JMP       FLOATC
000122 GIVOUTREC      LDY       OUTREC
000123                JMP       SNGFLT
000124 GIVINDENT      LDY       INDENT
000125                JMP       SNGFLT
000126 GIVKBD         LDY       KEYSAVE               ;SEE WHAT HE TYPED TO GET THE INTERRUPT.
000127                JMP       SNGFLT
000128 GIVEOF         LDY       EOFSV
000129                DFB       44
000130 GIVERR         LDY       ERRNUM
000131                JMP       SNGFLT
000132 BCDTSTR        LDA       #>FAC                 ;RETURN WITH A,FLAGS = OR OF ALL FAC BYTES.
000133                LDY       #<FAC
000134                JSR       LORALL
000135                BEQ       ISFALS
000136                JMP       LONGST0
000137 ISFALS         JMP       LONGST1
000138 DOMIN:         JSR       EVAL                  ;EVALUATE THE CONSTANT.
000139                BIT       VALTYP                ;TYPE = FLOATING?
000140                BVC       CHGSGN                ;YES, OK.
000141                JMP       LTWSCOMP
000142 CHGSGN         JMP       NEGOP
000143 SNERR6         JMP       SNERR
000144 ;WE ENCOUNTERED AN EXPRESSION IN PARENTHESES.
000145 ;GO EVALUATE IT.  PUT A 0 ON VERBSTACK TO PREVENT
000146 ;PROBLEMS.
000147 ;AN UPPER-BOUND.
000148 DOPAR          LDA       #0
000149                LDY       VRBPT
000150                INY
000151                STA       VRBSTK,Y              ;LAST PRCEDENCE 0
000152                INY
```

```
000153                STY       VRBPT                   ;STCK PNTR INCRMNTD.
000154                CPY       #$20
000155                BCS       CMPLXERR
000156                JSR       FRMEVL1A                ;BUT DEC TXTPTR FIRST
000157                DEC       VRBPT
000158                DEC       VRBPT                   ;POINT BELOW 'TOP OF STACK' CHARATCER
000159                RTS
000160 PARCHK:         JSR       CHKOPN                  ;ONLY POSSIBILITY LEFT IS
000161 PARCHK2         JSR       DOPAR
000162 ;RECURSIVELY EVALUATE THE FORMULA.
000163 CHKCLS:         LDA       #41                     ;CHECK FOR A RIGHT PAREN
000164                DFB       44
000165 CHKSMC:         LDA       #$3B                    ;A SEMICOLON.
000166                DFB       44
000167 CHKEQL:         LDA       #'='
000168                DFB       44
000169 CHKPND:         LDA       #'#'
000170                DFB       44
000171 CHKOPN:         LDA       #40
000172                DFB       44
000173 CHKCOM:         LDA       #44
000174 SYNCHR:         LDY       #0
000175                CMP       (TXTPTR),Y              ;CHARACTERS EQUAL?
000176                BNE       SNERR
000177                JMP       CHRGET
000178 CMPLXERR        LDX       #ERRST
000179                JMP       ERROR
000180 SNERR:          LDX       #ERRSN                  ;'SYNTAX ERROR'
000181                JMP       ERROR
000182 ISVAR:          LDX       #$FF                    ;ENTRY TO PTRGET THAT DOESN'T
                                                                 CREATE UNKNOWN ARRAYS.
000183                JSR       PTREVL
000184                LDX       VARPNTB
000185 ISVRET2         STA       FACMO
000186                STY       FACMO+1
000187                STX       FACMOB
000188                BIT       VALTYP
000189                BVC       GOOO                    ;STRING IS SET UP.
000190                BPL       DBLVAR                  ;BCD VAR
000191                LDX       #0
000192                STX       FACOV
000193                RTS
000194 DBLVAR:         JMP       DMOVFM                  ;DOUBLE MOVE MEMORY INTO FAC.
000195 DECTPT:         TXA                               ;SAVE THE X REGISTER
000196                PHA
000197                LDA       TXTPTR
000198                BNE       DECTP1                  ;NO CARRY UNLESS ZERO.
000199                DEC       TXTPTR+1                ;HIGH BYTE DECRIMENTED.
000200                LDX       TXTPTRB
000201                LDY       TXTPTR+1
000202                JSR       FIXYX
000203                STX       TXTPTRB
000204                STY       TXTPTR+1
000205 DECTP1          DEC       TXTPTR                  ;LOW BYTE DECRIMENTED.
000206                PLA                               ;RESTORE THE X REGISTER
000207                TAX
000208                RTS
000209 GOOO:           EQU       *
000210                BIT       INTFLG
000211                BPL       GOOOOO
000212 GOOO2:          LDY       #0
000213                LDA       (FACMO),Y               ;FETCH HIGH.
000214                TAX
000215                INY
000216                LDA       (FACMO),Y
000217                TAY                               ;PUT LOW IN Y.
000218                TXA                               ;GET HIGH IN A.
000219                JMP       GIVAYF                  ;FLOAT AND RETURN.
000220 GOOOOO:         EQU       *
000221                JMP       MOVFM                   ;MOVE ACTUAL VALUE IN.
000222 ISFUN:          CMP       #ONEFUN
000223                BCS       GOODFUN
000224                CMP       #FNTK
000225                BNE       *+5
000226                JMP       FNDOER                  ;USER DEFINED FUNCTION.
000227                CMP       #NOTTK
000228                BNE       SNERR
000229                JMP       NOTDO
000230 GOODFUN         ASL       A
000231                PHA
```

```
000393 ;   A holds its length
000394 FRETMP          STA       INDEX
000395                 STX       INDEXB
000396                 STY       INDEX+1                 ;GET LENGTH FOR LATER.
000397                 JSR       FRETMS                  ;FREE UP THE TEMPORARY DESC.
000398 ;
000399 ; Procedure: NOTNOW
000400 ; On Entry:
000401 ;   INDEX holds a pointer to a string descriptor
000402 ; On Exit:
000403 ;   A holds length of the string
000404 ;   INDEX holds pointer to the actual string
000405 NOTNOW          PHP                               ;SAVE CODES.
000406                 LDY       #0                      ;PREP TO GET STUFF.
000407                 LDA       (INDEX),Y               ;GET COUNT AND
000408                 PHA                               ;SAVE IT.
000409                 INY
000410                 LDA       (INDEX),Y
000411                 TAX                               ;SAVE LOW ORDER.
000412                 INY
000413                 LDA       (INDEX),Y
000414                 LDY       #0
000415                 JSR       FIXYA
000416                 STY       INDEXB
000417                 STA       INDEX+1                 ;SAVE HIGH ORDER.
000418                 STX       INDEX
000419                 JSR       MAKREL
000420                 LDX       INDEX
000421                 LDY       INDEX+1
000422                 PLA
000423                 PLP
000424                 RTS
000425 FREFC1          PHP                               ;SAVE THE FREE STATUS
000426                 JSR       FREFAC                  ;GET THE DESCRIPTOR TO THE STRING IN QUESTION.
000427                 PLP                               ;Get the FREE STATUS back
000428 FRENOW          BNE       FRERTS                  ;EXIT IF NOT TIME TO FREE A TEMP.
000429 FRESPA          EQU       *                       ;Free up String Space
000430                 CMP       #0
000431                 BEQ       FRERTS                  ;DON'T FRE NULL STRINGS
000432                 JMP       TSTFRE
000433 FRETMS          PHA                               ;Save location of Temporary
000434                 LDA       #>TEMPST                ;Get TEMP starting point
000435                 CMP       TEMPPT                  ;Current pointer past start of TEMPs?
000436                 PLA
000437                 BCC       *+5                     ;YES
000438                 LDA       #$FF                    ;NO, CLEAR Z FLAG.
000439                 RTS
000440                 CPY       LASTPT+1                ;LAST ENTRY TO TEMP?
000441                 BNE       FRERTS
000442                 CMP       LASTPT
000443                 BNE       FRERTS
000444                 STA       TEMPPT
000445                 SBC       #STRSIZ                 ;POINT TO LAST ONE.
000446                 STA       LASTPT                  ;UPDATE TEMP PNTR.
000447                 LDY       #0                      ;ALSO CLEARS ZFLG SO WE DO REST OF FRETMP.
000448 FRERTS:         RTS       ALL                     DONE.
000449 NOTFAC          LDA       FACMO
000450                 LDY       FACMO+1
000451                 LDX       FACMOB
000452 NOTNW2          STA       INDEX                   ;ALT ENTRY FOR NOTNOW.
000453                 STY       INDEX+1
000454                 STX       INDEXB
000455                 JMP       NOTNOW
000456
000457 ; ##############################################################################################
000458 ; #   END OF FILE:  STRUTILS.TEXT
000459 ; #   LINES      :  450
000460 ; #   CHARACTERS :  18646
000461 ; ##############################################################################################

+------------------------------------------------------------------------------
|
|  THAT'S ALL FOLKS!      LINES: 461   CHARACTERS: 19200
|
+------------------------------------------------------------------------------
```

```
+------------------------------------------------------------------------------
|
|  File    : "B3MATHK.TEXT.PRETTY"
|  Created : Tuesday, December 30, 1997               5:14:29 PM
|  Modified: Wednesday, December 31, 1997             4:37:06 PM
|
+------------------------------------------------------------------------------

000001 ; ##############################################################################################
000002 ; #   PROJECT  :  Apple /// Business BASIC 1.3 (6502 Assembly Source Code)
000003 ; #   FILE NAME:  B3MATHK.TEXT
000004 ; ##############################################################################################
000005
000006                 SBTL      "FLOATING POINT MATH PACKAGE CONFIGURATION."
000007 ; FLOATING POINT NUMBER REPRESENTATION
000008 ;
000009 ; The floating point format is as follows:
000010 ;
000011 ; The exponent is stored in excess of 128, ie.  with a bias of 128, so, the
000012 ;   exponent is a signed 8-bit number with 128 added.  An exponent of ZERO means
000013 ;   that the number is zero.  The other bytes may not be assumed to be zero.
000014 ;
000015 ; The mantissa is 23 bits long.  The binary point is to the left of the MSB
000016 ;   of the mantissa.  The mantissa is positive with a 1 as a 24th bit assumed
000017 ;   to be between the the binary point and the MSB.
000018 ;
000019 ; The number in memory looks like this: 84 A0 00 00   (The number is -10.)
000020 ;
000021 ;      Exponent     Sign              Mantissa
000022 ;  ---------------------------------------------------------------------
000023 ; | 1 0 0 0 0 1 0 0 | 1 | 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
000024 ;  ---------------------------------------------------------------------
000025 ;       8 Bits      1 Bit             23 bits
000026 ;
000027 ; To evaluate a number like the one above, first evaluate the exponent.
000028 ;
000029 ;      1 0 0 0 0 1 0 0 = 132
000030 ;
000031 ; subtract the bias (128) to get the true exponent, in this case it is 4.
000032 ; Next evaluate the mantissa:
000033 ;
000034 ; . 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
000035 ;   ^
000036 ;   | This is the IMPLIED bit between the MSB and the binary point.
000037 ;
000038 ;                                                      -1    -3
000039 ; . 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 = 2   + 2
000040 ;                                                    =.5   + .125
000041 ;                                                    =.625
000042 ;
000043 ; Next, multiply the mantissa by 2 ^ exponent.
000044 ;
000045 ; .625 * 2 ^ 4 = .625 * 16 = 10
000046 ;
000047 ; Since the sign bit is a 1, the sign of the number is negative and the final
000048 ;   result is -10.
000049                 PAGE
000050 ; TO KEEP THE SAME NUMBER IN THE FAC WHILE SHIFTING,
000051 ;  TO SHIFT RIGHT, EXP:=EXP+1
000052 ;  TO SHIFT LEFT,  EXP:=EXP-1
000053 ; Arithmetic routine calling convention for 1 argument functions:
000054 ;    The argument is in the FAC (Floating point Accumulator).
000055 ;    The result is left in the FAC.
000056 ; Arithmetic routine calling convention for 2 argument functions:
000057 ;    The 1st argument is in ARG (ARGEXP, HO, MO, LO AND ARGSGN)
000058 ;    The 2nd argument is in the FAC.
000059 ;    The result is left in the FAC.
000060 ; THE 'T' ENTRY POINTS TO THE 2-ARGUMENT OPERATIONS HAVE B
000061 ; SETUP IN RESPECTIVE REGISTERS. BEFORE CALLING ARG MAY
000062 ; POPPED OFF THE STACK AND INTO ARG, FOR EXAMPLE.
000063 ; THE OTHER ENTRY POINTSSUMES Y,A POINTS TO THE ARGUMENT
000064 ; SOMEWHERE IN MEMORY. IT IS UNPACKED INTO ARG BY 'CONUPK'
000065 ; ON THE STACK, THE SGN IS PUSHED ON FIRST, THE LO,MO,HO &
000066 ; NOTE ALL THINGS ARE KEPT UNPACKED IN ARG, FAC & ON THE S
000067 ; IT IS ONLY WHEN SOMETHING IS STORED THAT IT IS PACKED
000068 ; BYTES. UNPACKED FORMAT HAS A SGN BYTE REFLECTING THE S
000069 ; NUMBER (POSITIVE=0, NEGATIVE=-1) A HO,MO & LO WITH THE H
000070 ; OF THE HO TURNED ON. THE EXP IS THE SAMES STORED FORMAT.
000071 ; THIS IS DONE FOR SPEED OF OPERATION.
000072                 PAGE
```

```
000073                  SBTL    "FLOATING POINT ADDITION AND SUBTRACTION."
000074 FADDH:           LDA     #>FHALF
000075                  LDY     #<FHALF                 ;ENTRY TO ADD 1/2.
000076                  JMP     FADD                    ;UNPACK AND GO ADD UIT.
000077 FSUB             LDX     #0
000078                  JSR     CONUPK                  ;UNPACK ARGUMENT INTO ARG.
000079 FSUBT:           LDA     FACSGN
000080                  EOR     #$FF                    ;COMPLEMENT IT.
000081                  STA     FACSGN
000082                  EOR     ARGSGN                  ;COMPLEMENT ARISGN.
000083                  STA     ARISGN
000084                  LDA     FACEXP                  ;Y=ARGEXP..
000085                  JMP     FADDT
000086 FADD5:           JSR     SHIFTR                  ;DO A LONG SHIF
000087                  BCC     FADD4                   ;CONTINUE WITH ADDITION.
000088 FADD             LDX     #0
000089                  JSR     CONUPK
000090 FADDT:           EQU     *
000091                  BNE     *+5
000092                  JMP     MOVFA                   ;IF FAC=0, RESULT IS IN ARG.
000093                  LDX     FACOV
000094                  STX     OLDOV
000095                  LDX     #ARGEXP                 ;DEFAULT IS SHIFT ARGUMENT.
000096                  LDA     ARGEXP                  ;IF ARG=0, FAC IS RESULT.
000097 FADDC:           TAY                             ;ALSO COPY ACCA INTO ACCY.
000098                  BNE     *+3                     ;RETURN IF ZERO
000099                  RTS
000100                  SEC
000101                  SBC     FACEXP
000102                  BEQ     FADD4                   ;NO SHIFTING.
000103                  BCC     FADDA                   ;BR IF ARGEXP.LT.FACEXP.
000104                  STY     FACEXP                  ;RESULTING EXPONENT.
000105                  LDY     ARGSGN                  ;SINCE ARG IS BIGGER, IT'S
000106                  STY     FACSGN                  ;SIGN IS SIGN OF RESU.
000107                  EOR     #$FF                    ;SHIFT A NEGATIVE NUMBER OF PLACES.
000108                  ADC     #0                      ;COMPLETE NEGATION. W/ C=1.
000109                  LDY     #0                      ;ZERO OLDOV.
000110                  STY     OLDOV
000111                  LDX     #FAC                    ;SHIFT THE FAC INSTEAD.
000112                  BNE     FADD1
000113 FADDA:           LDY     #0
000114                  STY     FACOV
000115 FADD1:           CMP     #$100-7                 ;FOR SPEED AND NECESSITY.  GETS
000116 ;MOST LIKELY CASE TO SHIFTR FASTEST
000117 ;AND ALLOWS SHIFTING OF NEG NUMS
000118 ;BY 'QINT'.
000119                  BMI     FADD5                   ;SHIFT BIG.
000120                  TAY
000121                  LDA     FACOV                   ;SET FACOV.
000122                  LSR     1,X                     ;GETS 0 IN MOST SIG BIT.
000123                  JSR     ROLSHF                  ;DO THE ROLLING.
000124 FADD4:           BIT     ARISGN                  ;GET SULTING SIGN.
000125                  BPL     FADD2                   ;IF POSITIVE, ADD.
000126 ;CARRY IS CLEAR.
000127                  LDY     #FACEXP
000128                  CPX     #ARGEXP                 ;FAC IS BIGGER.
000129                  BEQ     SUBIT
000130                  LDY     #ARGEXP                 ;ARG IS BIGGER.
000131 SUBIT:           SEC
000132                  EOR     #$FF
000133                  ADC     OLDOV
000134                  STA     FACOV
000135                  LDA     3+1,Y
000136                  SBC     3+1,X
000137                  STA     FACLO
000138                  LDA     2+1,Y
000139                  SBC     2+1,X
000140                  STA     FACMO
000141                  LDA     2,Y
000142                  SBC     2,X
000143                  STA     FACMOH
000144                  LDA     1,Y
000145                  SBC     1,X
000146                  STA     FACHO
000147 FADFLT:          BCS     NORMAL                  ;HERE IF SIGNS DIFFER. IF CARRY,
000148 ;FAC IS SET OK.
000149                  JSR     NEGFAC                  ;NEGATE FAC.
000150 NORMAL:          LDY     #0
000151                  TYA
000152                  CLC
```

```
000233                  STA       FACEXP
000234                  JSR       MULDIV                    ;FIX UP EXPONENTS.
000235                  INC       FACEXP                    ;SCALE IT RIGHT.
000236                  BEQ       GOOVER                    ;OVERFLOW.
000237                  LDX       #$100-3-1                 ;SETUP PROCEDURE.
000238                  LDA       #1
000239 DIVIDE:          EQU       *                         ;THIS IS THE BEST CODE IN THE WHOLE PILE
000240                  LDY       ARGHO                     ;SEE WHAT RELATION HOLDS.
000241                  CPY       FACHO
000242                  BNE       SAVQUO                    ;C=0,1. N(C=0)=0.
000243                  LDY       ARGMOH
000244                  CPY       FACMOH
000245                  BNE       SAVQUO
000246                  LDY       ARGMO
000247                  CPY       FACMO
000248                  BNE       SAVQUO
000249                  LDY       ARGLO
000250                  CPY       FACLO
000251 SAVQUO:          PHP
000252                  ROL       A                         ;SAVE RESULT.
000253                  BCC       QSHFT                     ;IF NOT DONE, CONTINUE.
000254                  INX
000255                  STA       RESLO,X
000256                  BEQ       LD100
000257                  BPL       DIVNRM                    ;NOTE THIS REQ 1 MO RAM THEN NECESS.
000258                  LDA       #1
000259 QSHFT:           PLP                                 ;RETURN CONDITION CODES.
000260                  BCS       DIVSUB                    ;FAC .LE. ARG.
000261 SHFARG:          ASL       ARGLO                     ;SHIFT ARG ONE PLACE LEFT.
000262                  ROL       ARGMO
000263                  ROL       ARGMOH
000264                  ROL       ARGHO
000265                  BCS       SAVQUO                    ;SAVE A RESULT OF ONE FOR THIS POSITION
000266 ;AND DIVIDE.
000267                  BMI       DIVIDE                    ;IF MSB ON, GO DECIDE WHETHER TO SUB.
000268                  BPL       SAVQUO
000269 DIVSUB:          TAY                                 ;NOTICE C MUST BE ON HERE.
000270                  LDA       ARGLO
000271                  SBC       FACLO
000272                  STA       ARGLO
000273                  LDA       ARGMO
000274                  SBC       FACMO
000275                  STA       ARGMO
000276                  LDA       ARGMOH
000277                  SBC       FACMOH
000278                  STA       ARGMOH
000279                  LDA       ARGHO
000280                  SBC       FACHO
000281                  STA       ARGHO
000282                  TYA
000283                  JMP       SHFARG
000284 LD100:           LDA       #$40                      ;ONLY WANT TWO MORE BITS.
000285                  BNE       QSHFT                     ;ALWAYS BRANCHES.
000286 DIVNRM:          EQU       *
000287                  ROR       A
000288                  ROR       A
000289                  ROR       A
000290                  AND       #$C0
000291 ;GET LAST TWO BITS INTO MSB AND B6.
000292                  STA       FACOV
000293                  PLP                                 ;TO GET GARBAGE OFF STACK.
000294                  JMP       MOVFR                     ;MOVE RESULT INTO FAC, THEN
000295 ;NORMALIZE RESULTND RETURN.
000296 DV0ERR:          LDX       #ERRDV0
000297                  JMP       ERROR
000298                  PAGE
000299                  SBTL      "FLOATING POINT MOVEMENT ROUTINES."
000300 ;MOVE RESULT TO FAC.
000301 MOVFR:           LDA       RESHO
000302                  STA       FACHO
000303                  LDA       RESMOH
000304                  STA       FACMOH
000305                  LDA       RESMO
000306                  STA       FACMO
000307                  LDA       RESLO                     ;MOVE LO AND SGN.
000308                  STA       FACLO
000309                  JMP       NORMAL                    ;ALL DONE.
000310 ;MOVE MEMORY INTO FAC (UNPACKED).
000311 MOVFM:           STA       INDEX1
000312                  STX       INDEXB
```

