==========

TASC DOCUMENTATION

ENTERED BY: JUDIE MAC

SYSTEM REQUIR:APPLE II OR APPLE II+ 48K RAM

THE APPLESOFT COMPILER:
  TO EXTEND APPLESOFT LANGUAGE
  TO COMPLEMENT APPLESOFT INTERPRETER

1.
INCREASE EXECUTION SPEED COMPILED WITH TASC = 2 TO 20 TIMES
FASTER THAN W/INTERPRETER

2.
INTER-PROGRAM COMMUNICATON=COMMUNICATE PGMS WITH EACH OTHER
BY COMMON VARIABLES

3.
TRUE INTEGER ARITHMETIC

4.
SOURCE-CODE SECURITY=TASC CREATES MACHINE LANGUAGE EQUIVALENTS
OF APPLESOFT BASIC PGMS

5.
DISK-BASED COMPILATION

FILES ON DISK:
1 - TASC- APPLESOFT COMPILER
2 - PASS0,PASS1,PASS2- INTERNAL COMPONENTS OF TASC

3 - RUNTIME- LIBRARY OF MACHINE LANGUAGE ROUTINES
4 - ADR - UTILITY FOR BINARY FILES
5 - CREATE ADR - UTILITY FOR CREATING ADR ON OTHER DISKS
6 - BALL - DEMO PGM

2:DEMO RUN:
TASC IS SIMPLE TO USE.TO INVOKE THE COMPILER, FIRST BOOT UP YOUR DISK AND THEN TYPE:

]RUN TASC

THE NEXT TWO PROMPTS ASK YOU FOR THE NAMES OF THE SOURCE AND OBJECT FILES:

]SOURCE FILE BALL
]OBJECT CODE FILE: (DEFAULT BALL.OBJ) <RETURN>

THE SOURCE FILE IS A APLSOFT PGM NAMED BALL THAT EXISTS ON THE DISK. THE OBJECT FILE IS THE MACHINE LANGUAGE BINARY FILE THAT IS CREATED BY THE COMPILER. THE OBJECT FILE NAME DEFAULTS TO THE ORIGINAL FILE NAME WITH THE EXTENSION .OBJ ADDED, SO THAT THE OBJECT FILE PRODUCED FOR THE BALL.OBJ. THE DEFAULT IS SPECIFIED BY ENTERING <RETURN>. THE SOURCE FILE IS ASSUMED TO BE LOCATED ON THE SAME DISK AS THE COMPILER UNLESS YOU SPECIFY OTHERWISE. THE OBJECT FILE DEFAULTS TO THE SAME DISK THAT THE SOURCE IS ON. DIFFERENT SLOTS OR DRIVES CAN BE SPECIFIED USING THE NORMAL ,S<SLOT NUMBER> AND ,D <DRIVE NUMBER> SYNTAX. COMPILATION IS USUALLY SLIGHTLY FASTER IF ONLY ONE DRIVE IS USED. DISK COMMANDS CAN BE EXECUTED BY TYPING <CTRL-D> FOLLOWED BY THE COMMAND AND <RETURN> THE NEXT TWO PROMPTS ASK YOU WHETHER YOU WANT DEFAULT VALUES FOR ALL OTHER COMPILATION OPTIONS. SINCE MOST COMPILATIONS ARE PERFORMED WITH THE SAME SET OF OPTIONS, YOU SHOULD ENTER <RETURN> AFTER EACH PROMPT TO SPECIFY THE DEFAULT VALUES.

```
MEMORY USAGE:
DEFAULT CONFIGURATION <RETURN>
OPTIONS:
DEFAULT CONFIGURATION  <RETURN>
```

IF YOU HAD REFUSED THE DEFAULT CONFIGURATIONS ABOVE, YOU WOULD NEED TO EXPLICITLY SPECIFY THE VALUES OF SEVERAL COMPILATION OPTIONS. THE ACTUAL COMPILATION PROCESS STARTS WITHOUT FURTHER INPUT SINCE YOU HAVE SPECIFIED THE DEFAULTS ABOVE. WHEN COMPILATION BEGINS,THE DISK IS ACCESSED ALMOST CONSTANTLY TO EITHER READ THE SOURCE FILE OR TO WRITE THE OBJECT FILE. THE COMPILER LISTS THE SOURCE PROGRAM ON YOUR CONSOLE AS IT IS BEING COMPILED AND GENERATES APPROPRIATE MESSAGES IF IT ENCOUNTERS ANY ERRORS.THEN THE SOURCE STOPS LISTING,THE FIRST PARTS OF COMPILATION IS FINISHED, AND THE COMPILER PRINTS:

```
*****BEGINNING PASS2
```