```
000153                ASL       A                       ;AND NOW BY TEN.
000154                CLC
000155                LDY       #0
000156                ADC       (TXTPTR),Y
000157                SEC
000158                SBC       #'0'
000159 MLEXMI:         STA       TENEXP                  ;SAVE RESULT.
000160                JMP       FINEC
000161                SBTL      "FLOATING POINT OUTPUT ROUTINE."
000162 N.0999:         DFB       $91                     ; 99999.9499
000163                DFB       $43
000164                DFB       $4F
000165                DFB       $F9
000166                DFB       $99
000167 N.9999:         DFB       $94                     ; 999999.499
000168                DFB       $74
000169                DFB       $23
000170                DFB       $F8
000171                DFB       $00
000172 N.MIL:          DFB       $94                     ; 10E6
000173                DFB       $74
000174                DFB       $24
000175                DFB       $00
000176                DFB       0
000177 ;ENTRY TO LINPRT.
000178 INPRT:          LDA       #>INTXT
000179                LDY       #<INTXT
000180                LDX       #INTXTB
000181                JSR       STROUTR
000182                LDA       CURLIN+1
000183                LDX       CURLIN
000184 LINPRT:         STA       FACHO
000185                STX       FACHO+1
000186                LDX       #$90                    ;EXPONENT OF 16.
000187                SEC                               ;NUMBER IS POSITIVE.
000188                JSR       FLOATC
000189                JSR       FOUT
000190                JMP       STROUTR                 ;PRINT AND RETURN.
000191 FOUT:           JSR       ROUNDER
000192                LDY       #1
000193 FOUTC:          EQU       *
000194                LDA       #$2D                    ;PRINT NOTHING IF POSITIVE,
000195                DEY                               ;NEG SIGN IF NEGATIVE
000196                BIT       FACSGN
000197                BPL       FOUT1.1
000198                INY
000199                STA       FBUFFR-1,Y              ;STORE THE CHARACTER.
000200 FOUT1.1:         STA       FACSGN                  ;MAKE FAC POS FOR QINT.
000201                STY       FBUFPT                  ;SAVE FOR LATER.
000202                INY
000203                LDA       #'0'                    ;GET ZERO TO TYPE IF FAC=0.
000204                LDX       FACEXP
000205                BNE       *+5
000206                JMP       FOUT19
000207                LDA       #0
000208                CPX       #$80                    ;IS NUMBER .LT. 1.0 ?
000209                BEQ       FOUT37                  ;NO.
000210                BCS       FOUT7
000211 FOUT37:         LDA       #>N.MIL
000212                LDY       #<N.MIL                 ;MULTIPLY BY 106.
000213                JSR       FMULT
000214                LDA       #$100-6-0
000215 FOUT7:          STA       DECCNT                  ;SAVE COUNT OR ZERO IT.
000216 FOUT4:          LDA       #>N.9999
000217                LDX       #N.MILB
000218                LDY       #<N.9999
000219                JSR       FCOMP                   ;IS NUMBER .GT. 999999.499 ?
000220 ;OR 999999999.499?
000221                BEQ       BIGGES
000222                BPL       FOUT9                   ;YES. MAKE IT SMALLER.
000223 FOUT3:          LDA       #>N.0999
000224                LDX       #N.MILB
000225                LDY       #<N.0999
000226                JSR       FCOMP                   ;IS NUMBER .GT. 99999.9499 ?
000227 ; OR 99999999.9499?
000228                BEQ       FOUT38
000229                BPL       FOUT5                   ;YES. DONE MULTIPLYING.
000230 FOUT38:         JSR       MUL10                   ;MAKE IT BIGGER.
000231                DEC       DECCNT
000232                BNE       FOUT3                   ;SEE IF THAT DOES IT.
```

```
000233 ;THIS ALWAYS GOES.
000234 FOUT9:          JSR       DIV10                ;MAKE IT SMALLER.
000235                 INC       DECCNT
000236                 BNE       FOUT4                ;SEE IF THAT DOES IT.
000237 ;THIS ALWAYS GOES.
000238 FOUT5:          JSR       FADDH                ;ADD A HALF TO ROUND UP.
000239 BIGGES:         JSR       QINT
000240                 LDX       #1                   ;DECIMAL POINT COUNT.
000241                 LDA       DECCNT
000242                 CLC
000243                 ADC       #0*1+7               ;SHOULD NUMBER BE PRINTED IN E NOTATION?
000244 ;IE, IS NUMBER .LT. .01 ?
000245                 STA       ISARA                ;FOR PRINT USING.
000246                 BMI       FOUTPI               ;YES.
000247                 CMP       #0+$8                ;IS IT .GT. 999999 (999999999)?
000248                 BCS       FOUT6                ;YES. USE E NOTATION.
000249                 ADC       #$100-1              ;NUMBER OF PLACESEFORE DECIMAL POINT.
000250                 TAX                            ;PUT INTO ACCX.
000251                 LDA       #2                   ;NO E NOTATION.
000252 FOUTPI:         SEC
000253 FOUT6:          SBC       #2                   ;EFFECTIVELY ADD 5 TO ORIG EXP.
000254                 STA       TENEXP               ;THAT IS THE EXPONENT TO PRINT.
000255                 STX       DECCNT               ;NUMBER OF DECIMAL PLACES.
000256                 TXA
000257                 BEQ       FOUT39
000258                 BPL       FOUT8                ;SOME PLACES BEFORE DEC PNT.
000259 FOUT39:         LDY       FBUFPT               ;GET POINTER TO OUTPUT.
000260                 LDA       #'.'                 ;PUT IN '.'
000261                 INY
000262                 STA       FBUFFR-1,Y
000263                 TXA
000264                 BEQ       FOUT16
000265                 LDA       #'0'                 ;GET THE ENSUING ZERO.
000266                 INY
000267                 STA       FBUFFR-1,Y
000268 FOUT16:         STY       FBUFPT               ;SAVE FOLATER.
000269 FOUT8:          LDY       #0
000270                 LDX       #$80                 ;FIRST PASS THRU, ACCX HAS MSB SET.
000271 FOUT2:          LDA       FACLO
000272                 CLC
000273                 ADC       FOUTBL+2+1,Y
000274                 STA       FACLO
000275                 LDA       FACMO
000276                 ADC       FOUTBL+1+1,Y
000277                 STA       FACMO
000278                 LDA       FACMOH
000279                 ADC       FOUTBL+1,Y
000280                 STA       FACMOH
000281                 LDA       FACHO
000282                 ADC       FOUTBL,Y
000283                 STA       FACHO
000284                 INX                            ;IT WAS DONE YET ANOTHER TIME.
000285                 BCS       FOUT41
000286                 BPL       FOUT2
000287                 BMI       FOUT40
000288 FOUT41:         BMI       FOUT2
000289 FOUT40:         TXA
000290                 BCC       FOUTYP               ;CAN USE ACCA AS IS.
000291                 EOR       #$FF                 ;FIND 11.-A.
000292                 ADC       #$A                  ;CARRY STILL ON TO COMPLETE NEGATION.
000293 ;AND WILL ALWAYS BE ON AFTER.
000294 FOUTYP:         ADC       #'0'-1               ;GET A CHARACTER TO PRINT.
000295                 INY
000296                 INY
000297                 INY
000298                 INY                            ;BUMP POINTER UP.
000299                 STY       FDECPT
000300                 LDY       FBUFPT
000301                 INY                            ;POINT TO PLACE TO STORE OUTPUT.
000302                 TAX
000303                 AND       #$7F                 ;GET RID OF MSB.
000304                 STA       FBUFFR-1,Y
000305                 DEC       DECCNT
000306                 BNE       STXBUF               ;NOT TIME FOR DP YET.
000307                 LDA       #'.'
000308                 INY
000309                 STA       FBUFFR-1,Y           ;STORE DP.
000310 STXBUF:         STY       FBUFPT               ;STORE PNTR FOR LATER.
000311                 LDY       FDECPT
000312                 TXA                            ;COMPLEMENT ACCX
```

```
000313                EOR       #$FF                    ;COMPLEMENT ACCA.
000314                AND       #$80                    ;SAVONLY MSB.
000315                TAX
000316                CPY       #FDCEND-FOUTBL
000317                BNE       FOUT2                   ;CONTINUE WITH OUTPUT.
000318                LDY       FBUFPT                  ;GET BACK OUTPUT PNTR.
000319 FOUT11:         LDA       FBUFFR-1,Y              ;REMOVE TRAILING ZEROES.
000320                DEY
000321                CMP       #'0'
000322                BEQ       FOUT11
000323                CMP       #'.'
000324                BEQ       FOUT12                  ;RUN INTO DP. STOP.
000325                INY                               ;SOMETHING ELSE. SAVE IT.
000326 FOUT12:         LDA       #$2B
000327                LDX       TENEXP
000328                BEQ       FOUT17                  ;NO EXPONENT TO OUTPUT.
000329                BPL       FOUT14
000330                LDA       #0
000331                SEC
000332                SBC       TENEXP
000333                TAX
000334                LDA       #$2D                    ;EXPONENT IS NEGATIVE.
000335 FOUT14:         STA       FBUFFR-1+2,Y            ;STORE SIGN OF EXP
000336                LDA       #'E'
000337                STA       FBUFFR-1+1,Y            ;STORE THE 'E' CHARACTER.
000338                TXA
000339                LDX       #'0'-1
000340                SEC
000341 FOUT15:         INX                               ;MOVE CLOSER TO OUTPUT VALUE.
000342                SBC       #$A                     ;SUBTRACT 10.
000343                BCS       FOUT15                  ;NOT NEGATIVE YET.
000344                ADC       #'0'+$A                 ;GET SECOND OUTPUT CHARACTER.
000345                STA       FBUFFR-1+4,Y            ;STORE HIGH DIGIT.
000346                TXA
000347                STA       FBUFFR-1+3,Y            ;STORE  LOW DIGIT.
000348                LDA       #0                      ;PUT IN TERMINATOR.
000349                STA       FBUFFR-1+5,Y
000350                BEQ       FOUT20                  ;RETURN. (ALWAYS BRANCHES).
000351 FOUT19:         STA       FBUFFR-1,Y              ;STORE THE CHARACTER.
000352 FOUT17:         LDA       #0                      ;A TERMINATOR.
000353                STA       FBUFFR-1+1,Y
000354 FOUT20:         LDA       #>FBUFFR
000355                LDX       #0
000356                LDY       #<FBUFFR
000357                RTS                               ;ALL DONE.
000358 FHALF:          DFB       $80                     ;1/2
000359 ZERO:           DFB       000
000360                DFB       000
000361                DFB       000
000362                DFB       0
000363                DFB       0,0,0,0                 ;PTRGET POINTS TO ZERO WHEN IT DOESN'T CREATE.
000364 ;POWER OF TEN TABLE
000365 FOUTBL:         EQU       *
000366                DFB       $FF,$FE,$79,$60         ;-100,000
000367                DFB       $0,0,$27,$10            ;10,000
000368                DFB       $FF,$FF,$FC,$18         ;-1,000
000369                DFB       0,0,0,$64               ;100
000370                DFB       $FF,$FF,$FF,$F6         ;-10
000371                DFB       0,0,0,1                 ;1
000372 FDCEND:         EQU       *
000373                SBTL      "EXPONENTIATION AND SQUARE ROOT FUNCTION."
000374 ;SQUARE ROOT FUNCTION --- SQR(A)
000375 ;USE SQR(X)=X.5
000376 SQR:            JSR       MOVAF                   ;MOVE FAC INTO ARG.
000377                LDA       #>FHALF
000378                LDX       #FHALFB
000379                LDY       #<FHALF
000380                JSR       MOVFM                   ;PUT MEMORY INTO FAC.
000381 ;LAST THING FETCHED IS FACEXP. INTO ACCX.
000382 ; JMP FPWRT   ;FALL INTO FPW.
000383 ;EXPONENTIATION ---  XY.
000384 ;N.B.  00=1
000385 ;FIRST CHECK IF Y=0. IF SO, THE RESULT IS 1.
000386 ;NEXT CHECK IF X=0. IF SO THE RESULT IS 0.
000387 ;THEN CHECK IF X.GT.0. IF NOT CHECK THAT Y IS AN INTEGER.
000388 ;IF SO, NEGATE X, SO THAT LOG DOESN'T GIVE FCERR.
000389 ;IF X IS NEGATIVE AND Y IS ODD, NEGATE THE RESULT
000390 ;RETURNED BY EXP.
000391 ;TO COMPUTE THE RESULT USE XY=EXP((Y*LOG(X)).
000392 FPWRT:          BEQ       GOTOEXP                 ;IF FAC=0, JUST EXPONENTIATE THAT.
```

```
000313                  DFB       $DF
000314                  DFB       $E1
000315                  DFB       $86                     ; -41.34170209
000316                  DFB       $A5
000317                  DFB       $5D
000318                  DFB       $E7
000319                  DFB       $28
000320                  DFB       $83                     ; 6.2831853070
000321                  DFB       $49
000322                  DFB       $0F
000323                  DFB       $DA
000324                  DFB       $A2
000325                  DFB       $A6
000326                  DFB       $D3
000327                  DFB       $C1
000328                  DFB       $C8
000329                  DFB       $D4
000330                  DFB       $C8
000331                  DFB       $D5
000332                  DFB       $C4
000333                  DFB       $CE
000334                  DFB       $CA
000335                  PAGE
000336                  SBTL      "ARCTANGENT FUNCTION."
000337 ;USE IDENTITIES TO GET ARG BETWEEN 0 AND 1 AND THEN USE AN
000338 ;APPROXIMATION POLYNOMIAL TO COMPUTE ARCTAN(X).
000339 ATN:              LDA       FACSGN                  ;WHAT IS SIGN?
000340                  PHA                               ;(MEANWHILE SAVE FOR LATER.)
000341                  BPL       ATN1
000342                  JSR       NEGOP                   ;IF NEGATIVE, NEGATE FAC.
000343 ;USE ARCTAN(X)=-ARCTAN(-X) .
000344 ATN1:             LDA       FACEXP
000345                  PHA                               ;SAVE THIS TOO FOR LATER.
000346                  CMP       #$81                    ;SEE IF FAC .GE. 1.0 .
000347                  BCC       ATN2                    ;IT IS LESS THAN 1.
000348                  LDA       #>FONE
000349                  LDY       #<FONE                  ;GET PNTR TO 1.0 .
000350                  JSR       FDIV                    ;COMPUTE RECROCAL.
000351 ;USE ARCTAN(X)=PI/2-ARCTAN(1/X) .
000352 ATN2:             LDA       #>ATNCON
000353                  LDY       #<ATNCON                ;PNTR TO ARCTAN CONSTANTS.
000354                  JSR       POLYX
000355                  PLA
000356                  CMP       #$81                    ;WAS ORIGINAL ARGUMENT .LT. 1 ?
000357                  BCC       ATN3                    ;YES.
000358                  LDA       #>PI2
000359                  LDY       #<PI2
000360                  JSR       FSUB                    ;SUBTRACT ARCTAGN FROM PI/2.
000361 ATN3:             PLA                               ;WAS ORIGINAL ARGUMENT POSITIVE?
000362                  BPL       ATN4                    ;YES.
000363                  JMP       NEGOP                   ;IF NEGATIVE, NEGATE RESULT.
000364 ATN4:             RTS                               ;ALL DONE.
000365 ATNCON:           DFB       $0B                     ;DEGREE-1.
000366                  DFB       $76                     ; -.0006847939119
000367                  DFB       $B3
000368                  DFB       $83
000369                  DFB       $BD
000370                  DFB       $D3
000371                  DFB       $79                     ; .004850942156
000372                  DFB       $1E
000373                  DFB       $F4
000374                  DFB       $A6
000375                  DFB       $F5
000376                  DFB       $7B                     ; -.01611170184
000377                  DFB       $83
000378                  DFB       $FC
000379                  DFB       $B0
000380                  DFB       $10
000381                  DFB       $7C                     ; .03420963805
000382                  DFB       $0C
000383                  DFB       $1F
000384                  DFB       $67
000385                  DFB       $CA
000386                  DFB       $7C                     ; -.05427913276
000387                  DFB       $DE
000388                  DFB       $53
000389                  DFB       $CB
000390                  DFB       $C1
000391                  DFB       $7D                     ; .07245719654
000392                  DFB       $14
```

```
000393                DFB       $64
000394                DFB       $70
000395                DFB       $4C
000396                DFB       $7D                       ; -.08980239538
000397                DFB       $B7
000398                DFB       $EA
000399                DFB       $51
000400                DFB       $7A
000401                DFB       $7D                       ; .1109324134
000402                DFB       $63
000403                DFB       $30
000404                DFB       $88
000405                DFB       $7E
000406                DFB       $7E                       ; -.1428398077
000407                DFB       $92
000408                DFB       $44
000409                DFB       $99
000410                DFB       $3A
000411                DFB       $7E                       ; .1999991205
000412                DFB       $4C
000413                DFB       $CC
000414                DFB       $91
000415                DFB       $C7
000416                DFB       $7F                       ; -.3333333157
000417                DFB       $AA
000418                DFB       $AA
000419                DFB       $AA
000420                DFB       $13
000421                DFB       $81                       ; 1.0
000422                DFB       $00
000423                DFB       $00
000424                DFB       $00
000425                DFB       $00
000426
000427 ; ###############################################################################################
000428 ; #   END OF FILE:  B3EXPON.TEXT
000429 ; #   LINES      :  420
000430 ; #   CHARACTERS :  17948
000431 ; ###############################################################################################

+-------------------------------------------------------------------------------
|
|  THAT'S ALL FOLKS!      LINES: 431   CHARACTERS: 18500
|
+-------------------------------------------------------------------------------
```

```
+------------------------------------------------------------------------------
|
|  File    : "B3FREER.TEXT.PRETTY"
|  Created : Tuesday, December 30, 1997               5:14:26 PM
|  Modified: Wednesday, December 31, 1997             4:37:03 PM
|
+------------------------------------------------------------------------------