THE SECOND PART OF COMPILATION ALSO USED THE ???? EXTENSIVELY. TO INDICATE THAT IT IS STILL COMPILING, THE COMPILER PRINTS A PERIOD ON THE SCREEN EVERY FEW SECONDS. WHEN IS IS FINISHED, THE COMPILER

PRINTS:

    *****CODE GENERATION COMPLETE

AT THIS POINT,THE ACTUAL COMPILATION PROCESS IS COMPLETE.SO THAT YOU WILL RECEIVE A LISTING OF COMPILATION INFORMATION, ANSWER "Y" OR "YES" TO THE NEXT PROMPT:
COMPILATION INFORMATION AND LINE NUMBER REFERENCE TABLE YES

THIS INPUT ALSO ACCEPTS <CRTL D> DISK COMMANDS. IF YOU WANT TO LIST THE COMPILATION INFORMATION ON A PRINTER, YOU CAN FIRST TURN ON YOUR PRINTER BY ENTERING:<CTRL-D)PR#(PRINTER SLOT>

TASC PRINTS OUT THE DESIRED INFORMATION, DISPLAYS THE FOLLOWING MESSAGE, AND THEN RE-ENTERS THE INTERPRETER:

    *****COMPILATION COMPLETE
    ]

THE INCREASE IN THE BALL PROGRAMS EXECUTION SPEED IS QUITE APPARENT WHEN COMPARED TO THE SAME PROGRAM RUNNING UNDER THE INTERPRETER. COMPARE SPEEDS BY FIRST RUNNING THE INTERPRETED PROGRAM:

    ]RUN BALL

NEXT EXECUTE THE COMPILED PROGRAM BY ENTERING THE FOLLOWING DOS COMMANDS:

    ]BLOAD RUNTIME
    ]BRUN BALL.OBJ

NOTE THAT THE RUNTIME LIBRARY MUST BE BLOADED IN MEMORY BEFORE BALL.OBJ AND CAN BE BRUN.

YOU HAVE NOW SUCCESSFULLY COMPLETED THE DEMO RUN.

COMPILATION ******VOCABULARY

SOURCE FILE -
THE APPLESOFT PROGRAM IS COMMONLY CALLED A SOURCE FILE BECAUSE IT IS THE SOURCE FROM WHICH AN EQUIVALENT MACHINE LANGUAGE FILE IS CREATED. THE SOURCE FILE IS THE INPUT FILE TO THE COMPILER. CATALOG LISTS THE NAMES OF APPLESOFT SOURCE FILES WITH THE LETTER "A" PRECEDING THE SIZE OF EACH FILE.

OBJECT FILE -
TASC TRANSLATES SOURCE FILES INTO MACHINE LANGUAGE OBJECT FILES. THE OBJECT FILE IS THE OUTPUT FILE CREATED BY THE COMPILER.THE OBJECT FILES IS AN EXECUTABLE BINARY FILE THAT IS THE MACHINE LANGUAGE EQUIVALENT OF THE SOURCE. CATALOG LISTS THE NAMES OF TASC OBJECT FILES WITH THE LETTER "B" PRECEDING THE SIZE OF THE FILE.

COMPILETIME-
THE TIME DURING WHICH THE COMPILER IS TRANSLATING A SOURCE FILE INTO

AN OBJECT FILE.

RUNTIME -
THE TIME DURING WHICH A COMPILED PROGRAM IS EXECUTING. BY CONVENTION, RUNTIME REFERS TO THE EXECUTION TIME OF A COMPILED PROGRAM AND NOT TO THE EXECUTION TIME OF THE COMPILER.

RUNTIME LIBRARY -
A COLLECTION OF MACHINE LANGUAGE ROUTINES THAT ARE USED BY COMPILED OBJECT PROGRAMS. THESE ROUTINES ALL RESIDE IN THE FILE NAMES RUNTIME. RUN TIME MUST BE LOADED INTO MEMORY BEFORE AN OBJECT FILE CAN BE EXECUTED.

COMPILATION VS INTERPRETATION:

SINCE THE MICROPROCESSOR IN THE APPLE CAN EXECUTE ONLY ITS OWN MACHINE INSTRUCTIONS, IT DOES NOT EXECUTE APPLESOFT PROGRAM STATEMENTS DIRECTLY.INSTEAD,STATEMENTS MUST BE SIMULATED BY MACHINE LANGUAGE ROUTINES THAT PERFORM THE OPERATIONS SPECIFIED BY THE BASIC STATEMENT.

INTERPRETATION:
THE INTERPRETER TRANSLATES APPLESOFT SOURCE STATEMENTS LINE BY LINE AT RUNTIME. EACH TIME THE INTERPRETER EXECUTES AN APPLESOFT STATEMENT, IT MUST ANALYZE THE STATEMENT, CHECK FOR ERRORS AND CALL MACHINE LANGUAGE ROUTINES THAT PERFORM THE DESIRED FUNCTION. WHEN STATEMENTS MUST BE EXECUTED REPEATEDLY, AS MUST THOSE WITH A FOR/NEXT LOOP, THE TRANSLATION PROCESS MUST BE REPEATED EACH TIME THE STATEMENT IS EXECUTED. IN ADDITION,BASIC LINE NUMBERS ARE STORED IN A LIST. GOTO'S AND GOSUB'S FORCE THE INTERPRETER TO SEARCH THIS LIST TO FIND THE DESIRED LINE.THIS SEARCH IS QUITE SLOW WHEN THE NEEDED LINE IS NEAR THE END OF A LONG PROGRAM. THE INTERPRETER KEEPS TRACK OF VARIABLES USING A LIST,TOO. WHEN IT ENCOUNTERS A REFERENCE TO A VARIABLE, THE INTERPRETER SEARCHES FROM THE BEGINNING OF THE LIST TO FIND THE DESIRED VARIABLE. IF THE VARIABLE IS NOT PRESENT IN THE LIST, THE INTERPRETER MUST CREATE A NEW ENTRY FOR IT. THIS PROCEDURE ALSO SLOWS INTERPRETED PROGRAMS.

COMPILATION:
A COMPILER, ON THE OTHER HAND, TAKES A SOURCE PROGRAM AND TRANSLATES IT INTO A MACHINE LANGUAGE OBJECT FILE. THIS OBJECT FILE CONSISTS OF A LARGE NUMBER OF MACHINE LANGUAGE CALL'S TO ROUTINES IN THE RUNTIME LIBRARY AND TO ROUTINES IN THE APPLESOFT INTERPRETER, TASC ASSURES CLOSE LANGUAGE COMPATIBILITY WITH THE INTERPRETER. IN CONTRAST TO THE INTERPRETER, THE COMPILER ANALYZES ALL STATEMENTS BEFORE RUNTIME. IN ADDITION,ABSOLUTE MEMORY ADDRESSES ARE PROVIDED FOR VARIABLES AND PROGRAM LINES.THESE ADDRESSES ELIMINATE THE LIST SEARCHING THAT OCCURS WHILE AN INTERPRETED PROGRAM EXECUTES. TASC,UNLIKE THE INTERPRETER, IMPLEMENTS TRUE INTEGER ARITHMETIC AND INTEGER LOOP VARIABLES IN FOR/NEXT LOOPS.IN COMPARISON, THE APPLESOFT INTERPRETER CONVERTS ALL INTEGERS TO REAL NUMBERS BEFORE OPERATING ON THEM. THESE CONVERSIONS MAKE INTERPRETED INTEGER ARITHMETIC RELATIVELY INEFFICIENT. IN ADDITION,THE INTERPRETER FORBIDS USE OF INTEGERS AS LOOP CONTROLVARIABLES IN FOR/NEXT LOOPS.

PROGRAM DEVELOPMENT

1 CREATE AND EDIT APPLESOFT SOURCE
2 RUN AND DEBUG SOURCE WITH THE INTERPRETER
3 COMPILE SOURCE,CREATING A BINARY OBJECT FILE
4 EXECUTE COMPILED OBJECT FILE

DEBUGGING WITH THE APPLESOFT INTERPRETER***

DEBUGGING A PROGRAM INTENDED FOR COMPILATION IS A TWO STEP PROCESS THAT INVOLVES:

1.CREATING THE SOURCE PROGRAM,AND
2.RUNNING THE PROGRAM UNDER THE INTERPRETER TO CHECK FOR ERRORS.

CREATING A SOURCE PROGRAM:

AN APPLESOFT SOURCE PROGRAM REQUIRES THE USE OF THE EDITOR AVAILABLE WITHIN APPLESOFT. PROGRAMS ARE CREATED BY SIMPLY ENTERING APPLESOFT STATEMENTS FROM WITHIN APPLESOFT. ONCE A PROGRAM HAS BEEN CREATED, IT CAN BE SAVED TO DISK WITH SAVE. TASC CAN ONLY COMPILE APPLE SOFT DISK FILES.