000001 ; ###########################################################################################
000002 ; #   PROJECT  :  Apple /// Business BASIC 1.3 (6502 Assembly Source Code)
000003 ; #   FILE NAME:  B3FREER.TEXT
000004 ; ###########################################################################################
000005
000006                 SBTL      "General Pointer Maintenance"
000007 TSTFRE          LDX       INDEX
000008                 LDY       INDEX+1
000009                 STX       GRBTOP
000010                 STY       GRBTOP+1
000011                 LDX       INDEXB
000012                 STX       GRBTOPB
000013 *********************************************
000014 *   ENTRY CONDITIONS                        *
000015 *     A: LENGTH OF NEW ATOM (LOW)           *
000016 *     X: LENGTH OF NEW ATOM (HIGH)          *
000017 *     Y: DON'T CARE                         *
000018 *     GRBTOP: POINTER TO ATOM TO BE ADDED   *
000019 *                                           *
000020 *   EXIT CONDITIONS                         *
000021 *     A: UNKNOWN                            *
000022 *     X: UNKNOWN                            *
000023 *     Y: UNKNOWN                            *
000024 *                                           *
000025 *********************************************
000026 ;FRESML    EQU   *             ;This used to be here but not referenced
000027                 STA       TEMP
000028                 CLC
000029                 ADC       GRBTOP
000030                 STA       HEADER                    ;HEADER WILL POINT TO STRING INFO SPACE.
000031                 LDA       #0
000032                 ADC       GRBTOP+1
000033                 LDY       GRBTOPB
000034                 JSR       FIXADC
000035                 STA       HEADER+1
000036                 STY       HEADERB
000037                 LDA       TEMP                      ;STUFF LENGTH INTO STRING INFO AREA.
000038                 LDY       #2
000039 FREIT1          STA       (HEADER),Y
000040                 LDA       #0                        ;FOR THE OTHER TWO BYTES.
000041                 DEY
000042                 BPL       FREIT1
000043                 RTS
000044 FIXSBC          BCS       *+4                       ;THIS ROUTINE USED BY PEOPLE SUBTRACTING
000045                 DEY                                 ;MEMORY POINTERS.
000046                 DEY
000047 FIXSB2          CMP       #MINPG                    ;BYTE 2 IS ALWAYS KEEPED IN THE RANGE
000048                 BCS       *+5                       ;MINPG THROUGH MAXPG (2 -- $82)
000049                 ADC       #MAXPG-MINPG              ;SO THAT THE ON-THE-FLY BANK SWITCHING WILL WORK O.K.
000050                 DEY
000051                 CMP       #MAXPG
000052                 BCC       *+5
000053                 SBC       #MAXPG-MINPG
000054                 INY
000055                 SEC
000056                 RTS
000057 FIXADC          BCC       *+4                       ;JUST LIKE FIXSBC ONLY USED FOR ADD OPERATIONS.
000058                 INY
000059                 INY
000060                 JSR       FIXSB2
000061                 CLC
000062                 RTS
000063 FIXYAX          EQU       *                         ;THIS ROUTINE ALLOWS MIXED SUBTRACTION,
000064                 PHP                                 ;SO THAT A REGULAR 16 BIT QUANTITY CAN
                                                                        BE SUBTACTED
000065                 PHA                                 ;FROM A BANK.PAGE.BYTE (MEMORY) POINTER.
000066                 TXA                                 ;Y,A = BANK,PAGE POINTER VALUES.
000067                 ASL       A                         ;X IS PACKED PAGE COUNT TO SUBTRACT.
000068                 BCC       *+3                       ;ON RETURN A IS RESULT OF SBC,
000069                 DEY                                 ;AND Y IS ADJUSTED PROPERLY.
000070                 LSR       A
000071                 STA       CHARAC
```

```
000393                BPL       LREM2
000394                STA       PTR3                ;PTR3=ARG
000395                JMP       LRESDV              ;GOT THE RESULT.
000396 LORALL          STA       INDEX               ;RETURN A ZERO IFF (A.Y) IS ALL ZERO.
000397                STX       INDEXB
000398                STY       INDEX+1
000399                LDA       #0
000400                LDY       #7
000401 LORAL2          ORA       (INDEX),Y
000402                BNE       LGOTOR
000403                DEY
000404                BPL       LORAL2
000405                TAY                           ;SET Z FLAG.
000406 LGOTOR          RTS
000407 LUNPACK         EQU       *                   ;PUTS UNPACKBCD(FAC) IN NUMSTR.
000408 *SETS HIGH BIT OF FACSGNN IF NEGATIVE.
000409                LDA       FACEXP
000410                CMP       #$80
000411                LDA       #20
000412                ROL       A                   ;HIGH BIT OF FACEXP NOW LOW BIT OF A.
000413                STA       ISARA               ;NOW LOOKS LIKE BCD EXPONENT.
000414                LDA       FAC                 ;HIGH BYTE.
000415                PHA
000416                BPL       *+5                 ;DO TWOSCOMP IF NEGATIVE.
000417                JSR       LTWSCOMP            ;TWOS COMP OF FAC INTO FAC.
000418                LDA       #>FAC
000419                LDX       #0
000420                LDY       #<FAC
000421                JSR       LORALL
000422                BEQ       *+5
000423                JSR       LNRMFAC             ;NORMALIZE THE BEAST.
000424                LDA       #64                 ;MAXIMUM SHIFTT.
000425                SEC
000426                SBC       KIMY                ;# OF LEFT SHIFTS DONE BY LNRMFAC.
000427                STA       KIMY
000428                JSR       LZIPARG             ;ZERO OUT ARG.
000429 LCONVBCD        LDA       #>FAC
000430                JSR       LSHFTL              ;CARRY = HIGH BIT.
000431                LDX       #10
000432                SED                           ;DECIMAL MODE!
000433 LADDBCD         LDA       ARG-1,X             ;DOUBLE ARG AND ADD IN CARRY.
000434                ADC       ARG-1,X
000435                STA       ARG-1,X
000436                DEX
000437                BNE       LADDBCD
000438                CLD                           ;BETTER CLEAR THAT SUCCER!
000439                DEC       KIMY                ;DONE ONE BIT, ALL DONE?
000440                BNE       LCONVBCD
000441                LDX       #10
000442 LABCD2          LDA       ARG-1,X             ;MOVE THE RESULT INTO FACT.
000443                STA       FACT-1,X            ;SO UUNPACK WILL WORK CORRECTLY.
000444                DEX
000445                BNE       LABCD2
000446                JSR       UUNPACK
000447                PLA
000448                STA       FACSGN              ;HIGH BIT ON IF # WAS NEG.
000449                RTS
000450 LZIPARG         LDA       #0                  ;ZERO OUT 10 BYTES OF ARG.
000451                LDX       #10                 ;ARG.....ARG+9 (REALLY RES+1).
000452 LZIPAR2         STA       ARG-1,X
000453                DEX
000454                BNE       LZIPAR2
000455                RTS
000456 LOUT            EQU       *                   ;LONG INT. OUTPUT ROUTINE.
000457                JSR       LUNPACK             ;UNPACK TO BCD IN NUMSTR.
000458                LDX       #0
000459                LDA       #'0'
000460                STA       NUMSTR              ;AT LEAST ONE NUMBER.
000461                LSR       ISARA               ;NOW HAS # OF DIGITS.
000462                LDY       #1
000463 LLOOKLUP        INY
000464                DEC       ISARA               ;DONE LAST DIGIT?
000465                BMI       LISZERO             ;YES, ALL DONE.
000466                LDA       NUMSTR,Y            ;GET A BCD BYTE.
000467                CPX       #0
000468                BEQ       LOUT3
000469 LOUT5           ORA       #'0'
000470                STA       NUMSTR,X
000471                INX
000472                BNE       LLOOKLUP
```

```
000473 LOUT3           STA       KIMY
000474                 BIT       FACSGN
000475                 BPL       LLOKLUP
000476                 LDA       #'-'
000477                 STA       NUMSTR,X
000478                 INX
000479 LLOKLUP         LDA       KIMY
000480                 BNE       LOUT5                ;ALWAYS.
000481 LISZERO         CPX       #1                   ;AT LEAST ONE BYTE TO BE OUTPUT.
000482                 BCS       *+4
000483                 LDX       #1
000484                 STX       LENUM
000485                 LDA       #0
000486                 STA       NUMSTR,X
000487                 RTS                            ;ALWAYS END WITH A NULL.
000488 LONGST0         LDA       #0                   ;STORE ZERO IN FAC.
000489                 LDX       #7
000490 LONGST          STA       FAC,X
000491                 DEX
000492                 BPL       LONGST
000493                 RTS
000494 LONGST1         EQU       *                    ;STORE 1 IN FAC.
000495                 JSR       LONGST0
000496                 LDA       #1
000497                 STA       FAC+7
000498                 RTS
000499 DMOVFM          EQU       *
000500 LDFACT          STA       INDEX                ;STORE (A.Y) IN FAC.
000501                 STX       INDEXB
000502                 STY       INDEX+1
000503                 LDY       #7
000504 LDFAC2          LDA       (INDEX),Y
000505                 STA       FAC,Y
000506                 DEY
000507                 BPL       LDFAC2
000508                 RTS
000509 CONV2FLT        BIT       VALTYP               ;CONVERT TO FLOAT.
000510                 BMI       STR2FLT
000511                 BVC       CONV2RTS
000512 * CONVERT FROM LONG INTEGER TO FLOAT.
000513 LMAKFLT         LDA       FAC
000514                 PHA
000515                 BPL       *+5
000516                 JSR       LTWSCOMP             ;CONVERT TO PLUS.
000517                 LDA       #>FAC
000518                 LDX       #0
000519                 LDY       #<FAC                ;IN THIS CASE 0.
000520                 JSR       LORALL               ;WAS ALL OF FAC ZERO?
000521                 BNE       *+6                  ;YES, GIVE HIM ZERO.
000522                 PLA                            ;CLEAN UP STACK FOR DONN
000523                 JMP       GIVE0
000524                 JSR       LNRMFAC              ;BIT 6 OF FAC NOW ON.
000525                 JSR       LSHFTL               ;BIT 7 ON, KIMY=# OF LEFT SHIFTS.
000526                 LDA       FAC+4
000527                 STA       FACOV                ;MIGHT AS WELL PUT THE BITS IN.
000528                 LDA       FAC+3
000529                 STA       FACLO
000530                 LDA       FAC+2
000531                 STA       FACMO
000532                 LDA       FAC+1
000533                 STA       FACMOH
000534                 LDA       FAC
000535                 STA       FACHO
000536                 LDA       #$80+$3F             ;MAX EXPONENT WE COULD GET.
000537                 SEC
000538                 SBC       KIMY
000539                 STA       FACEXP               ;NOW HAS THE CORRECT EXPONENT.
000540                 PLA
000541                 STA       FACSGN
000542                 LDA       #0
000543                 STA       VALTYP               ;RESULT IS ZERO.
000544 CONV2RTS        RTS                            ;ALL DONE.
000545 STR2FLT         JMP       VAL                  ;THAT WAS EASY!
000546 CONV2LNG        EQU       *                    ;CONVERT FAC TO LONG INTEGER (ROUNDS).
000547                 BIT       VALTYP               ;WHAT DO WE NEED TO CONVERT?
000548                 BMI       STR2LNG              ;STRING!
000549                 BVS       CONV2RT2             ;STARTED OUT LONG!
000550 * CONVERT FROM FLOATING POINT TO LONG INT.
000551                 JSR       QINTRN               ;ADD .5 AND TRUNCATE.
000552                 JSR       LZIPARG              ;ARG STARTS AT ZERO.
```

```
000553                LDA       FACSGN
000554                PHA
000555                LDA       FACEXP
000556                SEC
000557                SBC       #$80                ;GET # BITS TO SHIFT.
000558                BMI       GTFACNA
000559                STA       YSAVE
000560                CMP       #$40                ;CAN WE REALLY FIT IT IN?
000561                BCC       *+5                 ;YES
000562                JMP       OVERR               ;NO --- OVERFLOW.
000563                LDA       #>ARG
000564 PASSABIT        JSR       ASLFAC
000565                JSR       LSHFTLC             ;SHIFT ARG LEFT WITH CARRY.
000566                DEC       YSAVE
000567                BNE       PASSABIT
000568 GTFACNA         LDA       #$40
000569                STA       VALTYP              ;RESULT IS LONG INT.
000570                LDA       #>ARG
000571                LDX       #0
000572                JSR       LDFACT              ;MOV ARG TO FAC.
000573                PLA
000574                BPL       *+5
000575                JSR       LTWSCOMP
000576                RTS
000577 ASLFAC          EQU       *                   ;SHIFTS FLOATING FAC LEFT BY ONE.
000578                ASL       FACOV
000579                ROL       FACLO
000580                ROL       FACMO
000581                ROL       FACMOH
000582                ROL       FACHO
000583 CONV2RT2        RTS
000584 STR2LNG         LDA       #>LINP              ;WASN'T THAT EASY
000585                LDY       #<LINP
000586                JMP       VALSTR              ;JUST LIKE THE VAL FUNCTION.  ALMOST.
000587 CONV2INT        JSR       CONV2FLT            ;JUST LIKE CONVERTING TO FLOAT.
000588 QINTRN          JSR       FADDH               ;WITH A ROUND ON THE END.
000589                JMP       INT
000590 CONV2STR        EQU       *                   ;CONVERT THE FAC TO A STRING.
000591                BIT       VALTYP              ;WHAT WAS THE BEAST?
000592                BMI       CONV2RT2            ;STRING!
000593                BVS       CONWASL             ;LONG!
000594                JMP       STRS                ;SAME AS STR$ IN THIS CASE.
000595 CONWASL         JSR       LOUT                ;OUTPUT THE # INTTO THE BUFFER.
000596                LDA       #>NUMSTR
000597                LDX       #NUMSTRB
000598                LDY       #<NUMSTR
000599                JMP       STRLIT              ;MAKE THIS SUCCER A STRING.
000600 LAND            LDA       #>FAC               ;LOGICAL AND FOR LONG INT.
000601                LDX       #0
000602                LDY       #<FAC
000603                JSR       LORALL              ;WAS FAC ZERO?
000604                BNE       *+3
000605                RTS
000606                LDA       #>ARG
000607                LDX       #0
000608                LDY       #<ARG
000609                JSR       LORALL
000610 LGIVM1          BEQ       *+5
000611                JMP       LONGST1
000612 LGIVM0          JMP       LONGST0
000613 LONGOR          LDA       #>FAC
000614                LDY       #<FAC               ;IN THIS CASE ZERO.
000615                JSR       LORALL
000616                STA       KIMY                ;0 IFF FAC WAS ALL ZERO.
000617                LDA       #>ARG               ;Y STILL 0="<ARG".
000618                LDY       #<ARG
000619                LDX       #0
000620                JSR       LORALL              ;ARG ZERO?
000621                ORA       KIMY
000622                BEQ       LGIVM0
000623                JMP       LONGST1             ;RESULT 1
000624 LDOCOMP         EQU       *                   ;RETURNS WITH 1 OR ZERO BASED ON COMPARE.
000625                LDA       ARG
000626                CMP       FAC
000627                BMI       LISL
000628                BNE       LISG                ;SIGNED NUMBERS
000629                LDX       #1
000630                JSR       LCMP2               ;COMPARE THE REST
000631                JSR       LCOMP               ;LDA ARG, CMP FAC.
000632                BEQ       LISEQ
```

```
000073                STY       VARNAM
000074                PHA
000075                LDA       TXTPTR
000076                SEC
000077                SBC       LOWDS
000078                CMP       #65
000079                BCS       WRDERR                ;BLOW UP. NAME TOO BIG!!
000080                LDY       VALTB2,X
000081                BNE       NTREAL                ;IS NOT A REAL VARIABLE.
000082                ADC       #$1                   ;C IS CLEAR.
000083 NTREAL         STA       FOUR6                 ;NAME LENGTH [+1].
000084                PLA
000085                BIT       SUBFLG
000086                BPL       YSTORE
000087                BVC       YSTORE
000088                LDY       #6
000089                STY       VARNAM                ;LENGTH FOR FN DEF. IS 6.
000090                LDY       #$10                  ;VALTYP FOR FUNCTIONS=$10 (SPECIAL CASE)
000091 YSTORE         STY       VALTYP                ;TYPE BYTE VALUE FROM THE TABLE.
000092                SEC
000093                ORA       SUBFLG                ;ALLOWS ARRAYS ONLY TO GET
000094                SBC       #'('                  ;TO ISARY.
000095                BNE       SPLVAR
000096 GARRAY         JMP       ISARY                 ;WE HAVE AN ARRAY HERE!!
000097 SPLVAR         BIT       SUBFLG
000098                BMI       SIMVAR
000099                BVS       GARRAY                ;GET AN ARRAY (FOR STORE OR RECALL).
000100 SIMVAR         CPY       #$C0                  ;IS IT FILE BUFF TYPE?
000101                BEQ       WRDERR                ;BLOW UP IF SO. FILBUFFS ARE ARRAYS ONLY
000102                LDY       #$0
000103                STY       SUBFLG                ;ALLOWS SUBSCRIPTS AGAIN.
000104                LDA       SMVARS                ;GET THE POINTER TO THE SIMPLE VARIABLES.
000105                LDX       SMVARS+1
000106                LDY       SMVARSB
000107                STY       SRCHPTB
000108 STXLOP         STX       SRCHPT+1              ;SAVE THE CURRENT WORKING POINTER.
000109 SRCHLP         STA       SRCHPT
000110                CMP       STREND                ;CHECK AGAINST END OF VARIABLES STORAGE.
000111                BNE       PNTER3
000112                CPX       STREND+1
000113                BNE       PNTER3
000114                LDY       SRCHPTB
000115                CPY       STRENDB
000116                BEQ       ADDVAR                ;VAR DOESN'T EXIST, SO CREATE IT!!
000117 PNTER3         LDY       #0
000118                LDA       (SRCHPT),Y            ;CURRENT ENTRY LENGTH.
000119                STA       VARNAM+1              ;SAVE IT IN HERE.
000120 COMPAR         LDA       (LOWDS),Y             ;VARIABLE NAME FROM TEXT.
000121                INY
000122                CMP       #'.'
000123                BCC       TYPCHK
000124                CMP       #'A'+$20              ;CHECK FOR LOWER CASE
000125                BCC       NOT2BIG
000126                CMP       #'Z'+1+$20
000127                BCS       TYPCHK
000128                SBC       #$1F                  ;CARRY CLEAR. NOW UPPER CASE.
000129 NOT2BIG        CMP       (SRCHPT),Y
000130                BEQ       COMPAR                ;AS LONG AS THEY MATCH.
000131 TYPCHK         LDA       VALTYP
000132                CMP       (SRCHPT),Y
000133                BNE       GONEXT                ;NOT THIS GUY, GO FURTHER.
000134                CPY       FOUR6                 ;CORRECT LENGTH?
000135                BNE       GONEXT                ;NOT THIS GUY, SORRY.
000136                TYA                             ;CARRY IS SET.
000137 FINUP          ADC       SRCHPT                ;FINAL CALCULATION TO
000138                BCC       CMPLTE                ;POINT TO THE VARIABLE
000139                INX
000140                CPX       #MAXPG
000141                BCC       *+7
000142                LDX       #MINPG
000143                INC       SRCHPTB
000144 CMPLTE         STX       VARPNT+1
000145                LDY       VARPNT+1
000146                STA       VARPNT
000147                LDX       SRCHPTB
000148                STX       VARPNTB
000149 MYMYMY         LDX       VALTYP
000150                CPX       #$80                  ;INTEGER TYPE?
000151                BNE       PTRRTS                ;NO, GOOD.
000152                STX       INTFLG                ;SET INTER FLAG.
```