RUNNING A PROGRAM WITH APPLESOFT:

PROGRAMS SHOULD BE DEBUGGED USING THE APPLESOFT INTERPRETER BEFORE BEING COMPILED. IF THE PROGRAM TO BE COMPILED USES TASC FEATURES THAT ARE NOT AVAILABLE IN THE INTERPRETER, IT MAY BE NECESSARY TO DEBUG THE PROGRAM WITH THE COMPILER.

TASC IS HIGHLY COMPATIBLE WITH THE APPLESOFT INTERPRETER.THIS COMPATIBILITY ALLOWS THE APPLESOFT INTERPRETER TO FUNCTION AS THE PRIMARY DEBUGGING TOOL. THE INTERPRETER PROVIDES MUCH BETTER DEBUGGING FACILITIES THAN A COMPILER,SINCE IT INCLUDES FEATURES SUCH AS TRACE.

THERE ARE SOME DRAWBACKS TO DEBUGGING WITH THE INTERPRETER: STATEMENTS THAT ARE ONLY EXECUTED UNDER SPECIAL CIRCUMSTANCES MAY NEVER BE EXAMINED, AND THE INTERPRETER HALTS EXECUTION WHEN IT ENCOUNTERS THE FIRST ERROR IN A PROGRAM.

DEBUGGING WITH THE COMPILER DOES NOT SUFFER FROM THESE DRAWBACKS SINCE THE COMPILER EXAMINES EVERY STATEMENT IN A PROGRAM, AND CAN CONTINUE THE COMPILATION EVEN IF IT ENCOUNTERS ERRORS.

IN GENERAL,COMPILING A PROGRAM IS AN EFFECTIVE WAY TO CHECK FOR SYNTAX ERRORS; HOWEVER,PROGRAM LOGIC ERRORS ARE MORE EASILY TRACKED DOWN WITH THE INTERPRETER.

COMPILATION******

NOTE**

IF A COMPILED PROGRAM DOES NOT RUN CORRECTLY, SEE ERROR MESSAGES AND DEBUGGING FOR SOME POSSIBLE SOLUTIONS.

OPTIONS:

THE DEMO RUN SHOWED ONLY THE MOST BASIC TYPE OF COMPILATION. TASC INCLUDES SEVERAL OPTIONS THAT CAN BE USED TO CONTROL COMPILATION MORE CLOSELY.THE REQUESTED OPTIONS CONTROL MEMORY ALLOCATION AND COMPILATION.TO EXPLICITLY SPECIFY THE VALUES FOR THESE OPTIONS,SIMPLY ANSWER "NO" WHEN THE COMPILER OFFERS THE DEFAULT VALUES.

MEMORY USAGE:

THE MEMORY USED BY THE COMPILED CODE AT RUNTIME IS DIVIDED INTO THREE AREAS:

1.RUNTIME
1 LIBRARY
2.OBJECT PROGRAM
3.VARIABLES

TASC ALLOWS THE LOCATION FOR EACH OF THESE BLOCKS TO BE SPECIFIED SEPARATELY. THE MEMORY ALLOCATION FEATURES CAN BE USED TO PROTECT MACHINE LANGUAGE PROGRAMS, SHAPE TABLES, THE HIRES SCREENS, OR ANY OTHER IMPORTANT PART OF MEMORY.

THE DEFAULT ALLOCATIONS ORDER,LIBRARY, PROGRAM,VARIABLES.THE LIBRARY IS ALLOCATED LOWEST AND PROGRAM AND VARIABLES FOLLOW. THE LIBRARY BEGINS AT LOCATION 2051, OR $803.

ALTERNATE ADDRESSES FO THE BLOCKS ARE SIMPLE TO SPECIFY. THE NEW LOCATION FOR THE LIBRARY IS ENTERED AS A NUMBER AND DEFAULTS TO $803. ADDRESSES CAN BE SPECIFIED IN EITHER HEXADECIMAL OR DECIMAL. HEX MUST HAVE ($)

THE LIBRARY MUST BE BLOADED BEFORE A COMPILED PROGRAM CAN BE RUN. BY DEFAULT THE LIBRARY IS LOADED AT $803.WHEN A PROGRAM IS COMPILED TO EXPECT THE LIBRARY AT A DIFFERENT ADDRESS, THE LIBRARY MUST BE LOADED IN AT THE CORRECT ADDRESS BY USING "A" OPTION WITH THE BLOAD COMMAND. MOVING BINARY FILES, WITH ADR UTILITY.

THE BEGINNING ADDRESS FOR THE OBJECT CODE MAY BE SPECIFIED WITH:

1. **THE WORD HGR1**
2. **THE WORD HGR2**
3. **A DECIMAL OR HEX NUMBER**
4. **<RETURN>**

HGR1 AND HGR2 SET THE BEGINNING OF THE PROGRAM ABOVE THE APPROPRIATE HIRES SCREEN. THEN 4K RUNTIME LIBRARY DEFAULTS TO THE SPACE BELOW THE FIRST HIRES SCREEN. THIS DEFAULT LOCATIONS IS SUGGESTED FOR PGMS OF HIRES.

VARIABLE SPACE MAY BE SPECIFIED EXPLICITLY OR ALLOWED TO DEFAULT. THE BEGINNING OF VARIABLE SPACE DEFAULTS TO THE END OF THE OBJECT CODE.

COMPILED PGMS USE THE NORMAL HIMEM POINTER TO DETERMINE THE TOP OF AVAILABLE STRING SPACE, AND STRINGS GROW DOWNWARD FROM THERE.

COMPILATION OPTIONS:

COMPILATION LISTING	YES
PAUSE ON ERRORS	YES
INTEGER ARITHMETIC	YES
INTEGER CONSTANTS	YES
RESUME/DEBUG CODE	NO

ANSWERING "NO" OR "N" TO THE DEFAULT OPTION PROMPT PROVIDES A CHANGE TO TURN EACH OF THESE OPTIONS ON OR OFF.

THE COMPILER NORMALLY LISTS THE SOURCE FILE. TURNING THE LISTING OPTION OFF SUPPRESSES THE LISTING. ERRORS, WARNING, AND SPECIAL MESSAGES ARE PRINTED AS USUAL.

PAUSE ON ERRORS OPTION:
ERRORS NORMALLY HALT COMPILATION AND ALLOW THE USER TO ABORT OR CONTINUE COMPILATION. TURNING THE PAUSE OPTION OFF SUPPRESSES THE PAUSE AFTER ANY ERROR MESSAGE ARE PRINTED.

INTEGER ARITHMETIC OPTION:
TASC INCLUDES A FULL INTEGER ARITHMETIC PACKAGE. TRUE INTEGER ARITHMETIC ALLOWS OPERATIONS ON INTEGERS TO BE PERFORMED IN ABOUT HALF THE NORMAL TIME.INCLUDING THE OPTION SUBSTANTIALLY INCREASES THE SPEED OF PGMS THAT USE INTEGERS.

INTEGER CONSTANTS:
CONSTANTS IN A COMPILED PROGRAM CAN BE TREATED AS INTEGERS OR FLOATING POINT NUMBERS. SELECTING THE INTEGER CONSTANTS OPTIONS ALLOWS CONSTANTS THAT ARE USED AS INTEGERS TO BE STORED IN INTEGER FORMAT. IF A CONSTANT IS NEEDED IN FP IT INCLUDES BOTH.

INTEGER CONSTANTS TAKE UP TWO BYTES IN THE OBJECT FILE:FP TAKES FIVE.THE INTEGER CONSTANTS OPTION SHOULD NORMALLY BE LEFT ON.

RESUME/DEBUG CODE OPTION:
TURNING ON THE RESUME/DEBUG CODE OPTION CAUSES CODE TO HANDLE THE RESUME STATEMENT TO BE INCLUDED IN THE OBJECT PGM. THE RESUME IN APPLESOFT ALLOWS AN ERROR TRAPPING ROUTINE TO RESUME EXECUTION AT THE BEGINNING OF THE STATEMENT THAT CAUSED THE ERROR. TASC ALSO FULLY SUPPORTS ONERR GOTO. THE COMPILED VERSION OF ONERR GOTO TRAPS ALL RUNTIME ERRORS, INCLUDING THOSE THAT OCCUR WITHIN ROUTINES FROM THE APPLESOFT INTERPRETER.

INCLUDING THE RESUME/DEBUG OPTION REQUIRES THE COMPILER TO GENERATE EXTRA CODE AT THE BEGINNING OF EACH STATEMENT THAT MAY GENERATE AN ERROR. SELECTING RESUME/DEBUG CODE OPTION CAUSES THE OBJECT CODE TO BE LARGER AND SOMEWHAT SLOWER.