```
000153                  LDX       #$0                     ;YES, CLEAR VALTYP.
000154                  STX       VALTYP
000155 PTRRTS           EQU       *
000156                  RTS
000157 GONEXT           CLC
000158                  LDA       VARNAM+1                ;THIS CODE FIXES
000159                  ADC       SRCHPT                  ;THE SEARCH POINTER
000160                  BCC       SRCHLP                  ;TO LOOK AT THE NEXT
000161                  INX                               ;VARIABLE NAME
000162                  CPX       #MAXPG
000163                  BCC       STXLOP
000164                  LDX       #MINPG
000165                  INC       SRCHPTB
000166                  BCS       STXLOP                  ;ALWAYS TAKEN.
000167 ; TEST FOR A LETTER. / CARRY OFF= NOT A LETTER.
000168 ;     CARRY ON= A LETTER.
000169 ISLETC:          CMP       #'A'
000170                  BCC       ISLRTS                  ;IF LESS THAN 'A', RET.
000171                  SBC       #'Z'+1
000172                  SEC
000173                  SBC       #$100-'Z'-1             ;RESET CARRY IF A .GT. 'Z'.
000174                  BCS       ISLRTS
000175                  CMP       #'A'+$20
000176                  BCC       ISLRTS
000177                  SBC       #'Z'+$21
000178                  SEC
000179                  SBC       #$100-'Z'-$21
000180 ISLRTS:          RTS                               ;RETURN TO CALLER.
000181 ;
000182 ; A looked for variable does not exist in the space.  Add it to
000183 ;   the variable tables.
000184 ADDVAR           SEC
000185                  LDA       FOUR6                   ;FETCH NAME LENGTH +1
000186                  STA       VARNAM+1                ;SAVE IT.
000187                  ADC       VARNAM                  ;C IS SET. ADD [# BYTES FOR TYPE] +1.
000188                  LDY       STREND+1
000189                  LDX       STRENDB
000190                  PHA                               ;TOTAL BYTES THIS ENTRY.
000191                  ADC       STREND                  ;CARRY IS CLEAR.
000192                  BCC       GOARND
000193                  INY                               ;STREND is crossing page bounds
000194                  CPY       #MAXPG                  ;Page <82?
000195                  BCC       GOARND
000196                  LDY       #MINPG                  ;NO! Page wraps to 2 and
000197                  INX                               ;  kick bank indicator up 1.
000198 GOARND           JSR       REASON                  ;FIND OUT IF THERE IS ENOUGH ROOM.
000199                  STY       STREND+1
000200                  STX       STRENDB
000201                  STA       STREND                  ;SAVE THE NEW 'END OF STORAGE' POINTER.
000202                  PLA                               ;RETRIEVE THE TOTAL ENTRY LENGTH.
000203                  LDY       #$0
000204                  STA       (SRCHPT),Y              ;ENTRY LENGTH.
000205                  TAY
000206                  DEY                               ;POINT TO THE ACTUAL LAST BYTE OF SPACE.
000207                  LDX       VARNAM                  ;TYPE LENGTH
000208                  LDA       #$0
000209 TILOOP           STA       (SRCHPT),Y              ;LOOP TO SET VARIABLE TO ZERO.
000210                  DEY
000211                  DEX                               ;TYPE LENGTH = TYPE LENGTH - 1.
000212                  BNE       TILOOP
000213                  LDA       VALTYP
000214                  STA       (SRCHPT),Y              ;STORE THE TYPE BYTE.
000215 TGHTLP           DEY                               ;BEGGINING OF NAME TRANSFER.
000216                  DEY
000217                  BMI       ALDONE                  ;GOES WHEN NAME IS TRANSFERED.
000218                  LDA       (LOWDS),Y               ;This loop upshifts the variable
000219                  CMP       #'A'+$20                ;  name as it transfers it from the
000220                  BCC       *+4                     ;  program to the space allocated for
000221                  SBC       #$20                    ;  it in the table.
000222                  INY
000223                  STA       (SRCHPT),Y
000224                  BPL       TGHTLP                  ;ALWAYS TAKEN.
000225 ALDONE           LDX       SRCHPT+1                ;GET THE POINTER INTO MEMORY.
000226                  LDA       VARNAM+1                ;GET THE # OF [NAME BYTES +1.] +1.
000227                  SEC                               ;SO WE INDEX TO THE FIRST BYTE
000228                  JMP       FINUP                   ;OF THE ACTUAL VARIABLE,
000229 ; VALTAB: table of # of bytes needed in descriptor (excluding link & name)
000230 ; VALTB2: table of variable TYPE bytes (same order as VALTAB)
000231 VALTAB          DFB       4,1,0,10                ;REAL,!,(FILLER),DBL PR
000232                  DFB       3,2,8                   ;STRING,INT,LONG INT
```

```
000072                  SBC       #MAXPG-MINPG
000073                  STA       ARYPNT+1
000074                  INC       ARYPNTB
000075 CHKUPR           CPX       ARYPNT+1                ;DO UPPER POINTER BYTES MATCH?
000076                  BNE       ZERITA                  ;NO. SO CLEAR SOME MORE.
000077                  LDA       ARYPNTB
000078                  CMP       VARTABB
000079                  BNE       ZERITA
000080                  CPY       VARTAB                  ;DO LOW ORDERS MATCH?
000081                  BCC       ZERITA                  ;CLEAR SOME MORE.
000082                  LDA       DIMFLG
000083                  BNE       DIMRTS1                 ;ALL DONE IF IN A DIM STATEMENT.
000084 ; AT THIS POINT LOWTR,Y POINTS TO THE INDICES.
000085 ; DIMENSIONS. STRATEGY:
000086 ; NUMDIM=NUMBER OF DIMENSIONS.
000087 ; CURTOL.
000088 ; INLPNM:GET A NEW INDICE.
000089 ; MAKE SURE INDICE IS NOT TOO BIG.
000090 ; MULTIPLY CURTOL BY CURMAX.
000091 ; ADD INDICE TO CURTOL.
000092 ; NUMDIM=NUMDIM-1.
000093 ; BNE INLPNM.
000094 ; USE CURTOL*4 AS OFFSET.
000095 GETDEF           LDA       COUNT                   ;AND IT'S JUST THAT SIMPLE!
000096                  STA       INDEX+1
000097                  LDA       #0
000098                  JSR       FMPTR1                  ;POINT TO THE INDICES.
000099                  LDY       #0                      ;ZERO CURTOL.
000100                  STY       CURTOL
000101                  STY       CURTOL+1
000102 INLPNM           PLA                               ;GET LOW INDICE.
000103                  TAX
000104                  STA       INDICE
000105                  PLA                               ;AND THE HIGH PART
000106                  STA       INDICE+1
000107                  CMP       (ARYPNT),Y              ;COMPARE WITH MAX INDICE.
000108                  BCC       INLPN2
000109                  BNE       BSERR7                  ;IF GREATER, 'BADSUBSCRIPT' ERROR.
000110                  INY
000111                  TXA
000112                  CMP       (ARYPNT),Y
000113                  BCC       INLPN1
000114 BSERR7:          JMP       BSERR
000115 OMERR1:          JMP       OMERR
000116 INLPN2:          INY
000117 INLPN1:          LDA       CURTOL+1                ;DON'T MULTIPLY IF CURTOL=0.
000118                  ORA       CURTOL
000119                  CLC                               ;PREPARE TO GET INDICE BACK.
000120                  BEQ       ADDIND                  ;GET HIGH PART OF INDICE BACK.
000121                  STY       INDEX                   ;SAVE IT FOR LATER.
000122                  LDA       (ARYPNT),Y              ;GET THE MAX INDICE.
000123                  PHA                               ;(FOR THE MULTIPLICATION).
000124                  DEY
000125                  LDA       (ARYPNT),Y              ;HIGH BYTE.
000126                  TAX
000127                  PLA
000128                  TAY                               ;LOW BYTE.
000129                  JSR       UMULT                   ;MULTIPLY CURTOL BY LOWTR,Y,Y+1.
000130                  TXA
000131                  ADC       INDICE                  ;ADD IN INDICE.
000132                  TAX
000133                  TYA
000134                  LDY       INDEX
000135 ADDIND:          ADC       INDICE+1
000136                  INY
000137                  STX       CURTOL
000138                  STA       CURTOL+1
000139                  DEC       INDEX+1                 ;COUNT OF DIMS.
000140                  BNE       INLPNM                  ;YES
000141                  JSR       FMAPTR                  ;POINT TO THE VALUES.
000142                  LDX       #$0
000143                  LDY       VARNAM                  ;MAKE X,Y=#OF BYTES PER ELEMENT.
000144                  JSR       UMULT                   ;DO FINAL MULT. TO TAKE ELEMENT
000145                  TXA                               ;SIZE INTO ACCOUNT.
000146                  ADC       ARYPNT                  ;CARRY IS KNOWN TO BE CLEARED.
000147                  STA       VARPNT                  ;LOW ORDER OF ACTUAL VARIABLE POINTER.
000148                  TYA
000149                  ADC       ARYPNT+1
000150                  LDY       ARYPNTB
000151                  JSR       FIXADC
```

```
000153 DATE01          STA       YSAVE
000154                 BRK
000155                 DFB       GETCLOK                ;GET THE TIME FROM SOS.
000156                 DW        DATETAB
000157                 LDA       #3
000158                 STA       KIMY                   ;3 FIELDS BETWEEN /'S OR :'S.
000159                 LDX       #0
000160 DATE02          LDA       CATBUF+2,Y             ;GET A BYTE OF TIME.
000161                 STA       CATBUF+2,X             ;SLAP IT INTO SPOT FOR STRING.
000162                 INX
000163                 INY
000164                 LDA       CATBUF+2,Y
000165                 STA       CATBUF+2,X             ;THIS MAY LOOK SILLY SO FAR, BUT IT WORKS.
000166                 INX
000167                 INY
000168                 LDA       YSAVE                  ;GET THE DELIMETER.
000169                 STA       CATBUF+2,X             ;HERE X GETS AHEAD OF Y.
000170                 INX
000171                 DEC       KIMY                   ;DONE YET?
000172                 BNE       DATE02
000173                 LDA       #8                     ;LENGTH OF ACTUAL STRING TO RETURN.
000174                 BNE       GOTITNOW               ;SAME AS PREFIX$.
000175 PREFIXSET       JSR       CHKEQL                 ;SET PREFIX CODE
000176                 JSR       GETNAME                ;GET NEXT STRING AS IF IT IS A PATHNAME.
000177                 BRK
000178                 DFB       SETPREF                ;SET PREFIX.
000179                 DW        PREFTB2
000180                 BNE       *+6
000181                 JSR       FILSOS                 ;New SOS prefix...refill SOSPATH
000182                 RTS                              ;ALL DONE.
000183 JSERROR         JMP       SERROR
000184 PROGPFXS        EQU       *                      ;Return PROG Prefix as a string
000185                 LDY       PROGPATH               ;Get length
000186                 LDA       #>PROGPATH+1           ;Set STRNG1 pointer to PROGPATH+1
000187                 STA       STRNG1
000188                 LDA       #<PROGPATH+1
000189                 STA       STRNG1+1
000190                 LDA       #0
000191                 STA       STRNG1B                ;Set the Bank too
000192                 JMP       STRCP                  ;Do the string copy to String Space
000193 PROGPFX         EQU       *                      ;Set PROGPATH to a given string
000194                 JSR       CHKEQL                 ;SET PREFIX CODE
000195                 JSR       GETNAME                ;GET NEXT STRING AS IF IT IS A PATHNAME
000196                 LDY       NAMBUF                 ;Get length
000197                 LDA       NAMBUF,Y               ;Check if last char is a '/'
000198                 CMP       #'/'
000199                 BEQ       PROGPFX1
000200                 INY                              ;Length = length + 1
000201                 LDA       #'/'
000202                 STA       NAMBUF,Y               ;Add a '/' at the end if needed
000203                 STY       NAMBUF
000204 PROGPFX1        EQU       *
000205                 JSR       FILPROG                ;It's ok so put it in PROGPATH
000206                 JSR       CNVTPFX1
000207                 JSR       SETSOS                 ;Reset prefix to SOSPATH
000208                 RTS
000209 *
000210 * INSTR.
000211 *
000212 * FIND STRING WITHIN A STRING.
000213 *
000214 INSTR           PLA                              ;STARTING POSITION.
000215                 STA       TEMP
000216                 JSR       CHKCLS                 ;CLOSE PAREN
000217                 PLA
000218                 STA       INDEX                  ;STRING TO FIND.
000219                 STA       FACMO
000220                 PLA
000221                 STA       INDEX+1
000222                 STA       FACMO+1
000223                 PLA
000224                 STA       INDEXB
000225                 STA       FACMOB
000226                 PLA
000227                 STA       ARGMO                  ;SOURCE STRING.
000228                 PLA
000229                 STA       ARGMO+1
000230                 PLA
000231                 STA       ARGMOB
000232                 JSR       NOTNOW                 ;INDEX=(INDEX)
```

```
000233                  STA     YSAVE                   ;LENGTH OF STRING TO FIND.
000234                  LDY     #0
000235                  LDA     (ARGMO),Y               ;LENGTH OF SOURCE STRING.
000236                  CMP     YSAVE
000237                  BCC     GIVMZERO                ;MUST BE AT LEAST AS BIG.
000238                  STA     KIMY                    ;LENGTH OF SOURCE STRING.
000239                  INY
000240                  LDA     (ARGMO),Y
000241                  STA     INDEX2
000242                  TAX
000243                  INY
000244                  LDA     (ARGMO),Y
000245                  STA     INDEX2+1
000246                  LDA     KIMY
000247                  SEC
000248                  SBC     YSAVE
000249                  STA     KIMY                    ;LAST CHAR TO BE MATCHED.
000250                  DEC     TEMP
000251                  JSR     RELINX                  ;MAKE INDEX2 ABSOLUTE.
000252                  LDA     INDEX2
000253                  CLC
000254                  ADC     TEMP
000255                  STA     INDEX2
000256                  BCC     TRYAGM
000257                  LDX     INDEX2+1                ;INC INDEX2+1
000258                  INX
000259                  CPX     #MAXPG
000260                  BCC     *+7
000261                  LDX     #MINPG
000262                  INC     INDEX2B
000263                  STX     INDEX2+1
000264 TRYAGM           LDA     KIMY                    ;LAST TO CMP.
000265                  CMP     TEMP
000266                  BCC     NMTCH2
000267                  LDY     #0
000268 INMAT2           LDA     (INDEX2),Y
000269                  CMP     (INDEX),Y
000270                  BNE     INNOTIT                 ;NO MATCH.
000271                  INY                             ;THAT'S ONE CHAR MATCHED.
000272                  CPY     YSAVE                   ;LAST CHAR?
000273                  BCC     INMAT2                  ;NO, TEST NEXT CARS.
000274                  LDY     TEMP                    ;LOAD CHARACTER COUNT.
000275                  BCS     GIVMIT                  ;ALWAYS.
000276 INNOTIT          INC     TEMP                    ;POSITION TO BEGIN WITH.
000277                  INC     INDEX2                  ;WHERE TO GET THE CARACHTER.
000278                  BNE     TRYAGM
000279                  INC     INDEX2+1
000280                  BNE     TRYAGM                  ;ALWAYS
000281 NMTCH2           EQU     *
000282 GIVMZERO         LDY     #$FF                    ;Y HAS VALUE TO BE RETURNED.
000283 GIVMIT           INY
000284                  TYA
000285                  PHA
000286                  JSR     FRECNOW                 ;FREE THE SOURCE STRING?
000287                  LDX     ARGMOB
000288                  LDA     ARGMO
000289                  LDY     ARGMO+1
000290                  JSR     FRETNOW
000291                  PLA
000292                  TAY
000293                  JMP     SNGFLT                  ;GO FLOAT Y.
000294 *
000295 * SUB$
000296 *
000297 * PARTIAL STRING SUBSTITUTION.
000298 *
000299 SUBLEFT          JSR     PTRGET                  ;GOT TO HAVE A VARIABLE.
000300                  JSR     CHKSTR
000301                  LDA     VARPNT                  ;POINTER TO VARIABLE.
000302                  PHA
000303                  LDA     VARPNT+1
000304                  PHA
000305                  LDA     VARPNTB
000306                  PHA
000307                  LDA     ISARA
000308                  PHA
000309                  JSR     CHKCOM                  ;MUST HAVE COMMA.
000310                  JSR     GETBYT
000311                  TXA
000312                  PHA
```

```
000313                JSR       CHRGOT                    ;COMMA AFTER SECOND ARG?
000314                CMP       #','
000315                BNE       FAKEIT                    ;NO, ASSUME LENGTH OF REPLACEMENT STRING.
000316                JSR       CHRGET
000317                JSR       GETBYT                    ;GET LENGTH LIMIT.
000318                DFB       44
000319 FAKEIT         LDX       #$FF
000320                TXA
000321                PHA                                 ;LENGTH LIMIT.
000322                JSR       CHKCLS                    ;CLOSE PAREN.
000323                JSR       CHKEQL                    ;EQUALS SIGN.
000324                LDA       #$FF
000325                STA       VALTYP                    ;FORMULA MUST BE STRING.
000326                JSR       FRMEVL
000327                PLA                                 ;LENGTH LIMIT
000328                STA       DELTA
000329                PLA
000330                STA       YSAVE                     ;STARTING POSITION.
000331                BEQ       SUBFUC                    ;ILL. QUAN. ERROR.
000332                DEC       YSAVE
000333                LDY       #0
000334                LDA       (FACMO),Y                 ;LENGTH OF STRING.
000335                CMP       DELTA
000336                BCS       *+4
000337                STA       DELTA
000338                LDY       #0                        ;TEMP NOW HAS LENGTH TO REPLACE.
000339                PLA
000340                STA       ISARA
000341                PLA
000342                STA       FORPNTB
000343                PLA
000344                STA       FORPNT+1                  ;STRING TO CLOBBER.
000345                PLA
000346                STA       FORPNT
000347                LDA       (FORPNT),Y
000348                STA       DELTA+1                   ;LENGTH OF STRING TO CLOBBER.
000349                INY
000350                LDA       (FORPNT),Y
000351                STA       INDEX2
000352                INY
000353                LDA       (FORPNT),Y
000354                STA       INDEX2+1
000355                LDA       YSAVE                     ;POSITION TO START REPLACEMENT.
000356                CLC
000357                ADC       DELTA
000358                BCC       *+5                       ;IS THE RESULTING STRING GOING TO BE
000359 SUBFUC         JMP       FCERR                     ;LONGER THAN 255 CHARS?
000360                CMP       DELTA+1                   ;LONGER THAN THE ORIGINAL?
000361                BCC       GOTITTY                   ;NO, DON'T CREATE A NEW STRING.
000362                BEQ       GOTITTY                   ;STILL O.K.
000363 *THE RESULT IS GOING TO BE BIGGER THAN THE ORIGINAL.
000364 *MOVE THE ORIGINAL TO A NEW SPOT AND SWITCH ALL POINTERS.
000365                LDX       FORPNTB
000366                STX       DSCPNTB
000367                LDX       FORPNT                    ;FOR STRIN2.
000368                LDY       FORPNT+1
000369                JSR       STRIN2                    ;GET A NEW SPOT.
000370                TAY
000371                DEY
000372                LDA       #'                        ' ;INITIALIZE THE NEW STRING TO BLANKS
000373 SUBLNK1        STA       (FRESPC),Y                ;IN CASE OF HOLES.
000374                DEY
000375                BNE       SUBLNK1
000376                STA       (FRESPC),Y
000377                LDA       DELTA                     ;GOT TO SAVE THESE THINGS FROM THE LET CODE.
000378                PHA
000379                LDA       YSAVE
000380                PHA
000381                LDA       FACMO
000382                PHA
000383                LDA       FACMO+1
000384                PHA
000385                LDA       FACMOB
000386                PHA
000387                JSR       COPY.M                    ;OUT WITH THE OLD AND IN WITH THE NEW.
000388                PLA
000389                STA       FACMOB
000390                PLA
000391                STA       FACMO+1
000392                PLA
```

```
000232                CMP       #2
000233                BNE       DOPA29                ;LOOKING FOR REFERENCES TO RESOLVE.
000234                LDA       PROCPNT
000235                STA       POINT2
000236                LDA       PROCPNT+1
000237                STA       POINT2+1              ;SAVE CURRENT PROCPNT.
000238                LDA       #6
000239                STA       PROCFLG               ;LOOK FOR DEFINITIONS NOW.
000240                JSR       PERFIND
000241                BNE       DOPA25                ;FOUND IT IF NOT ZERO.
000242                JMP       INVERROR
000243 DOPA29          JSR       NXTPROC               ;GO LOOK AT NEXT.
000244                JMP       DOPA22                ;THIS WILL EVEN WORK ON TYPE 2'S.
000245 DOPA25          LDY       #10                   ;GOT A MATCH!
000246                LDA       (PROCPNT),Y           ;ADD OFFSST OF THIS
000247                CLC                             ;LABEL TO BEGINING OF THIS PROCEDURE.
000248                LDY       #12
000249                ADC       (PROCPNT),Y
000250                TAX
000251                DEY
000252                LDA       (PROCPNT),Y
000253                LDY       #13
000254                ADC       (PROCPNT),Y
000255                PHA                             ;NOW THAT WE HAVE THE ADDRESS OF THE .DEF
000256                TXA                             ;WE NEED TO CONVERT IT INTO AN
                                                                OFFSET INTO THE SEGMENT.
000257                SEC                             ;THAT OFFSET WILL THEN BE ADDED TO ALL .REF'S
000258                SBC       POINT1                ;BEGINNING OF SEGMENT.
000259                TAX
000260                PLA
000261                SBC       POINT1+1
000262                LDY       POINT2+1
000263                STA       POINT2+1
000264                STY       PROCPNT+1
000265                LDA       POINT2                ;RESTORE OLD PROCPNT, AND CLOBBER POINT2
000266                STX       POINT2                ;WITH VALUE OF THIS LABLE.
000267                STA       PROCPNT               ;POINTER INTO TYPE 2 (REFERENCE) FIELD.
000268                LDY       #12
000269                LDA       (PROCPNT),Y           ;# OF REFERENCES TO LABLE.
000270                STA       KIMY
000271                TAX
000272                BEQ       DOPA29
000273                JSR       NXTPROC               ;POINT TO TABLE OF REFERENCES.
000274 DOPA27          LDY       #0
000275                CLC
000276                LDA       POINT1                ;ADD REFERENCE OFFSET TO BASE ADDR.
000277                ADC       (PROCPNT),Y
000278                STA       OFFSST
000279                INY
000280                LDA       POINT1+1
000281                ADC       (PROCPNT),Y
000282                STA       OFFSST+1
000283                CLC
000284                DEY
000285                LDA       POINT2                ;GET VALUE OF LABLE.
000286                ADC       (OFFSST),Y
000287                STA       (OFFSST),Y
000288                INY
000289                LDA       POINT2+1
000290                ADC       (OFFSST),Y            ;THE VALUE OF THE LABLE IS ADDED
                                                                INTO THE REFERENCE.
000291                STA       (OFFSST),Y
000292                LDA       #2
000293                JSR       ADDPROC
000294                DEX
000295                BNE       DOPA27
000296                LDA       KIMY                  ;GOT TO SKIP OVER ALL THOSE ENTRYS.
000297                EOR       #$7
000298                CLC
000299                ADC       #1
000300                BCC       *+4
000301                INC       PROCPNT+1
000302                AND       #$7
000303                ASL       A
000304                BCC       *+4
000305                INC       PROCPNT+1
000306                JSR       ADDPROC
000307                JMP       DOPA20
000308 DOPAS3          LDA       INVPNT                ;GOT ALL LINKS RESOLVED!
000309                STA       PROCPNT
```

```
000310                STA       POINT2
000311                LDA       INVPNT+1
000312                STA       PROCPNT+1
000313                STA       POINT2+1
000314                LDY       #8
000315                LDA       (PROCPNT),Y             ;WHAT TYPE OF INFO IS FIRST?
000316 DOPA31          CMP       #$B                     ;.PROC?
000317                BEQ       DOPA37                  ;YES--SAVE A COPY.
000318                CMP       #$C
000319                BEQ       DOPA37                  ;.FUNC'S GET SAVED TOO.
000320                CMP       #$2
000321                BNE       DOPA32                  ;.REFS ARE VARIABLE LENGTH-- TREAT SPECIALLY.
000322                JSR       DOPA13                  ;SKIP OVER THE EXTRA REFERENCES.
000323 DOPA32          JSR       NXTPROC                 ;POINT TO NEXT ENTRY.
000324                BNE       DOPA31                  ;LOOP TILL DONE.
000325                LDA       POINT2                  ;NEW INVPNT AT END OF LINKAGE
000326                STA       INVPNT                  ;INFO.
000327                LDA       POINT2+1
000328                STA       INVPNT+1
000329                LDA       POINT1                  ;NEW PROCPNT IS BEGINING
000330                STA       PROCPNT                 ;OF PROCEDURE CODE.
000331                LDA       POINT1+1
000332                STA       PROCPNT+1
000333                JMP       DOFILE
000334 DOPA37          LDY       #15                     ;MOVE 16 BYTES.
000335 DOPA38          LDA       (PROCPNT),Y
000336                STA       (POINT2),Y              ;TRANSFER A BYTE.
000337                DEY
000338                BPL       DOPA38
000339                LDA       #16
000340                CLC
000341                ADC       POINT2
000342                STA       POINT2
000343                BCC       DOPA32                  ;UPDATE DESTINATION POINTER AND CONTINUE.
000344                INC       POINT2+1
000345                BNE       DOPA32                  ;ALWAYS.
000346 DOCRNCH         EQU       *                       ;MOVE PROCPNT<INVTAB.INVPNT
000347                LDA       PROCPNT
000348                STA       HIGHDS                  ;NOT NEEDED SINCE THEY ARE THE SAME.
000349                LDA       PROCPNT+1               ;FOR THE BLOCK MOVE.
000350                STA       HIGHDS+1
000351                LDA       INVPNT
000352                STA       HIGHTR
000353                LDA       INVPNT+1
000354                STA       HIGHTR+1
000355                LDA       PROCTABB
000356                STA       HIGHDSB
000357                STA       LOWTRB
000358                STA       HIGHTRB
000359                LDA       INVTAB
000360                STA       LOWTR
000361                LDA       INVTAB+1
000362                STA       LOWTR+1
000363                LDA       #2
000364                CLC
000365                ADC       HIGHTR                  ;INCLUDE 2 BYTES OF END MARK.
000366                STA       HIGHTR
000367                BCC       *+4
000368                INC       HIGHTR+1
000369                JSR       BLTUC                   ;SKRUNCH.
000370                LDA       HIGHDS
000371                STA       INVPNT
000372                LDA       HIGHDS+1
000373                STA       INVPNT+1
000374                LDA       HIGHDSB
000375                STA       INVPNTB
000376                RTS
000377 DEC2PROC        LDA       #2
000378                STA       TEMP
000379                LDA       PROCPNT                 ;SUBTRACT A FROM PROCPNT.
000380                SEC
000381                SBC       TEMP
000382                STA       PROCPNT
000383                BCS       *+4
000384                DEC       PROCPNT+1
000385                RTS
000386 MAKESUR0        JSR       DEC2PROC                ;BUMP POINTER.
000387                LDY       #0
000388                LDA       (PROCPNT),Y
000389                BNE       INVERROR
```

```
000390                INY
000391                LDA       (PROCPNT),Y             ;THIS CODE MAKES SURE
000392                BNE       INVERROR                ;NEXT 2 GUYS ARE ZERO.
000393                RTS
000394 INVERROR        LDX       #ERRIN                  ;BAD INVOKE!
000395                JMP       ERROR
000396 DOAREL          JSR       DEC2PROC                ;HANDLES ONE RELATIVE RELOCATION TABLE.
000397                LDY       #1
000398                LDA       (PROCPNT),Y             ;GET COUNT OF SELF REL POINTERS.
000399                STA       TEMPTR+1
000400                DEY
000401                LDA       (PROCPNT),Y
000402                STA       TEMPTR
000403                BNE       DOARE1
000404 DOARE0          DEC       TEMPTR+1
000405                BMI       DOARTS                  ;ALL DONE.
000406 DOARE1          DEC       TEMPTR
000407                JSR       RELPROC                 ;OFFSST=PROCPNT-(PROCPNT)
000408                CLC
000409                LDY       #0
000410                LDA       (OFFSST),Y
000411                ADC       (SWPPNT),Y
000412                STA       (OFFSST),Y
000413                INY
000414                LDA       (OFFSST),Y
000415                ADC       (SWPPNT),Y
000416                CLC
000417                ADC       #$20
000418                STA       (OFFSST),Y
000419                LDA       TEMPTR
000420                BNE       DOARE1                  ;USUALLY GOES.
000421                BEQ       DOARE0                  ;ALWAYS.
000422 DOARTS          RTS                               ;DONE.
000423 RELPROC         JSR       DEC2PROC                ;BUMP PROCPNT
000424                SEC
000425                LDY       #0
000426                LDA       PROCPNT
000427                SBC       (PROCPNT),Y
000428                STA       OFFSST
000429                INY
000430                LDA       PROCPNT+1
000431                SBC       (PROCPNT),Y
000432                STA       OFFSST+1
000433                RTS
000434 ADDPROC         STA       TEMP
000435                CLC
000436                LDA       PROCPNT
000437                ADC       TEMP
000438                STA       PROCPNT
000439                BCC       *+4
000440                INC       PROCPNT+1
000441                RTS
000442 NXTPROC         LDA       #16
000443                JSR       ADDPROC
000444                LDY       #8
000445                LDA       (PROCPNT),Y
000446                RTS
000447 PERFORM         LDA       #$B                     ;LOOKING FOR A PROCEDURE.
000448                STA       PROCFLG
000449                LDA       #0
000450 PERFEXF         STA       NAMPNT
000451                PLA                               ;PULL & SAVE THE
000452                STA       SAFE                    ;  RETURN ADDRESS WHERE
000453                PLA                               ;  IT WILL BE SAFE
000454                STA       SAFE+1
000455                LDA       #0
000456                STA       NPARAMS
000457                STA       NPOINTS
000458                LDA       #'                      '
000459                LDX       NAMPNT
000460                INX
000461                LDY       #8
000462 PERFE1          STA       NAMBUF-1,X
000463                INX
000464                DEY
000465                BNE       PERFE1
000466                LDX       NAMPNT
000467                JSR       CHRGOT
000468                BNE       PERFE3
000469 PERERR:         JMP       SNERR
```

```
000470 PERFE05         JMP       PERFE5
000471 PERFE2          JSR       CHRGET                ;GET FUNCTION NAME.
000472 PERFE3          BEQ       PERFE05               ;LAST BYTE?
000473                 BCC       PERFE35
000474                 JSR       ISLETC
000475                 BCS       PERFE35
000476                 CMP       #'('                  ;PARAMETER LIST COMMING?
000477                 BEQ       PERFE4
000478                 TXA
000479                 BEQ       PERERR
000480                 JMP       PERFE6
000481 PERFE35         INX
000482                 CMP       #'Z'+1
000483                 BCC       *+4
000484                 SBC       #$20
000485                 STA       NAMBUF-1,X            ;BUILD PROCEDURE NAME IN NAMBUF.
000486                 BNE       PERFE2                ;ALWAYS.
000487 ;
000488 ;  INTERPRET PARAMETER LIST
000489 ;
000490 PERFE4          LDA       #$20                  ;DON'T CARE ABOUT VAR TYPE NOW
000491                 STA       VALTYP
000492                 JSR       CHRGET
000493                 CMP       #'@'                  ;ADDRESS PARAMETER?
000494                 BEQ       PERVAL1               ;YES, IT'S OK
000495                 CMP       #'%'                  ;INTEGER?
000496                 BEQ       PERINT1               ;YES, OK
000497                 CMP       #'&'                  ;LONG INTEGER?
000498                 BEQ       PERLONG1              ;YES, OK
000499                 CMP       #'$'                  ;STRING?
000500                 BNE       *+5                   ;NO! OK
000501                 JMP       TMERR                 ;CAN'T PASS STRINGS (OR FUZZ BALLS)
000502                 JSR       DOPAR                 ;SO MUST BE REAL (I HOPE)
000503                 JSR       CONV2FLT
000504                 JSR       ROUNDER
000505                 LDA       FACEXP
000506                 PHA
000507                 LDA       FACSGN                ;PUSH THE FAC.
000508                 ORA       #$7F
000509                 AND       FACHO
000510                 PHA
000511                 LDA       FACMOH
000512                 PHA
000513                 LDA       FACMO
000514                 PHA
000515                 LDA       #2
000516                 BNE       PERVAL2               ;ALWAYS
000517 PERINT1         JSR       CHRGET
000518                 JSR       DOPAR
000519                 JSR       CONV2INT
000520                 JSR       AYINT
000521                 LDA       FACMO                 ;HIGH BYTE FIRST
000522                 PHA
000523                 LDA       FACMO+1
000524                 PHA
000525                 LDA       #1                    ;TOOK ONE WORD.
000526                 BNE       PERVAL2               ;ALWAYS
000527 PERLONG1        JSR       CHRGET
000528                 JSR       DOPAR
000529                 JSR       CONV2LNG
000530                 LDX       #0
000531 PERLNG1         LDA       FAC,X
000532                 PHA
000533                 INX
000534                 CPX       #8
000535                 BCC       PERLNG1
000536                 LDA       #4                    ;FOUR WORDS
000537                 BNE       PERVAL2
000538 PERVAL1         JSR       CHRGET                ;EAT THE @
000539                 JSR       PTRGET                ;GET POINTER
000540                 LDA       VARPNT+1
000541                 PHA
000542                 LDX       NPOINTS
000543                 STA       BANKPNT+1,X
000544                 LDA       VARPNT
000545                 PHA
000546                 STA       BANKPNT,X
000547                 LDA       VARPNTB
000548                 STA       BANKPNTB,X
000549                 INC       NPOINTS
```

```
000232                STA       INDEX1+1
000233                LDY       FRETOPB
000234                STY       INDEX1B
000235                SEC
000236                SBC       HEADER
000237                JSR       FIXSBC
000238                STA       LOWTR+1                 ;OH, WHAT I'D GIVE FOR A 68000!!
000239                TYA
000240                SBC       HEADER+1
000241                STA       LOWTRB
000242                LDA       STREND
000243                PHA
000244                LDA       STREND+1                ;GOT TO SAVE STREND.
000245                PHA
000246                LDA       STRENDB
000247                PHA
000248                LDA       HIMEM
000249                STA       STREND
000250                LDA       HIMEM+1
000251                STA       STREND+1
000252                LDA       HIMEMB
000253                STA       STRENDB
000254                LDA       #0
000255                STA       DELTA                   ;ZERO OUT DELTA
000256                STA       DELTA+1                 ;SO POINTERS DON'T GET UPDATED
000257                STA       DELTAB                  ;BY THIS MOVE.
000258                JSR       MVDWN                   ;MOVE IT TO A LOWER ADDRESS.
000259                PLA
000260                STA       STRENDB
000261                PLA
000262                STA       STREND+1
000263                PLA
000264                STA       STREND
000265                LDA       LOWTR                   ;FIX FRETOP
000266                STA       FRETOP
000267                LDA       LOWTR+1
000268                STA       FRETOP+1
000269                LDA       LOWTRB
000270                STA       FRETOPB
000271                LDA       HIMEM+1
000272                LDY       HIMEMB
000273                SEC
000274                SBC       HEADER
000275                JSR       FIXSBC                  ;I WISH I WERE A MOTOROLA 68000,
000276                STA       HIMEM+1                 ;YES THAT IS WHAT I'D TRUELY LIKE TO BE,
000277                TYA                               ;CUZ IF I WERE A M. 68000.
000278                SBC       HEADER+1                ;EVERYONE WOULD LOVE TO PROGRAM ME!
000279                STA       HIMEMB
000280                LDA       INDEX2
000281                STA       SEGSIZ5
000282                LDA       INDEX2+1
000283                STA       SEGSIZ5+1
000284                BRK
000285                DFB       MCHG                    ;CHANGE SEG.
000286                DW        SEGTAB5
000287                BEQ       *+5
000288                JMP       SERROR
000289                RTS
000290 EXPAND          EQU       *
000291                LDY       #0                      ;DOES THE OPPOSITE OF SCRUNCH.
000292                TAX                               ;MAX?
000293                BNE       EXPAN0
000294                LDY       #15                     ;SOMETHING BIG.
000295                LDA       #255                    ;CLOSE ENOUGH.
000296 EXPAN0          STA       SEGSIZ6
000297                STY       SEGSIZ6+1
000298 EXPAN1          EQU       *
000299                BRK
000300                DFB       MCHG
000301                DW        SEGTAB6
000302                BNE       EXPAN1                  ;CRUDE BUT EFFECTIVE.  IF IT CAN'T BE DONE,
000303                LDA       SEGSIZ6
000304                ORA       SEGSIZ6+1
000305                BNE       EXPAN2
000306                TYA                               ;Y NON-ZERO IF CALLED WITH ARGUMENT OF ZERO.
000307                BNE       *+5
000308                JMP       OMERR                   ;TRIED TO EXPAND BUT NO MEMORY ERROR.
000309                RTS
000310 EXPAN2          LDA       HIMEM                   ;DO HOWEVER MUCH YOU CAN.  HANG ON AN ERROR.
000311                STA       HIGHTR
```

```
000313 *
000314 UREPEAT          LDY       #0
000315                  LDX       SAVLEN                  ;GET LIT LENGTH
000316 UOUTLP           LDA       (LOWTR),Y
000317                  JSR       OUTDO
000318                  INY
000319                  DEX
000320                  BNE       UOUTLP                  ;DO NEXT ONE
000321                  DEC       REP                     ;ANOTHER TIME?
000322                  BNE       UREPEAT                 ; YEP
000323                  BEQ       UDOLITS                 ;DONE, DO NEXT ONE!
000324                  PAGE
000325 ******************************
000326 USTRVAR          LDA       USVSTRL                 ;GET STR LEN
000327                  STA       SAVLEN
000328                  LDA       REP                     ;IS FIELD LEN > STRING LENGTH
000329                  SEC
000330                  SBC       SAVLEN                  ; ???
000331                  BEQ       UEXACT                  ;JUST FITS SEND IT
000332                  BCS       USFITS                  ;YES
000333                  LDX       REP                     ;GET MAX LENGTH
000334                  STX       SAVLEN                  ; LIMIT TO FIELD SIZE
000335                  JMP       UEXACT
000336 USFITS           LDY       DELIM                   ;WHAT DOING?
000337                  CPY       #'A'
000338                  BEQ       UEXACT
000339                  CPY       #'A'+$20                ;LOWER CASE?
000340                  BEQ       UEXACT
000341                  CPY       #'C'                    ; CENTER IT?
000342                  BEQ       *+6
000343                  CPY       #'C'+$20                ;LOWER CASE?
000344                  BNE       UEATME
000345                  LDX       SAVLEN
000346                  BEQ       UEATME
000347                  LSR       A
000348 UEATME           TAX                               ;X=LENGTH OF LEADING SPACES
000349                  INX
000350 ULSPCS           DEX                               ; AM I DONE?
000351                  BEQ       UEXACT
000352                  JSR       OUTSPC
000353                  DEC       REP
000354                  BNE       ULSPCS
000355 UEXACT           LDY       #0                      ;STR START
000356                  LDX       SAVLEN                  ;GET STRING LENGTH
000357                  BEQ       UNULL                   ;IF NULL STRING, DO SPEC
000358 UEXACT2          LDA       (INDEX),Y               ; GET STRING CHR
000359                  JSR       OUTDO                   ;PRINT IT
000360                  INY
000361                  DEC       REP
000362                  DEX                               ; AM I DONE?
000363                  BNE       UEXACT2                 ;ON WITH THE SHOW
000364 UNULL            LDX       REP
000365 UFILL3           BEQ       UFRESTR
000366                  JSR       OUTSPC
000367                  DEX
000368                  JMP       UFILL3
000369 UNUMVAR          JSR       UNUMEDIT                ;DO NUMBERS
000370 UFREMSK          JSR       UFRESPC                 ;FREE UP MASK AREA
000371                  JMP       UNXTEXP                 ;DO NEXT EXPR
000372                  PAGE
000373 UFRESTR          EQU       *
000374                  JSR       FRECNOW                 ;FREE IT IF IT WAS A TEMPORARY STRING
000375                  JMP       UFREMSK                 ;NOW MASK TOO
000376 ****************************************
000377 *
000378 * BEGIN SUBROUTINES
000379 *
000380 ********************************
000381 UFRESPC          LDA       #MSKLEN                 ;FIXED LENGTH
000382                  LDY       FRESPC+1
000383                  LDX       FRESPC                  ;ADRS OF AREA TO FREE
000384                  STX       INDEX                   ;FOR FRESPA.
000385                  STY       INDEX+1
000386                  LDX       FRESPCB
000387                  STX       INDEXB
000388                  JMP       FRESPA
000389 ******************************
000390 UREGET           PLA                               ;CLEAR STK
000391 UGETCHR          JSR       UGETCH
000392                  PHP                               ;SAVE STATUS
```