TURNING ON THE RESUME/DEBUG CODE OPTION HAS THE ADVANTAGE THAT ANY RUNTIME ERROR MESSAGES INCLUDE THE OBJECT CODE ADDRESS. NORMALLY, ONLY SOME OF THE ERRORS GENERATED BY THE RUNTIME LIBRARY INCLUDE AN OBJECT CODE ADDRESS. THE RESUME/DEBUG OPTION CAN BE USEFUL FOR DEBUGGING WITH THE COMPILER. HOWEVER, INCLUDING IT DOES DECREASE THE

SPEED AND INCREASE THE LENGTH OF THE COMPILED CODE. THE RESUME/DEBUG OPTION SHOULD BE LEFT OFF UNLESS IT IS ABSOLUTELY NEEDED. IF THE OPTION IS TURNED OFF THE COMPILER WILL IGNORE ALL RESUME STATEMENTS.

TERMINATING COMPILATION

THE COMPILER RUNS IN MACHINE LANGUAGE SO CTRL-C WOULD NORMALLY BE IGNORED AND RESET WOULD BE NECESSARY INSTEAD. TO TERMINATE:

CTRL-C IF USED STOPS COMPILATION, BUT ALSO DELETES THE OBJECT FILE IF COMPILATION IS ABORTED. TASC MODIFIES DOS, SO BY USING CTRL-C,OR RESET THEN DOS MUST BE REBOOTED. CORRECTLY EXITING THE COMPILER RESTORES DOS TO ITS NORMAL STATE.

COMPILING LARGE PROGRAMS

LARGE PROGRAMS MAY DISPLAY A SYMBOL TABLE FULL ERROR. ONE WAY TO CORRECT THIS MESSAGE IS TO TURN OFF THE INTEGER CONSTANTS OPTION. IF CONSTANT IS NEEDED LATER IT CAN BE A FP, AND ENTERED INTO THE SYMBOL TABLE. THE INTEGER ENTRY TAKES FIVE LOCATIONS, FP ENTRY TAKES EIGHT LOCATIONS. WITH INTEGER CONSTANTS OPTION OFF, CONSTANTS ARE STORED IN FP FORM. TURNING OFF THE OPTION SAVES FIVE LOCATIONS FOR EVERY CONSTANT THAT IS REFERENCED AS FP VALUE.

TURNING OFF THE INTEGER CONSTANTS OPTION ALSO SLOWS DOWN THE OBJECT CODE,SO THE OPTION SHOULD BE LEFT ON WHENEVER POSSIBLE.

#2 SEPARATE PROGRAM INTO PARTS:

USE THE COMMON COMMAND AND TRY TO SPLIT THE PROGRAM INTO PARTS AS NATURAL DIVISIONS AS POSSIBLE. PROGRAMS WITHOUT NATURAL DIVISIONS MAY PRESENT A PROBLEM SO YOU MUST MAKE ARTIFICIAL DIVISIONS.THE TECHNIQUE OF SPLITTING A LARGE PROGRA INTO SMALLER PROGRAMS THAT RUN IN SEQUENCE CAN SOLVE ALMOST ANY PROBLEM WITH PROGRAM SIZE. PASSING VALUES WITH COMMON MAKES SEPARATING PROGRAMS A MANAGEABLE PROBLEM.

EXECUTING A COMPILED PROGRAM****

1. INTERPRETED PROGRAMS ARE STORED AS APPLESOFT FILES INDICATED BY A "A".THESE FILES ARE EXECUTED BY A RUN COMMAND.

2. COMPILED PROGRAMS ARE STORED AS BINARY "B" AND MUST BE EXECUTED BY A BRUN COMMAND. THE NORMAL SEQUENCE FOR EXECUTING A COMPILED PROGRAM IS :

```
BLOAD RUNTIME
BRUN <FILE NAME>
```

COMPILED PROGRAMS WILL ONLY WORK IN APPLESOFT LANGUAGE NOT **INTEGER BASIC**

USING THE AMPERSAND(&):

ONCE THE COMPILED PGM HAS BEEN LOADED AND EXECUTED, IT CAN BE RE-EXECUTED BY TYPING AN (&) FOLLOWED BY <RETURN>. USING THE (&) IS

MUCH MORE CONVENIENT THAT HAVING TO CALL THE CODE.