```
000393                CMP       #'                    ' ;A SPACE?
000394                BEQ       UREGET                ;YES, IGNORE
000395                PLP                             ;RECOVER STATUS
000396                RTS
000397 *********************************
000398 UGETCH          INC       SPCNDX
000399 IGOTCH          LDY       SPCNDX
000400                LDA       #0                    ;IF END OR OVRFLO
000401                CPY       #$FF                  ; PAST MAX?
000402                BNE       UGETCKL               ;NOT OVRFLO
000403                DEC       SPCNDX                ;FOR REUSE
000404                BNE       UGETNE
000405 UGETCKL         CPY       SPCLEN
000406                BCS       UGETNE
000407                LDA       (SPCPTR),Y
000408                CMP       #','                  ;END OF SPEC?
000409                BEQ       UGETEND               ;YES
000410                CMP       #'A'                  ; IS IT < A ?
000411                BCC       UGETCKD               ;YES
000412                CMP       #'Z'+1                ; IS IT A LETTER
000413                BCC       UGETVS                ;YES SET V ON
000414 UGETCKD         CMP       #':'                  ;NO, IS IT A DIGIT ?
000415                BCS       UGETNE                ;NO SPECIAL
000416                SEC
000417                SBC       #'0'                  ; TAKE OUT ZERO
000418                SEC
000419                SBC       #$D0                  ;AND IT COMPLEMENT
000420 UGETNE          TAY                             ; EQU ON ZERO
000421 UGETEND         CLV
000422                RTS
000423 UGETVS          BIT       *+4                   ; SET V BIT ON
000424                CMP       #$40                  ;CS,VS,NE
000425                RTS
000426                PAGE
000427 UATOMSK         LDY       MSKNDX
000428                INY
000429                CPY       #MSKLEN               ;TOO MUCH?
000430                BCS       USYERR                ;YEP
000431                STA       (MASKPT),Y            ;ADD IT
000432                STY       MSKNDX
000433                RTS
000434 USYERR          JMP       USNERR
000435 *******************************
000436 UDOMASK         PHA                             ;SAVE A
000437                LDY       DIGCTR                ; ANY TO DO
000438                BEQ       UDONT                 ;EXIT
000439                LDX       #1                    ;ASSUME AFTER
000440                LDA       #4                    ;DPT MASK BIT
000441                BIT       EMASK                 ;DPT YET?
000442                PHP                             ;SAVE Z BIT
000443                BNE       UAFDPT                ;YES, NO CHECK
000444                LDA       NMASK
000445                CMP       #ZTYP                 ;Z SPEC?
000446                BNE       UNOTZ                 ;NO
000447                LDA       FLTMSK                ;FLOAT AND Z IS SYNTAX
000448                BNE       USYERR                ;BAD SPEC!
000449 UNOTZ           DEX                             ;X=0 FOR B4 DPT
000450 UAFDPT          TYA                             ;GET FIELD SIZE
000451                STA       DIGB4DPT,X            ;SAVE FOR EDITOR
000452                LDX       NMASK
000453                BEQ       USYERR                ;NO DIGITS B4 DPT AND FLOAT SPEC!
000454                CPX       #CMATYP               ;IS THIS #
000455                BNE       UANYSIZ               ;YES ANY SIZE
000456                CMP       #5                    ;MIN SIZE IS #,###
000457                BCC       USYERR                ;FIELD TOO SMALL FOR COMMA INSERT
000458 UANYSIZ         LDA       #0                    ;DIGITS = 0
000459 UDOMSKL         JSR       UATOMSK               ;STICK IT IN
000460                DEC       DIGCTR
000461                BNE       UDOMSKL
000462                PLP                             ;GET Z BIT
000463                BNE       UDONT
000464                STY       TENZNDX
000465 UDONT           LDY       #2                    ;RESTORE Y
000466                LDA       #0
000467                STA       NMASK                 ;NEW SUB FIELD
000468                PLA
000469                RTS
000470                PAGE
000471                REP       60
000472 *
```

```
000473 *      Spec Syntax Checker
000474 *
000475 * Validates the Syntax of all Specs, Builds Numeric
000476 *   Edit Mask and Pointers to Literal Specs
000477 *
000478                REP       60
000479 IMSYNCK         EQU       *
000480                LDY       #1
000481                STY       SPECTYP                  ; NONE IS DEFAULT
000482                DEY
000483                STY       CMAFILL                  ;NO FILLER
000484                STY       DIGB4DPT
000485                STY       DIGAFDPT
000486                STY       FLTMSK                   ;NO FLOAT
000487                STY       DIGCTR
000488                STY       EMASK                    ; NO EDITING
000489                STY       NMASK                    ; NO DIGITS
000490                DEY                                ;Y=FF
000491                STY       DPTNDX                   ;NO DPT
000492                STY       TENZNDX                  ; NO 10**0 DIGIT
000493                STY       MSKNDX                   ;INIT AT FF
000494                JSR       UGETDL                   ; GET NEXT REP & DELIM
000495                BPL       UNUMTYP                  ;GO DO NUM TYPES
000496                STA       SPECTYP                  ;SET TYPE
000497                BIT       SPECTYP                  ;LIT/STR ?
000498                BVS       UCHEKCM                  ;ALL DONE IF STRING!
000499                LDA       DELIM
000500                CMP       #'/'                     ; CR OUT?
000501                BEQ       UCRLIT
000502                CMP       #'X'                     ; SPACE OUT?
000503                BEQ       USPLIT
000504                CMP       #'X'+$20
000505                BEQ       USPLIT
000506                LDA       SPCNDX
000507                SEC                                ;PLUS 1
000508                ADC       SPCPTR
000509                LDY       SPCPTR+1
000510                LDX       SPCPTRB
000511                BCC       *+6
000512                INY
000513                JSR       FIXYX
000514                STA       LOWTR
000515                STY       LOWTR+1
000516                STX       LOWTRB
000517                LDX       #0-1
000518 UQSCAN          INX                                ;NEXT CHR
000519                JSR       UGETCH                   ;MOVE PTR ALONG!
000520                CMP       #0                       ; END OF SPEC?
000521                BEQ       USNERRL                  ;NOTAIL"
000522                CMP       #'"'                     ; END YET?
000523                BNE       UQSCAN
000524                STX       SAVLEN                   ;RETURN LENGTH OF LITERAL
000525                JSR       UGETCHR                  ; NEXT AFTR "
000526 UCHEKCM         JSR       IGOTCH                   ;DONE AFTER SPEC?
000527                BNE       *+3
000528                RTS
000529 USNERRL         JMP       USNERR
000530 UCRLIT          LDA       #$0A                     ;A LF.
000531                STA       LITRLCH+1
000532                LDY       #2
000533                LDA       #$0D                     ;A CR!!
000534                BNE       *+6
000535 USPLIT          LDA       #'                       ' ;A SPACE
000536                LDY       #1
000537                STA       LITRLCH
000538                LDX       #<LITRLCH
000539                LDA       #LITRLCH
000540                STA       LOWTR
000541                STX       LOWTR+1
000542                STY       LOWTRB
000543                STY       SAVLEN
000544                BNE       UCHEKCM                  ; DONE NOW?
000545                PAGE
000546 UNUMTYP         EQU       *
000547                LDY       #2                       ;A TWO
000548                CMP       #$02                     ; IS IT * ?
000549                BNE       UDOLLR
000550                CPY       REP                      ;** OR 2*?
000551                BNE       UNSERR                   ;BAD STUFF
000552                LDA       #$10                     ; ** BIT ON
```

```
+------------------------------------------------------------------------------
|
|  File    : "B3PRU2.TEXT.PRETTY"
|  Created : Tuesday, December 30, 1997               5:14:31 PM
|  Modified: Wednesday, December 31, 1997             4:37:08 PM
|
+------------------------------------------------------------------------------

000001 ; ###########################################################################################
000002 ; #   PROJECT  :  Apple /// Business BASIC 1.3 (6502 Assembly Source Code)
000003 ; #   FILE NAME:  B3PRU2.TEXT
000004 ; ###########################################################################################
000005
000006                  REP       70
000007 *
000008 * GETDL fetches the next monochromatic delimiter sequence
000009 *   from the spec string via the subroutine UGETCH.
000010 *
000011 * It converts the Repeat Factor (REPFAC) into Binary in REP,
000012 *   and compresses consecutive occurances of the same delimiter,
000013 *   incrementing REP to compensate.
000014                  REP       70
000015 * INPUTS DATA ITEMS
000016 *
000017 * SPCNDX    Points to last used char in IMAGE string
000018 *
000019 * SPCPTR    Points to String Base
000020 * SPCLEN    String length (1-255)
000021 *
000022 * OUTPUT DATA ITEMS
000023 *
000024 * DELIM     The actual valid delimiter
000025 *
000026 * REP       The number of them
000027 *
000028                  REP       70
000029 *
000030 * ERROR EXITS
000031 *
000032 * If an Invalid Delinator is found, a SYNTAX Error will occur.
000033 *
000034 * If a repeat factor is >255 or the sum of repeat and
000035 *   consecutive delinators is >255 then an ILLEGAL QUANTITY
000036 *   Error and SYNTAX Error respectively, will occur.
000037 *
000038 * If SPCNDX is at the end of IMAGE when GETDL is called, it
000039 *   will return up 1 level higher via the stack clear
000040 *   of the caller's Return Address.
000041                  REP       70
000042 UGETDL          EQU       *
000043                  LDY       #0
000044                  STY       DELIM
000045                  STY       REP
000046 UCOMA           JSR       UGETCHR                 ; GET A CHAR
000047                  BNE       UGOT1
000048                  CMP       #','                    ; IS IT A COMMA?
000049                  BEQ       UCOMA                   ;YES, IGNORE
000050                  PLA                               ; NO MUST BE EOI
000051                  PLA
000052                  RTS
000053 UGOT1           BCS       UDLCHR                  ;DO DELIM
000054                  LDX       DELIM                   ;GOT A DELIM YET?
000055                  BEQ       UG1STD                  ;NOTHING SO FAR
000056                  CPX       #'9'                    ;ONLY DIGITS SO FAR ?
000057                  BNE       UGVAL                   ;NO, GO VALIDATE
000058 UG1STD          LDY       REP                     ;REP*10>250?
000059                  CPY       #26
000060                  BCC       *+8                     ;NO CHANCE
000061 UIQERR          JSR       UFRESPC
000062                  JMP       FCERR                   ;DUMB USERS!!
000063                  LDX       #'9'
000064                  STX       DELIM                   ;I GOT A DIGIT BEFORE
000065                  TYA
000066                  ASL       A
000067                  ASL       A
000068                  ASL       A                       ;A=REP*8
000069                  ADC       REP
000070                  ADC       REP                     ;A=REP*10
000071                  STA       REP
000072                  LDY       SPCNDX
```

```
000073                LDA       (SPCPTR),Y              ;ADD THE DIGIT
000074                AND       #$0F                    ; ZAP ASCII
000075                ADC       REP                     ;REP*10+DIGIT
000076                BCS       UIQERR                  ;DIG >5 FOR REP=25
000077                STA       REP                     ;REP=REP*10+DIGIT
000078 UGNXTC         JSR       UGETCHR                 ;INC & LDA
000079                BEQ       UGEND                   ;END OF SPEC!
000080                BNE       UGOT1                   ;PROCESS
000081 UGVAL          DEC       SPCNDX                  ;PICK SAME NXT TIME
000082 UGEND          LDA       DELIM
000083                CMP       #'Z'+1
000084                BCC       *+4
000085                SBC       #$20
000086                LDY       #DLMCNT-1
000087 UVALOOP        CMP       DLMTBL,Y
000088                BEQ       UGETND
000089                DEY
000090                BPL       UVALOOP                 ; TRY NEXT
000091 UGERRX         JSR       UFRESPC
000092                JMP       USNERR                  ;BAD DELIM
000093 UGETND         LDA       STYPTBL,Y
000094                RTS
000095 UDLCHR         LDY       DELIM
000096                BEQ       U1STDL
000097                CPY       #'9'                    ;REP FAC ONLY?
000098                BNE       UDLCK
000099                LDY       REP                     ;ALL 0 REP?
000100                BEQ       UIQERR                  ;THE GUY IS NUTS!
000101                BNE       USETDL
000102 U1STDL         LDY       #1
000103                STY       REP
000104 USETDL         STA       DELIM                   ; SET DELIM
000105                CMP       #'"'                    ;LITERAL START?
000106                BEQ       UGEND                   ;YES, STOP NOW
000107                BNE       UGNXTC                  ;DO NEXT
000108 UDLCK          CMP       DELIM                   ; SAME DELIM AS LAST?
000109                BNE       UGVAL                   ;NO, DONE
000110                CMP       #'"'                    ;" AFTER DELIM="?
000111                BEQ       UGERRX
000112                INC       REP
000113                BNE       UGNXTC                  ;DO NEXT
000114                BEQ       UGERRX
000115 DLMTBL         EQU       *
000116                ASC       'AX"+-&$.ECR/#Z*'
000117 STYPTBL        EQU       *
000118 DLMCNT         EQU       STYPTBL-DLMTBL
000119                DFB       $FF                     ;STRG
000120                DFB       $80                     ;LIT
000121                DFB       $80                     ;LIT
000122                DFB       $21                     ;EDIT
000123                DFB       $20                     ;EDIT
000124                DFB       CMATYP                  ;$44
000125                DFB       $10                     ;EDIT
000126                DFB       $04                     ;EDIT
000127                DFB       $08                     ;EDIT
000128                DFB       $FF                     ;STRG
000129                DFB       $FF                     ;STRG
000130                DFB       $80                     ;LIT
000131                DFB       DTYP                    ;$42
000132                DFB       ZTYP                    ;$41
000133                DFB       $02                     ;EDIT
000134 *        FORMAT OF TYPE BYTES
000135 *     BIT 7 6 5 4 3 2 1 0
000136 *         T T S $ E . * +
000137 *
000138 *         WHERE S=SIGN AND +=WHICH SIGN
000139 *         TT=0 1  FOR DIGIT TYPE
000140 *           =0 0  FOR EDITING TYPE
000141 *           =1 0  FOR LITERAL TYPE
000142 *           =1 1  FOR STRING TYPE
000143 *
000144 FILLTABL        ASC       '0   '
000145 CMATBL          DFB       0,0,0,$80
000146                PAGE
000147 ******************************
000148 * UNPACK BCD MANTISSA
000149 ******************************
000150 * ALL 6502 REGS DESTORYED
000151 ******************************
000152 * OUTPUTS
```

```
000153 * A IS ZERO
000154 * P HAS ZERO FLAG SET
000155 * Y 18  OR $12
000156 * X 10  OR $0A
000157 * OUTPUT FORM IS ALWAYS
000158 * 112233445566778899@
000159 *   @ =$00
000160 ********************************
000161 UUNPACK         EQU       *
000162                 LDY       #2
000163                 LDX       #$FF
000164                 STX       HALF                ; HIGH HALF FIRST
000165                 INX                           ;X=0
000166 UHLOOP          LDA       FACT,X              ; GET HALF
000167                 BIT       HALF                ; WHICH HALF?
000168                 BPL       UPUSELOW            ; LSH
000169                 INC       HALF                ;LOW HALF NEXT
000170                 LSR       A
000171                 LSR       A
000172                 LSR       A
000173                 LSR       A
000174                 BPL       UPUSEA
000175 UPUSELOW        DEC       HALF                ;UPR HALF NEXT
000176                 INX                           ; NEW BYTE TOO
000177                 AND       #$F                 ; MASK TOP
000178 UPUSEA          EQU       *                   ;IN HALF BCD. (NOT ASCII)
000179                 CPY       #3
000180                 BCS       UGTABYT             ;MIDDLE OF #, DON'T SKIP ZEROES.
000181                 CMP       #0                  ;LEADING ZERO?
000182                 BNE       UGTABYT             ;NO, SKIP.
000183                 DEC       ISARA               ;VIRTUAL EXPONENT FOR LONG INTEGER #'S.
000184                 DEC       ISARA
000185                 BNE       UHLOOP              ;ALWAYS.
000186 UGTABYT         STA       BCDSTR,Y            ; PUT IT
000187                 INY                           ; NEXT CHAR
000188                 CPY       #22
000189                 BNE       UHLOOP              ; NO SO LOOP
000190                 LDA       #0
000191                 STA       BCDSTR,Y            ;TRAILING ZERO
000192                 RTS
000193                 PAGE
000194 UNUMEDIT        INC       MSKNDX              ;= LENGTH
000195                 LDA       FACEXP              ;THIS SEPERATES BCD EXP
000196                 CMP       #$80                ;EXTEND SIGN TO C
000197                 ROR       A
000198                 ROR       A                   ;MANT SGN TO BIT 7
000199                 STA       FACSGN              ;SAVE IT
000200                 ROL       A                   ;EXP NOW TRUE SIGNED #
000201                 SEC
000202                 SBC       #1                  ;ADJUST FOR BCD FORM
000203                 CLC
000204                 ADC       ISARA               ;ADD SCALE FACTOR.
000205                 STA       FACEXP              ;-63 -> +64
000206                 CMP       #MINSGN             ;MINEXP
000207                 BCS       UROKNOW
000208                 CMP       #MAXSGN+1
000209                 BCC       UROKNOW
000210                 JSR       UFRESPC
000211                 JMP       FCERR
000212 MINSGN          EQU       $9D
000213 MAXSGN          EQU       $63
000214 UROKNOW         JSR       USGNSCN             ; DO FIXED SIGN
000215                 LDY       #0
000216                 STY       CMACTR
000217                 STY       DIGCTR              ;GET NEXT INIT
000218                 LDA       #$08                ;EEEE BIT
000219                 BIT       EMASK               ;INTEGER OR EE FORMAT?
000220                 BEQ       *+5                 ;INTEGER
000221                 JMP       UEEEDIT
000222                 BIT       CMAFILL             ;CMA INSERTION?
000223                 BPL       UNOCMAS             ;NO EXP OK
000224                 LDA       DIGB4DPT            ;AT LEAST 5 THERE
000225                 AND       #3                  ;IS IT DIVISIBLE BY 4?
000226                 CMP       #1                  ;C=0 IF TRUE
000227                 LDA       DIGB4DPT
000228                 TAX
000229                 BCC       UNOSBC1
000230                 SBC       #1
000231 UNOSBC1         EQU       *
000232                 LSR       A                   ;CALC MAC DIGITS
```

```
+------------------------------------------------------------------------------
|
|  File    : "DISKSTUF2.TEXT.PRETTY"
|  Created : Tuesday, December 30, 1997               5:14:34 PM
|  Modified: Wednesday, December 31, 1997             4:37:11 PM
|
+------------------------------------------------------------------------------

000001 ; ###############################################################################################
000002 ; #   PROJECT  :  Apple /// Business BASIC 1.3 (6502 Assembly Source Code)
000003 ; #   FILE NAME:  DISKSTUF2.TEXT
000004 ; ###############################################################################################
000005
000006 *
000007 * ROUTINE TO CALL BEFORE EACH READ OR WRITE - DETERMINES IF THE
000008 * GUY IS DOING THE CORRECT OPERATION ON THE CORRECT TYPE OF FILE
000009 PREBIN:         LDA       #BINTIP
000010                DFB       44
000011 PRETXT:         LDA       #TXTTYP
000012                STA       TYPSAV
000013                LDY       FCBNDX
000014                LDA       FCB,Y
000015                BEQ       DUMSHIT
000016                LDA       FCB+XFLGS,Y               ;IS OPEN COMPLETE YET?
000017                ASL       A                         ;BIT 6 ON IF NO
000018                BPL       CKTYP                     ;YES, JUST CHECK OPERATION TYPE
000019                ASL       A                         ;MAKE 0 STATUS
000020                STA       FCB+XFLGS,Y
000021                LDA       FCB+XUID,Y
000022                AND       #$0F
000023                CMP       #UNKNTY
000024                BNE       FINOPN
000025                JSR       GETNDX                    ;CALC PTR TO FILE BUFFER
000026                LDA       RWRFNM                    ;SAVE REFNUM
000027                PHA
000028                LDA       #>NDXPTR
000029                STA       PTHPTR
000030                LDA       #<NDXPTR+1
000031                STA       PTHPTR+1
000032                JSR       GETFI                     ;GET FILE INFO
000033                LDA       FID                       ;HAS THE TYPE ALREADY BEEN SET UNBEKNOWNST TO US?
000034                CMP       #UNKNTY
000035                BNE       GTTYP1                    ;IF SO, DON'T RESET IT.
000036 * THE FILE TYPE HAS NOT YET BEEN SET, SO WE CAN NOW SET IT TO
000037 * WHICHEVER TYPE (TEXT OR BINARY) THAT WAS REQUESTED
000038                JSR       TSTOUT
000039                LDA       TYPSAV
000040                STA       FID
000041                LDY       #SFI
000042                JSR       SETGO
000043                LDA       TYPSAV                    ;A TEXT FILE?
000044                CMP       #TXTTYP
000045                BNE       GTTYP1
000046                PLA                                 ;GET REF NUM
000047                PHA                                 ;KEEP STACK CLEAN
000048                STA       RWRFNM
000049                JSR       EXRTS                     ;SET UP IS.NEW.LINE
000050 GTTYP1:         PLA                                 ;GET BACK REFNUM
000051                STA       RWRFNM
000052                LDY       FCBNDX
000053                LDA       FCB+XUID,Y
000054                AND       #$F0
000055                ORA       TYPSAV
000056                STA       FCB+XUID,Y
000057 FINOPN          JSR       RPOSN
000058                LDY       FCBNDX
000059 *
000060 * NOW CHECK FOR A FILE TYPE MISMATCH
000061 *
000062 CKTYP:          LDA       FCB+XUID,Y                ;GET TYPE OF FILE
000063                EOR       TYPSAV
000064                AND       #$0F
000065                BNE       ERTYP
000066                RTS
000067 ERTYP:          JMP       CHKERR
000068                PAGE
000069 * GET THE NECESSARY BUFFER POINTER AND
000070 * PUT THE FILE NAME THERE FOR PREBIN
000071 DOPEN:          JSR       GTFLNO
000072                BEQ       DOP2                      ;BRANCH IF NOT PREVIOUSLY OPEN
```

```
000073                STX       XSAV                    ;SAVE FILE #
000074                JSR       CLOSEM                  ;GO DO IT.
000075                LDX       XSAV                    ;GET REF # BACK
000076                JSR       GTFLNO1
000077 DOP2           LDX       #$00                    ;LOOK FOR 'AS' OPTION
000078                LDA       #ASTKN
000079                JSR       TRYESC
000080                BNE       NOTAS                   ;NAW, FORGET IT
000081                JSR       CHRGET                  ;AS WHAT?
000082                CMP       #INPTKN                 ;INPUT?
000083                BNE       AS1
000084                LDX       #$10                    ;INPUT ONLY
000085 AS1            CMP       #OUTTKN                 ;OUTPUT?
000086                BNE       AS2
000087                LDX       #$20                    ;OUTPUT ONLY
000088 AS2            LDA       #EXTKN                  ;EXTENSION?
000089                JSR       TRYESC
000090                BNE       AS3
000091                LDX       #$60                    ;EXTENSION.
000092 AS3:           TXA                               ;AS SOMETHING?
000093                BEQ       GVERR                   ;NO, HE SCREWED UP.
000094                JSR       CHRGET
000095 NOTAS          TXA
000096                LDY       FCBNDX                  ;PUT FILE MODE INTO FCB
000097                STA       FCB+XUID,Y
000098                LSR       A
000099                LSR       A
000100                LSR       A
000101                LSR       A
000102                CMP       #2
000103                BCC       *+4
000104                LDA       #$3
000105                STA       INREQ
000106                JSR       CHKCOM                  ;SYNTAX IS OPEN #N,<NAME>
000107                JSR       GETNAME
000108                JSR       GRECLN                  ;GET REC LEN
000109                JMP       NOTAS2
000110 GVERR:         JMP       SNERR
000111 GVTT           JSR       CLOSEM2                 ;CLOSE IN SOS&WIPE FCB
000112                JMP       MSMTCH
000113 * NOW GO OPEN IT & CREATE IT IF NECCESSARY
000114 NOTAS2:        LDA       #UNKNTY                 ;UNKNOWN TYPE OF FILE
000115                STA       TYPSAV
000116                JSR       OPNP2
000117                LDY       FCBNDX
000118                LDA       #$40                    ;EVEN IF A KNOWN TYPE, MUST READ IN DATA LATER.
000119                STA       FCB+XFLGS,Y
000120 * FILE HAS BEEN OPENED- NOW SET UP FCB
000121                LDA       REFOUT                  ;GET REFNUM
000122                STA       FCB+XRFNM,Y
000123                LDA       FSTYP                   ;IS IT A DIRECTORY?
000124                CMP       #$0D
000125                BCS       GTDIR
000126                LDA       FID
000127                CMP       #UNKNTY                 ;ONLY ALLOW UNDESIGNATED FILES,
000128                BEQ       OPG2
000129                CMP       #BINTIP                 ;BINARY DATA
000130                BEQ       OPG2
000131                CMP       #TXTTYP
000132                BNE       GVTT
000133 OPG2           AND       #$0F                    ;GET FILE TYPE TO L.O. 4 BITS
000134                ORA       FCB+XUID,Y
000135                STA       FCB+XUID,Y
000136                LDA       FAUX                    ;GET RECLEN FROM DIRECTORY
000137                STA       FCB+XRECL,Y
000138                LDA       FAUX+1
000139                STA       FCB+XRECL+1,Y
000140                ORA       FAUX
000141                BNE       RNDYSLAB
000142                LDA       #2
000143                STA       FAUX+1
000144                STA       FCB+XRECL+1,Y
000145 RNDYSLAB       LDA       FID                     ;IF TEXT FILE, DO SPECIAL
000146                CMP       #TXTTYP
000147                BEQ       GTTXT
000148 * EVERYTHING READY. GET BUFFER
000149 OPLP:          LDY       FAUX+1
000150                LDX       FAUX
000151                STX       DVSR                    ;FOR EXTENSION MODE
000152                STY       DVSR+1
```

```
000153                BEQ       OPGD1
000154                INY                           ;ODD SIZE RECLEN, GIVE HIM ROOM IN BUFFER
000155 OPGD1          STY       IOPGCN              ;ASK SOS FOR BUFFER
000156                LDA       #0
000157                STA       IOPGCN+1
000158                LDA       #2                  ;FIND ANYTHING ANYWHERE
000159                STA       ISRCHMD
000160                LDA       #18
000161                STA       ISEGID
000162                LDY       #FND                ;FIND A SEGMENT.
000163                JSR       SETUP
000164                JSR       GOSOS
000165                BEQ       OPGOOD
000166                CMP       #$E1                ;SEG REQUEST DENIED?
000167                BNE       OPSHIT
000168                LDY       FAUX+1
000169                LDX       FAUX
000170                BEQ       *+3
000171                INY
000172                TYA
000173                JSR       SCRUNCH
000174                JMP       OPLP
000175 GTDIR:         JMP       SETCAT
000176 OPSHIT         JMP       SERROR
000177 GTTXT          JSR       EXRTS               ;DO STUFF (SET IS.NEW.LINE)
000178                LDA       #$FF                ;SEGNUM OF $FF INDICATES A TEXT FILE
000179                STA       OSEGNM
000180 OPGOOD         LDX       FCBNDX
000181                LDA       BSBNKP
000182                ORA       #$80                ;MAKE A VIRTUAL ADDRESS.
000183                STA       FCB+XBUFPT,X
000184                TAY
000185                LDA       BSBNKP+1            ;GET ACTUAL PAGE NO
000186                SEC
000187                SBC       #$20                ;MAKES A VIRTUAL
000188                STA       FCB+XBUFPT+1,X
000189                JSR       FIXSBC
000190                LDA       OSEGNM              ;SAVE THE SEG NUM FOR CLOSE
000191                STA       FCB+XSEGNM,X
000192                LDA       FCB,X               ;PUT REFNUM BACK
000193                STA       RWRFNM
000194                LDA       FCB+XUID,X          ;IS IT AS EXTENSION?
000195                AND       #$40
000196                BEQ       NOEXT
000197                LDA       FID
000198                CMP       #UNKNTY
000199                BEQ       NOEXT
000200                CMP       #TXTTYP             ;DO TEXT FILES DIF.
000201                PHP
000202                LDA       #0
000203                STA       FCB+XFLGS,X
000204                PLP
000205                BEQ       TXTEXT
000206                JSR       GETRN1
000207                LDA       RMNDR
000208                TAY
000209                ORA       RMNDR+1             ;AT BEG. OF REC?
000210                BEQ       OPGOT
000211                TYA
000212                BNE       *+4
000213                DEC       RMNDR+1             ;SO AS EXTENSION WILL WORK PROPERLY.
000214                DEC       RMNDR
000215 OPGOT          LDA       RMNDR+1
000216                PHA
000217                LDA       RMNDR
000218                PHA
000219                JSR       RPOSN
000220                LDY       FCBNDX              ;PUT INDEX INTO BUFFER
000221                PLA
000222                STA       FCB+XBUFOFS,Y
000223                PLA
000224                STA       FCB+XBUFOFS+1,Y
000225                RTS
000226 NOEXT:         JSR       GETNDX
000227                LDA       OSEGNM              ;TEXT FILE?
000228                BMI       XYZZY
000229                LDA       NAMBUF              ;LEN OF NAME
000230                TAX
000231                INX
000232                LDY       #0
```

```
000152 * NOW LOAD THE PROGRAM....
000153                 JSR       DOLD2                    ;LOAD IT...
000154                 JSR       FLOAD                    ;CLEAN UP.
000155 * NOW RESTORE THE OLD VARIABLES
000156                 LDA       ARYTAB                   ;THIS IS WHERE TO MOVE TO.
000157                 STA       LOWTR
000158                 SEC                                ;ALSO CALCULATE DELTA
000159                 SBC       FAC
000160                 STA       DELTA
000161                 LDA       ARYTAB+1
000162                 STA       LOWTR+1
000163                 SBC       FAC+1                    ;FAC+0,1 IS WHERE TO START MOVING DOWN FROM
000164                 LDY       ARYTABB
000165                 STY       LOWTRB
000166                 JSR       FIXSBC
000167                 STA       DELTA+1
000168                 TYA
000169                 SBC       FAC+2
000170                 STA       DELTAB
000171                 LDA       FAC
000172                 STA       ARYTAB
000173                 STA       INDEX1                   ;BEGIN MOVE LOCATION
000174                 LDA       FAC+1
000175                 STA       ARYTAB+1
000176                 STA       INDEX1+1
000177                 LDA       FAC+2
000178                 STA       INDEX1B
000179                 STA       ARYTABB
000180                 JSR       MVDWN                    ;MOVE IT FOLKS!
000181 FRUN:           JSR       STXTPT                   ;RESET TEXT POINTER
000182                 JSR       STKINI                   ;CLEAN UP THE STACK
000183                 LDA       LINNUM                   ;DID HE SPECIFY A LINE?
000184                 ORA       LINNUM+1
000185                 BEQ       *+5
000186                 JSR       LUK4IT                   ;POSITION TO THE LINE
000187                 JMP       NEWSTT                   ;AND GO RUN!
000188 * ROUTINE TO SET UP FOR CHAIN AND RUN
000189 LDRUN:          LDA       #1                       ;ONLY LOAD FILES, DON'T CREATE ANY
000190                 STA       CMDFLG                   ;Set COMMAND call flag
000191                 LDX       #PRGTY
000192                 JSR       OPENIT
000193                 LDA       #1                       ;(For SELECTOR operations,
000194                 STA       RNFLG                    ;  set flag that a program has been run)
000195                 JSR       CHRGOT                   ;WHAT WAS LAST CHAR?
000196                 BEQ       DATSALL                  ;IF A TERMINATOR, OK
000197                 JSR       CHKCOM                   ;MUST HAVE COMMA IF NOT TERMINATOR
000198                 JMP       LINGET                   ;GET THE LINE NUMBER
000199 DATSALL:        LDA       #0                       ;NO LINE SPECIFIED
000200                 STA       LINNUM
000201                 STA       LINNUM+1
000202                 RTS
000203                 PAGE
000204 *
000205 * HERE IS WHAT YOU'VE BEEN WAITING FOR ALL THIS LISTING!!!
000206 *    READ AND WRITE !!!!!
000207 *
000208 DREAD:          SEC
000209                 ROR       IOFLG
000210                 JSR       FILNUM                   ;GET FILE NUMBER AND REC# IF SPECIFIED
000211                 JSR       PREBIN
000212                 JSR       TSTIN                    ;TEST IF A BINARY FILE AND IN INPUT MODE-
000213                 JSR       CHRGOT
000214                 CMP       #$3B
000215                 BNE       RDRTS
000216                 JSR       CHRGET                   ;EAT THE SEMICOLON
000217                 BNE       DRD2                     ;ALWAYS
000218 RDRTS           RTS
000219 DRD1            JSR       CHRGOT                   ;MORE VARS?
000220                 BEQ       RDRTS
000221                 JSR       CHKCOM                   ;SEP'D BY COMMAS
000222 DRD2            JSR       GETNDX                   ;GET PTR INTO FILE BUFFER
000223                 LDA       (NDXPTR),Y               ;GET DESCRIPTOR
000224                 BNE       DRD3                     ;IF NOT AN END-OF-RECORD
000225                 JSR       DRDNXT                   ;READ IN THE NEXT RECORD IF NON-EMPTY.
000226                 JMP       DRD2
000227 ODERR           JMP       CHKEOF
000228 *
000229 * WE HIT AN END OF RECORD MARK. GO TO THE NEXT RECORD IF WE'RE NOT
000230 * AT THE END OF THIS ONE.
000231 DRDNXT          LDY       FCBNDX                   ;IF BUFOFS=0000, GIVE 'OUT-OF-DATA'
```

```
000232                LDA       FCB+XBUFOFS,Y
000233                ORA       FCB+XBUFOFS+1,Y
000234                BEQ       ODERR
000235                JMP       NXRCD                   ;GO TO THE NEXT RECORD
000236 DRD3            JSR       GETVAL                  ;TYPE OF VAR IN THE FILE
000237                STA       VALTYP
000238                BMI       RDSTRI                  ;READ A STRING
000239                PHA
000240                JSR       MYPTRGET
000241                BIT       VALTYP                  ;WHAT DOES HE WANT?
000242                BMI       MSMTCH
000243                JSR       NTINT3
000244                TAY                               ;SAVE DSC OF WHAT HE WANTS
000245                PLA                               ;GET TYPE OF DATA WE HAVE
000246                CMP       #VINT
000247                CLC                               ;ASSUM NOT INT
000248                BNE       *+3
000249                SEC
000250                ROR       INTFLG
000251                STA       VALTYP
000252                TYA
000253                PHA                               ;SAVE VALTYP OF VAR READING INTO
000254                LDA       NDXPTR
000255                LDY       NDXPTR+1
000256                LDX       NDXPTRB
000257                CLC
000258                ADC       #1
000259                BCC       *+3
000260                INY
000261                JSR       ISVRET2                 ;UNPACK VAL INTO FAC
000262                PLA                               ;SEE WHAT THE USER IS RDING INTO
000263                PHA
000264                CMP       #DDLNT                  ;LNG INT?
000265                BNE       NTLNT
000266                JSR       CONV2LNG
000267 DRD5            PLA
000268                JSR       GETVAL
000269                STA       VALTYP
000270                LSR       A
000271                LSR       A
000272                LSR       A
000273                LSR       A                       ;GET INT FLG INTO CARRY
000274                ROR       INTFLG
000275                JSR       LETP2                   ;PUT IT IN THE VAR
000276 DRD6            JSR       GETNDX
000277                LDA       (NDXPTR),Y
000278                STA       DSCRPT                  ;SAVE THE DESCRIPTOR
000279                JSR       CALCLEN
000280                JSR       UPOFS
000281                CMP       FCB+XRECL+1,Y           ;AT THE END OF THE BUFFER?
000282                BCC       DRD7                    ;NO.
000283                LDA       FCB+XBUFOFS,Y           ;MAYBE, CHECK LOW BYTE.
000284                CMP       FCB+XRECL,Y
000285                BCC       DRD7
000286                JSR       DRDNXT                  ;YES, READ THE NEXT BUF.
000287 DRD7            JMP       DRD1                    ;LOOP.
000288 NTLNT:          JSR       CONV2FLT
000289                JMP       DRD5
000290 MSMTCH          JMP       ERTYP
000291 MYPTRGET        JSR       PTRGET                  ;SAME AS PTRGET,
000292                STA       FORPNT                  ;BUT PUTS PTRS IN FORPNT ALSO
000293                STY       FORPNT+1
000294                LDX       VARPNTB
000295                STX       FORPNTB
000296                RTS
000297 *
000298 * WAS A STRING
000299 *
000300 RDSTRI          JSR       MYPTRGET
000301                BIT       VALTYP                  ;ANYTHING ELSE ILLEGAL
000302                BPL       MSMTCH
000303                LDY       #0
000304                LDA       (NDXPTR),Y              ;GET DESCRIPTOR
000305                INY
000306                CMP       #DDMXSTR                ;255-CHAR STRING?
000307                BNE       NTMXSTR
000308                LDA       #$FF
000309                DEY                               ;STR LEN=255
000310                DFB       44
000311 NTMXSTR         LDA       (NDXPTR),Y              ;GET STR LEN
```

```
000312                PHA                           ;SAVE THE LEN
000313                INY                           ;COPY CHARS FROM STRING
000314                TYA
000315                CLC
000316                ADC       NDXPTR              ;UPDATES NDXPTR
000317                STA       STRNG1
000318                LDA       NDXPTR+1
000319                ADC       #0
000320                STA       STRNG1+1
000321                LDA       NDXPTRB
000322                STA       STRNG1B
000323                PLA
000324                TAY
000325                JSR       STRFI1
000326                JSR       INPCOM
000327                JMP       DRD6
000328 *
000329 * HERE IS THE 'WRITE' OPERATION---
000330 *
000331 DWRITE:        LSR       IOFLG               ;SPECIFY A WRITE OP.
000332                JSR       FILNUM              ;GET FILE # AND REC #S
000333                JSR       PREBIN              ;MAKE SURE A BINARY FILE
000334                JSR       TSTOUT              ;WITH OUTPUT ALLOWED
000335                JSR       CHRGOT
000336                CMP       #$3B
000337                BNE       WRRTS
000338                JSR       CHRGET              ;EAT THE SEMICOLON
000339                BNE       DWR2                ;ALWAYS
000340 WRRTS:         RTS
000341 DWR1:          JSR       CHRGOT              ;ANY MORE VARS TO WRITE?
000342                BEQ       WRRTS
000343                JSR       CHKCOM              ;VARS MUST BE SEPD BY COMMAS
000344 DWR2           JSR       GETXPR              ;GET AN EXPRESSION TO WRITE
000345 DWR3           LDX       FCBNDX
000346                LDA       FCB+XBUFOFS,X       ;GOT AN EMPTY BUFFER?
000347                ORA       FCB+XBUFOFS+1,X
000348                BNE       DWR4
000349                JSR       POSREC
000350                JSR       SETPARMS            ;YES, EMPTY BUFFER - WRITE IT
000351                LDY       #WRT                ; TO RESERVE THE DISK SPACE.
000352                JSR       SETGO
000353                JSR       POSREC              ;REPOSITION.
000354 DWR4           JSR       GETNDX              ;GET NDXPTR TO DATA
000355                JSR       CALCLEN             ;CALC LEN OF THIS ITEM
000356                JSR       TSTFIT              ;WILL IT FIT HERE?
000357                BCS       ITFITS              ;YES.
000358 * SEE IF DATA ITEM WILL FIT IN AN EMPTY RECORD
000359                LDY       FCBNDX
000360                LDA       FCB+XRECL+1,Y
000361                BNE       NXREC               ;IF A RECLEN>255, YES FOR SURE
000362                LDA       LENSAV
000363                BEQ       DNTFIT              ;IF A 256-BYTE STRING, SORRY CHARLIE
000364                LDA       FCB+XRECL,Y
000365                CMP       LENSAV              ;ENOUGH ROOM IN THE RECORD
000366                BCC       DNTFIT              ;NOPE.
000367 NXREC          JSR       NXRCD               ;GO TO THE NEXT RECORD
000368                JMP       DWR3                ;AND LOOP
000369 DNTFIT         JMP       ODERR
000370 * MOVE IT INTO THE DATA BUFFER
000371 ITFITS:        LDY       #0
000372                LDX       LENSAV
000373                DEX
000374                STX       XSAV
000375                LDA       DSCRPT              ;GET THE DESCRIPTOR
000376                STA       (NDXPTR),Y          ;PUT IT INTO THE BUFFER
000377                INY
000378                DEX
000379                CMP       #DDSTR              ;IF A LONG STRING, DO SPECIAL
000380                BNE       ITF1
000381                STX       XSAV                ;PUT THE STRING LEN OUT
000382                TXA
000383                STA       (NDXPTR),Y          ;SAVE LEN OF STRING
000384                BEQ       DATDON              ;IF A NULL STRING
000385                INY
000386 ITF1           LDX       #0                  ;FOR ABS INDIRECT
000387                LDA       (VARPNT,X)
000388                STA       (NDXPTR),Y          ;MOVE IT INTO THE FILE BUFFER
000389                INY
000390                INC       VARPNT
000391                BNE       *+4
```