HALTING EXECUTION OF COMPILED PGM:
<RESET> FOLLOWED BY NEW

NEW:

NEW CAUSES THE INTERPRETER TO RESET POINTERS, BUT NOT CLEAR THE PROGRAM SPACE. THEREFORE THE PROGRAM CAN BE SAFELY RE-EXECUTED IF NO PROGRAM LINES HAVE BEEN TYPED IN AND STORED INTO THE PROGRAM SPACE.

IMMEDIATE COMMANDS:
NONE

STATEMENTS NOT IMPLEMENTED:

THE FOLLOWING COMMANDS ARE \ NOT \ INCLUDED IN TASC

CONT	DEL	LIST
LOAD	SAVE	LOMEM:
&	RECALL	NOTRACE
SHLOAD	STORE	TRACE

THE FOLLOWING COMMANDS ARE SUPPORTED WITH SOME LIMITATIONS:

DEF FN DIM <CTRL-C>

DEF FN:
IN THE INTERPRETER, A DEF FN DOES NOT DEFINE A FUNCTION UNTIL THE DEF FN STATEMENT IS ACTUALLY EXECUTED AT RUNTIME. THE COMPILER, ON THE OTHER HAND SCANS ALL FUNCTIONS DEFINITIONS AT COMPILETIME. THEREFORE FUNCTION DEF CAN BE LOCATED ANYWHERE WITHIN THE SOURCE FILE. THE SOURCE CANNOT CONTAIN MORE THAN ONE DEF FOR A GIVEN FUNCTION,EVEN IF IDENTICAL.

DIM:
EXECUTING A DIM STATEMENT IN WHICH THE SPECIFIED DIMENSIONS ARE CONSTANTS SETS ASIDE THE SAME AMOUNT OF STORAGE FOR THE ARRAY EACH TIME THE PGM IS RUN.

EXECUTING A DIM STATEMENT IN WHICH THE SPECIFIED DIMS ARE ARITHMETIC EXPRESSIONS SETS ASIDE SPACE FOR THE ARRAY DEPENDING ON THE COMPUTED VALUE OF THE EXPRESSION.

DEFAULT DIM- IF AN ARRAY REF IS ENCOUNTERED BEFORE A DIM STATEMENT,THE ARRAY IS GIVEN THE DEFAULT MAX VALUE OF 10 FOR EACH DIM OF THE ARRAY.APPLESOFT ALLOWS THE USE OF 0 AS AN ARRAY SUBSCRIPT,SO AN ARRAY DIMED AT 10 IS ACTUALLY 11 (0-10)

CTRL-C:
TYPING CTRL-C DURING INPUT HALTS THE PGM

IF THEN:

THE COMPILER SUPPORTS ITS USE, BUT AN IF/THEN STATEMENT WITH A STRING EXPRESSION IS FLAGED AS AN ERROR DURING COMPILATION.

GET:
WORKS FINE

READ:
WORKS FINE

ONERR GOTO:
USED WITH RESUME/DEBUG NEW,END,STOP:

NEW-ERASES THE CURRENT PROGRAM
STOP-PRINTS THE MESSAGE "BREAK IN ####"
END- TERMINATE EXECUTION

GARBAGE COLLECTION:
EACH TIME GC IS NECESSARY, THE PROGRAM SIMPLY SUSPENDS EXECUTION WHILE THE GC ROUTINE HOUSE-CLEANS.

LANGUAGE ENHANCEMENTS:
INTEGER:

CHR$	COLOR=	DRAW
FOR	HCOLOR=	HLIN
HPLOT	HTAB	IN#
LEFT$	LET	MID$
ON GOSUB/TO	PDL	PLOT
POKE(2)	PR#	RIGHT$
ROT=	SCALE=	SCRN
SPEED=	SPC	SUBSCRIPTS
TAB	VLIN	VTAB
WAIT	XDRAW	

THE ABOVE OPERATIONS EXPECT INTEGER VALUES.

INTEGER ARITHMETIC PACKAGE:

ADDITION	MULTIPLICATION
NEGATION	SUBTRACTION

THE FOLLOWING INTEGER OR FP

AND	FRE	IF/THEN
NOT	OR	POS

THE FOLLOWING RETURN INTEGER VALUES:

ASC	LEN	PDL
PEEK	POS	SCRN

COMMON:

CLEAR COMMON:

```
CLEAR CHAIN:
DEFCOMMON:
USECOMMON:
```

THE DEFCOMMON AND USECOMMON STATEMENTS ARE DESIGNED FOR CREATING LARGE SYSTEMS OF PROGRAMS THAT COMMUNICATE WITH EACH OTHER.