```
000312                ADC       #$0                     ; increment the program size by bank
000313                STA       BCDSTR+4                ; and resave it.
000314                DEY                               ;One less bank in counter.
000315                JMP       PGMLEN3
000316 SZRTS          RTS                               ;All done.  Let's go back!
000317 ;
000318 ;   This routine does a NUMBYTES subtract
000319 ;
000320 DOSUB          LDY       #0                      ;Start loop at 0
000321                SEC
000322                PHP                               ;Dummy Push to secure Stack
000323 LOOP           EQU       *
000324                PLP                               ;Pull Status off stack
000325                LDA       BCDSTR+2,Y
000326                SBC       BYTBLOCK,Y
000327                STA       BCDSTR+2,Y
000328                PHP                               ;Save status of calculation
000329                INY                               ;Increment loop
000330                CPY       #4                      ;At limit?
000331                BNE       LOOP                    ;No, do next byte
000332                PLP                               ;Clean up stack
000333                RTS
000334 ;
000335 ; Routine to Get the Record Length (Default is 512)
000336 GRECLN         LDA       #0                      ;DEFAULT REC LEN OF 512
000337                STA       INAUXID
000338                LDA       #2
000339                STA       INAUXID+1
000340                JSR       CHRGOT                  ;IS THERE A RECLEN SPECIFIED?
000341                BEQ       GRCRTS
000342                JSR       CHKCOM                  ;MUST HAVE A COMMA
000343                JSR       FRMNUM
000344                JSR       POSINT
000345                LDX       FACLO                   ;RECLEN MUST BE >3, <32767
000346                LDA       FACMO
000347                BMI       BUMSIZ
000348                BNE       GRC2
000349                CPX       #3
000350                BCS       GRC2
000351 BUMSIZ         JMP       FCERR
000352 GRC2           STA       INAUXID+1
000353                STX       INAUXID
000354 GRCRTS         RTS
000355 WRTRCD2        JSR       WRTRCD
000356                BNE       *+5
000357 IDNUMR         LDA       #0
000358                RTS
000359                JMP       SERROR
000360 WRTRCD         LDY       FCBNDX                  ;SHOULD WE WRITE THIS?
000361                LDA       FCB+XFLGS,Y
000362                BPL       IDNUMR
000363                LDA       FCB+XBUFOFS+1,Y
000364                PHA
000365                LDA       FCB+XBUFOFS,Y
000366                PHA
000367                ORA       FCB+XBUFOFS+1,Y         ;ANYTHING TO WRITE?
000368                BEQ       IDNUM2                  ;NO, QUIT IT.
000369                JSR       TSTFTX                  ;DO WE HAVE A TRAILING 0 ON THIS RECORD?
000370                PHP                               ;CARRY SET IF SO.
000371                JSR       POSREC                  ;DO POSITION
000372                JSR       SETPARMS
000373                PLP
000374 IDNUM2         PLA
000375                ADC       #0                      ;ADD 1 IF EXTRA NULL TO WRITE.
000376                STA       INBYTES
000377                PLA
000378                ADC       #0
000379                STA       INBYTES+1
000380                LDY       #WRT                    ;AND WRITE THE DATA
000381                JSR       SETUP
000382                JMP       GOSOS
000383 NXRCD:         JSR       WRTRCD2                 ;WRITE IF NEC.
000384                LDX       FCBNDX                  ;INCREMENT RECORD NUMBER
000385                INC       FCB+XRNUM,X
000386                BNE       NXR2
000387                INC       FCB+XRNUM+1,X
000388 NXR2:          JMP       RPOSN
000389 SETPARMS       LDA       #0
000390                LDY       FCBNDX
000391                STA       FCB+XBUFOFS,Y
```

```
000392                STA       FCB+XBUFOFS+1,Y
000393                JSR       GETNDX
000394                LDA       #>NDXPTR
000395                STA       SBFPTR
000396                LDA       #<NDXPTR
000397                STA       SBFPTR+1
000398                LDY       FCBNDX
000399                LDA       FCB+XRECL,Y
000400                STA       INBYTES
000401                LDA       FCB+XRECL+1,Y
000402                STA       INBYTES+1
000403                LDA       FCB+XRFNM,Y
000404                STA       RWRFNM
000405                RTS
000406 *
000407 * MISC ROUTINES
000408 *
000409 SERROR:        LDX       #0
000410 FNDAER1        CMP       ERRTBL,X
000411                BEQ       FOUNAR1
000412                INX
000413                INX
000414                BCS       FNDAER1
000415                LDX       #SSSSSS
000416                STA       SOSLOC
000417                JMP       ERROR
000418 FOUNAR1        INX
000419                LDA       ERRTBL,X
000420                TAX
000421                JMP       ERROR
000422 GETFISET:      LDA       #>FATRB                 ;GET INFO ON THIS FILE.
000423                STA       FLSTPTR
000424                LDA       #<FATRB
000425                STA       FLSTPTR+1
000426                LDA       #$B                     ;FIND OUT EVERYTHING
000427                STA       INLNGTH
000428                LDY       #GFI
000429                RTS
000430 SETDSP:        LDY       FCBNDX                  ;SET UP DISPLACEMENT FOR REC
000431                LDA       #0                      ;SPECIFY AT BEGINNING OF RECORD
000432                STA       FCB+XBUFOFS,Y
000433                STA       FCB+XBUFOFS+1,Y
000434                LDA       FCB+XRNUM,Y             ;POSITION TO RNUM*RECLEN
000435                STA       MLTPLR
000436                LDA       FCB+XRNUM+1,Y
000437                STA       MLTPLR+1
000438                LDA       FCB+XRECL,Y
000439                STA       MLTPLR2
000440                LDA       FCB+XRECL+1,Y
000441                STA       MLTPLR2+1
000442                JSR       MUL                     ;GET BYTE # IN FILE
000443                LDY       #4
000444 QTIP:          LDA       RSLT-1,Y                ;MOVE RESULT TO DSPLMNT
000445                STA       DSPLMNT-1,Y
000446                DEY
000447                BNE       QTIP
000448                STY       BASE                    ;MAKE IT RELATIVE TO BEGINNING OF FILE
000449 PPRTS          RTS
000450 POSREC         JSR       SETDSP                  ;POSITION TO THIS REC
000451                LDY       #STM
000452                JSR       SETUP
000453                JSR       GOSOS                   ;DO THE SET.MARK
000454                BEQ       PPRTS                   ;EVERYTHING IS OK
000455                CMP       #SENBK                  ;IF NOT A BLOCK DEV,
000456                BEQ       PPRTS                   ;DON'T DO IT!
000457                CMP       #$4D
000458                BEQ       *+5
000459                JMP       SERROR
000460 *WANT TO SET MARK BEYOND EOF
000461                BIT       IOFLG
000462                BMI       PSRERR
000463                JSR       TSTOUT                  ;TEST IF WE CAN FIRST
000464                LDX       FCBNDX                  ;GO TO NEXT REC
000465                INC       FCB+XRNUM,X
000466                BNE       *+5
000467                INC       FCB+XRNUM+1,X
000468                JSR       SETDSP
000469                LDY       #STE                    ;SET.EOF
000470                JSR       SETGO
000471                LDX       FCBNDX
```

```
000472                LDA       FCB+XRNUM,X
000473                BNE       *+5
000474                DEC       FCB+XRNUM+1,X
000475                DEC       FCB+XRNUM,X
000476                JMP       POSREC
000477 PSRERR          JSR       GETNDX
000478                TYA                                 ;A=0.
000479                STA       (NDXPTR),Y
000480                JMP       CHKEOF
000481
000482 ; ##############################################################################################
000483 ; #   END OF FILE:  FILESTUF.TEXT
000484 ; #   LINES      :  475
000485 ; #   CHARACTERS :  21734
000486 ; ##############################################################################################

+------------------------------------------------------------------------------
|
|  THAT'S ALL FOLKS!      LINES: 486   CHARACTERS: 22288
|
+------------------------------------------------------------------------------
```

```
000232                BEQ       NMREOF
000233                INY
000234                BNE       OEOF
000235 NMREOF         LDY       CTPL+$13
000236                LDA       CTPL+$14
000237                JSR       GIVAYF
000238                JSR       FOUT
000239                LDY       #$FF
000240 CNTLN          INY
000241                LDA       FBUFFR+1,Y              ;MOVE BLOCK COUNT IN
000242                BNE       CNTLN                   ;COUNT #OF DIGS
000243                LDX       #5                      ;5 DIGIT COUNT
000244 MVBCNT         LDA       FBUFFR,Y                ;GET DIG
000245                STA       CATBUF+8,X
000246                DEX
000247                DEY
000248                BPL       MVBCNT
000249                LDA       #'0'                    ;FILL REST OF COUNT WITH 0S
000250 ZBC:           DEX
000251                BMI       CMP3
000252                STA       CATBUF+9,X
000253                BPL       ZBC
000254 * NOW PUT IN THE DATES
000255 CMP3:          LDA       CTPL+$22                ;GET DATE LAST MODIFIED
000256                LDX       CTPL+$21
000257                LDY       #31
000258                JSR       GENDATE
000259                LDA       CTPL+$19
000260                LDX       CTPL+$18
000261                LDY       #46
000262                JSR       GENDATE                 ;CREATE DATE.
000263                LDA       CTPL+$24                ;GIVE TIME LAST MOD'D
000264                LDX       CTPL+$23
000265                LDY       #40
000266                JSR       GENTIME
000267                LDA       CTPL+$1B
000268                LDX       CTPL+$1A                ;GIVE TIME LAST CREATED
000269                LDY       #55
000270                JSR       GENTIME
000271                CLC
000272                RTS
000273 *
000274 * PRINT THE SUMMING UP INFO, IF POSSIBLE.
000275 *
000276 CEND           LDA       FCB+XRECL,X             ;IS IT A ROOT DIR W/INFO?
000277                ORA       FCB+XRECL+1,X
000278                BEQ       CEND2
000279                LDY       #SUMML                  ;SUMMING MSG LEN.
000280 CEN2:          LDA       SMMSG-1,Y
000281                STA       CATBUF,Y
000282                DEY
000283                BNE       CEN2
000284                LDA       FCB+XRECL,X             ;GET LOW OF TOTAL BLOCKS
000285                TAY
000286                LDA       FCB+XRECL+1,X
000287                LDX       #53
000288                JSR       DONUM
000289                LDX       FCBNDX
000290                LDA       FCB+XBLKS,X
000291                TAY
000292                LDA       FCB+XBLKS+1,X
000293                LDX       #33
000294                JSR       DONUM
000295                LDX       FCBNDX
000296                LDA       FCB+XRECL,X
000297                SEC
000298                SBC       FCB+XBLKS,X
000299                TAY
000300                LDA       FCB+XRECL+1,X
000301                SBC       FCB+XBLKS+1,X
000302                LDX       #14
000303                JSR       DONUM
000304                LDX       FCBNDX
000305                DEC       FCB+XFLGS,X
000306 CEND2:         INC       FCB+XFLGS,X
000307                JMP       CBL
000308 FRNTYP         ASC       "TYP=  "
000309 PROTYP         ASC       "PRO=  "
000310 TYPTB:         ASC       "UNKNWN"
000311                ASC       "BAD   " ;6 CHARS EACH"
```

```
000312                ASC       "PASCOD"
000313                ASC       "PASTXT"
000314                ASC       "TEXT  "
000315                ASC       "PASDTA"
000316                ASC       "BINARY"
000317                ASC       "FONT  "
000318                ASC       "FOTO  "
000319                ASC       "BASIC "
000320                ASC       "DATA  "
000321                ASC       "WPTEXT"
000322                ASC       "SYSTEM"
000323                ASC       "RESERV"
000324                ASC       "RESERV"
000325                ASC       "CAT   "
000326                ASC       "RPSDAT"
000327                ASC       "RPSIDX"
000328                ASC       "AFDISC"
000329                ASC       "ASMOD " "
000330                ASC       "AFRPT "
000331                ASC       "SCNLIB"
000332 *
000333 * ROUTINE TO READ CATALOG INFO FROM SOS
000334 *
000335 RDCAT           LDY       FCBNDX
000336                LDA       FCB,Y                     ;CAT REF NUM
000337                STA       RWRFNM
000338                LDA       FCB+XBUFOFS,Y
000339                CMP       #$FF                      ;END OF ONE DIR BLOCK?
000340                BNE       RDCAT3                    ;NO.
000341                LDA       #5                        ;YES, SKIP JUNK IN BETWEEN DIR BLOCKS
000342                JSR       RDCAT2
000343                BCS       NMCAT                     ;BRANCH IF NO MORE CATALOG
000344 RDCAT3         LDA       #$27
000345 RDCAT2         STA       INBYTES
000346                STA       LENSAV
000347                LDA       #0
000348                STA       INBYTES+1
000349                LDA       #>CTPL
000350                STA       SBFPTR
000351                LDA       #<CTPL
000352                STA       SBFPTR+1
000353                LDY       #RED                      ;READ WITH EOF CHECK
000354                JSR       SETUP
000355                JSR       GOSOS
000356                BEQ       OKCAT
000357                CMP       #SEEOF                    ;WAS IT AN END-OF-FILE ERROR?
000358                BNE       CTR2                      ;NO, BLOW UP
000359 NMCAT:         SEC
000360                RTS
000361 OKCAT:         JMP       UPOFS                     ;CLEAN UP STUFF
000362 CTR2:          JMP       CLTERR
000363                PAGE
000364 *
000365 * HERE IS ROUTINE TO GEN THE DATE FOR EVERY BODY UNDER THE SUN---
000366 *
000367 * ENTER A=HI,X=LO, Y=PT TO PUT DATE IN CATBUF
000368 *  DATE IS(FROM HI TO LO) 7 BITS YEAR, 4 BITS MONTH, 5 BITS DAY
000369 *
000370 GENDATE:       STA       CTPL+1                    ;PRESERVE STUFF
000371                STX       CTPL
000372                STY       CTPL+2
000373                TXA                                 ;COMPUTE DAY
000374                AND       #$1F
000375                TAY                                 ;G2DIGS EXPECTS Y REG
000376                LDA       #3                        ;3 CHAR POSITIONS INTO DATE
000377                JSR       G2DIGS
000378                LDA       CTPL+1                    ;NOW DO THE YEAR
000379                LSR       A
000380                ROR       CTPL                      ;PREP TO DO THE MONTH
000381                TAY
000382                LDA       #6
000383                JSR       G2DIGS
000384                LDA       CTPL                      ;GET THE MONTH
000385                LSR       A
000386                LSR       A
000387                LSR       A
000388                LSR       A
000389                TAY
000390                LDA       #0                        ;AT BEGINNING OF BUF
000391                JSR       G2DIGS
```

```
000392                  LDA       #'/'
000393                  STA       CATBUF+2,Y              ;Y-REG IS SET UP BY G2DIGS
000394                  STA       CATBUF+5,Y
000395                  RTS
000396 *
000397 * GIVE 2 DIGITS SOMEWHERE INTO THE CATBUF
000398 *
000399 G2DIGS           PHA
000400                  JSR       SNGFLT
000401                  JSR       FOUT
000402                  PLA
000403                  CLC                               ;CALC PLACE TO PUT DIGS
000404                  ADC       CTPL+2
000405                  TAY
000406                  LDA       FBUFFR                  ;GET DIG#1
000407                  STA       CATBUF,Y
000408                  STA       CATBUF+1,Y              ;PUT IT BOTH PLACES
000409                  LDA       FBUFFR+1                ;ANOTHER DIG?
000410                  BEQ       AZERH                   ;NO, ZERO FOR HIGH DIGIT
000411                  STA       CATBUF+1,Y
000412                  RTS
000413 AZERH            LDA       #'0'
000414                  STA       CATBUF,Y
000415                  RTS
000416 * ROUTINE TO GENERATE THE TIME FOR THE CATALOG
000417 GENTIME          STX       CTPL                    ;PRESERVE MINUTES
000418                  STY       CTPL+2                  ;POS IN THE LINE
000419                  TAY                               ;PUT HOURS INTO Y-REG
000420                  LDA       #0                      ;POSITION RELATIVE TO (CTPL+2)
000421                  JSR       G2DIGS
000422                  LDA       #$3A                    ;COLON IN THE TIME
000423                  STA       CATBUF+2,Y
000424                  LDY       CTPL
000425                  LDA       #3                      ;NOW PRINT MINUTES
000426                  JMP       G2DIGS
000427 *
000428 * ROUTINE TO OUTPUT A NUMBER TO CATBUF,X
000429 DONUM:           STX       XSAV
000430                  JSR       GIVAYF
000431                  JSR       FOUT
000432                  LDY       #0
000433                  LDX       XSAV                    ;MOVE # IN.
000434 DONUM2:          LDA       FBUFFR,Y
000435                  BEQ       DONUMR
000436                  STA       CATBUF,X
000437                  INY
000438                  INX
000439                  BNE       DONUM2                  ;ALWAYS
000440 DONUMR           RTS
000441 SMMSG:           ASC       'BLOCKS FREE:       BLOCKS USED:       TOTAL BLOCKS:      '
000442 SUMML            EQU       *-SMMSG
000443                  SBTL      "Subroutine Jump Table" "
000444 JMPRTAB          DW        DOPAR
000445                  DW        PTRGET
000446                  DW        MVUP
000447                  DW        MVDWN
000448                  DW        BLTUC
000449                  DW        ERRDIR
000450                  DW        LINGET
000451                  DW        GOTOB
000452                  DW        GETADR
000453                  DW        FNDLNC0
000454                  DW        FNDLIN
000455                  DW        INITCNS
000456                  DW        RESLST-$2000
000457                  DW        NOTNOW
000458                  DW        ERROR
000459                  DW        SERROR
000460                  DW        SCRUNCH
000461                  DW        EXPAND
000462                  DW        FREFAC
000463                  DW        FRENOW
000464                  DW        FRECNOW
000465                  DW        FRESPA
000466                  DW        OPENIT
000467                  DW        GOSOS
000468                  DW        CLSALL
000469                  DW        GIVAYF
000470                  DW        POSINT
000471                  DW        FIN
```

```
000472                  DW        NWSTT
000473                  DW        PTRGT3
000474                  DW        PNTREL                    ;30
000475                  DW        RELPTR2                   ;31
000476                  DW        DATAN
000477                  DW        STRCP
000478                  DW        INPCOM
000479                  DW        LETP2                     ;35
000480                  DW        FOUT                      ;36
000481                  DW        NEWRET
000482                  DW        JUMPDO
000483                  DW        INT
000484                  DW        FBUFFR-$2000
000485                  DW        RESL2-$2000
000486                  DW        SETUP
000487                  DW        SETGO
000488                  DW        CONV2STR
000489                  DW        JMPRTAB
000490
000491 ; ##############################################################################################
000492 ; #   END OF FILE:  CATALOG.TEXT
000493 ; #   LINES      :  484
000494 ; #   CHARACTERS :  19482
000495 ; ##############################################################################################

+------------------------------------------------------------------------------
|
|  THAT'S ALL FOLKS!      LINES: 495   CHARACTERS: 20034
|
+------------------------------------------------------------------------------
```

```
000388              popcode  := trim_leading(popcode );   popcode  := trim_trailing(popcode );
000389              poperand := trim_leading(poperand);   poperand := trim_trailing(poperand);
000390              pcomment := trim_leading(pcomment);   pcomment := trim_trailing(pcomment);
000391
000392              if (plabel<>'') or (popcode<>'') or (poperand<>'') or (pcomment<>'') then begin
000393                {
000394                writeln(diagnostic,'LABEL   = "',plabel,'"');
000395                writeln(diagnostic,'OPCODE  = "',popcode,'"');
000396                writeln(diagnostic,'OPERAND = "',poperand,'"');
000397                writeln(diagnostic,'COMMENT = "',pcomment,'"');
000398                }
000399
000400                if (plabel='') and (popcode='') then
000401                  s := pcomment
000402                else begin
000403                  while length(plabel )<kWidth_Label    do plabel   := concat(plabel  ,kSpace);
000404                  while length(popcode)<kWidth_Opcode   do popcode  := concat(popcode ,kSpace);
000405                  while length(poperand)<kWidth_Operand do poperand := concat(poperand,kSpace);
000406                  s := concat(plabel,kSpace,popcode,kSpace,poperand,kSpace,pcomment);
000407                end;
000408
000409                s := trim_trailing(s);
000410
000411                if nobrkspace then s := use_nobreak_spaces(s);
000412
000413                cnt_chars := cnt_chars + length(s);
000414                writeln(argg,s);
000415              end else begin
000416                cnt_chars := cnt_chars + 1;
000417                writeln(argg);
000418              end;
000419            end;
000420          end; {while}
000421          {++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}
000422
000423          write_info('',false);
000424
000425          close(argg);
000426        end;
000427
000428        close(argf);
000429      end;
000430
000431      argi := argi + 1;
000432    end; {while}
000433  end;
000434
000435  writeln(diagnostic);
000436  writeln(diagnostic,'That''s all folks!');
000437 end.
000438
000439 { finis }

+------------------------------------------------------------------------------
|
|  THAT'S ALL FOLKS!      LINES: 439   CHARACTERS: 13956
|
+------------------------------------------------------------------------------

###
```