MENU

```
1000  INPUT "WHICH PACKAGE?".N
1010 IF N = 1 THEN ? D$"BRUN GL"
1020 IF N = 2 THEN ? D$"BRUN AP"
1030 IF N = 3 THEN ? D$"BRUN AR"
```

GL

```
10 REM!DEFCOMMON A,B(3,4),C$
   .
   .
1000 REM!CLEAR CHAIN
1010 ? D$"BRUN GL1"
```

GL1

```
10  REM!USECOMMON A1,B1(3,4),C1$
    .
    .
1000 REM!CLEAR CHAIN
1010 ? D$"BRUN GL2"
```

GL2

```
10 REM!USECOMMON A2,B2(3,4),C2$
   .
   .
1000 REM!CLEAR CHAIN
1010 ? D$"BRUN GL3"
```

GL3

```
10 REM!USECOMMON A3,B3(3,4),C3$
   .
   .
1000 ? D$"BRUN MENU"
```

PASS1:
FIRST PASS, PERFORMS SYNTAX ANALYSIS AND GENERATES MOST OF THE CODE. ALSO COLLECTS INFO ABOUT VARIABLES,LINE NUMBERS, AND STORES IT IN SYMBOL TABLE

PASS2:

USES INFO FROM PASS1 AND USES THE SYMBOL TABLE TO STORE INFO.ALSO DECIDES HOW MUCH STORAGE SHOULD BE ALLOCATED.

SYNTAX ANALYSIS:
LEXICAL-PRINT,FOR BECOMES UNDER LEXICAL TOKENS

ERROR MESSAGES AND DEBUGGING:

DECLARATION:
INTEGER OR COMMON DECLARATIONS OUT OF SEQUENCE OR NOT AS BEGINNING OF PGM USECOMMON AND DEFCOMMON BOTH DECLARED IN A SINGLE PROGRAM VARIABLE DECLARED AS COMMON MORE THAN ONCE.

INCOMPLETE:
INCOMPLETE EXPRESSION MISSING RIGHT PARENTHESIS IN EXPRESSION

OPERAND:
ILLEGAL OPERAND IN EXPRESSION ARITHMETIC CONSTANT TOO LARGE

REDEFINED:
FUNCTION DEFINED MORE THAN ONCE SPECIFIED ARRAY DIM DIFFERENT THAN THE FIRST DIM SPECIFIED

SUBSCRIPT:
FIRST SUBSCRIPT MISSING DIM NOT AN INTEGER CONSTANT DIM NEGATIVE OR GREATER THAN 32767 MORE THAN 88 SUBSCRIPTS DIFFERENT NUMBER OF SUBSCRIPTS THAN IN FIRST USAGE

SYMBOL TABLE FULL:
COMPILER OUT OF SYMBOL TABLE SPACE.

SYNTAX:
MISSING OR ADDED CHARACTER OR ITEM LINE NUMBER GREATER THAN 65534

TOO COMPLEX:
EXPRESSION TOO COMPLEX OBJECT CODE OR VARIABLE EXTEND PAST 48K

TYPE MISMATCH:
NUMERIC EXPRESSION WHERE STRING WAS EXPECTED STRING EXPRESSION IN IF/THEN

UNDEFINED LINE NUMBER OR FUNCTIONS PRODUCE FATAL ERROR AT THE BEGINNING OF PASS2.

SELF-MODIFYING PROGRAMS:
PHONE LIST ON APPLE DEMO THIS PROGRAM WILL NOT COMPILE PROPERLY. MUST BE RE-WRITTEN IN STRAIGHTFORWARD METHODS TO BE COMPILED.

ADR:
PRINTS OUT THE DECIMAL BEGINNING ADDRESS AND LENGTH OF THE MOST RECENTLY LOADED FILE. MUST USE THIS INFO WITH "A" AND "L" PARAMETERS TO BSAVE THE M.L. PROGRAM.

THE NORMAL PROCESS FOR MOVING A PROGRAM IS:

```
BLOAD <FILE NAME>
EXEC ADR
BSAVE <FILE NAME>,A<ADDRESS>,L<LENGTH>
```

CREATE ADR: TO TRANSFER ADR TO ANOTHER DISK

1. **LOAD CREATE ADR FROM TASC DISK**
2. **REMOVE TASC DISK**
3. **INSERT DISK FOR NEW COPY**
4. **TYPE RUN**

CONVERTING 3.2 TO 3.3:
USE MUFFIN ON DOS MASTER