DECUS
PROGRAM LIBRARY

DECUS NO.	8-159
TITLE	CINET-BASIC
AUTHOR	Bud Pembroke and David Gillette
COMPANY	Computer Instruction NETWORK Salem, Oregon
DATE	October 21, 1968
SOURCE LANGUAGE	

CINET-BASIC

DECUS Program Library Write-up

DECUS No. 8-159

This interpretive compiler was patterned after Dartmouth BASIC. It was built by modifying DEC's FOCAL, and uses many of the same subroutines and/or methods. Included are statements such as Let, Print, GO TO, IF---then, For and Next, GOSUB and Return, Input; and commands such as RUN, Edit, ERASE, LIST and Interupt. Error messages are given in terms of an error number and line number.

The program occupies locations 0000-3252 and 4600-7600. The user's written BASIC code is stored from 3252 on. This working storage can be expanded by deleting the trigonometric and exponential functions.

Minimum hardware: 4K PDP-8, ASR-33 Teletype.

BIN tape only.

PREFACE

CINET-BASIC is an interpretive language written by the Computer Instruction NETWORK, using FOCAL subroutines. It is patterned after Dartmouth Basic, and written for a standard PDP-8, 8/s, 81 or 8L computer with 4K storage.

CINET-BASIC is set up to run on a PDP-8/S only; there is a parity error check. This will cause other PDP-8 computers to halt at location 2654. You may disable this test by placing a Nop (7000) in location 2654.

CONTENTS

How to Load and Start CINET-BASIC

TO LOAD

1. Place the CINET-BASIC tape in the tape reader.

2. Turn the Teletype switch to ON LINE.

3. Place 7777 in the switch register.
   (CINET-BASIC is loaded with the BIN loader.)

4. Depress LOAD ADD Key.

5. Depress Start Key.

6. Set tape reader switch to start.

The CINET BASIC tape is in three sections. It will stop after each part and give a check sum test. (If accumulator is clear, the program is loading correctly. If the accumulator is not clear, reload the preceding section.

7. After checking accumulator, depress the CONT switch on the computer. Repeat this process until last section is in and then start CINET BASIC. (See TO START.)

   Section one of CINET BASIC tape IS page zero. If only page zero has been clobbered, then read in section one only.

   Section three contains the extended functions:

   SIN
   COS
   ATN
   EXP
   LOG

   These are loaded at the option of the user. They consume approximately 800 locations of the user's program storage area and may be omitted by not loading section three of the tape.

   Caution: If section three is not loaded at the same time as sections one and two, be sure to type "ERASE ALL" prior to loading section three.

TO START

Place 0200 in the switch register, press the LOAD ADD key, and then press the START key. The computer will then type:

```
0000 #XXXX
     READY
```

Caution: To save reloading CINET-BASIC, always hit "STOP" switch on the computer before turning power off.

Area of Storage used by CINET-BASIC:

BASIC is stored from 0000 to 3252 and from 4577 to the BIN Loader. The area from 3252 to 4577 is temporary storage for your BASIC programs. Machine language programs in this area will not affect CINET-BASIC. However, this area will be used when a CINET-BASIC program is loaded.

STORED PROGRAM

Computers are unable to solve a problem unless a "program", or list of instructions, has been stored in the computer memory. To solve a problem, the programmer must follow a sequence of three steps:

1. Write the program. Using the CINET-BASIC language, write a list of instructions which will solve the given problem.

2. Load the program. Loading the list of instructions into the computer's memory is usually done by simply typing "ERASE ALL" and then typing the program on the Teletype keyboard. Should an error be made, the entire line may be deleted by typing a back arrow (←). Single characters may be omitted by depressing the RUBOUT key, one RUBOUT for each bad character.

3. Run the program. Loading a program is like loading a gun: You are only prepared for action; nothing happens until you pull the trigger. After a program has been loaded, you execute or run the program by typing "RUN" and a Carriage Return on the Teletype.

WRITING THE PROGRAM

Below is a sample program written in CINET-BASIC. The program computes the total amount paid back in a y ar for a give principal and rate of interest.

```
10 PRINT "PLEASE TYPE THE PRINCIPAL AND INTEREST RATE"
20 INPUT P, I
30 LET A = P*I+P
40 PRINT "AMOUNT IS", A
50 GO TO 10
60 END
```

Explanation of program

Statement 10:
The computer will type on the Teletype:

PLEASE TYPE THE PRINCIPAL AND INTEREST RATE

Statement 20:
The computer will type a "?" and wait while you type the principal and then the interest rate.

Statement 30:
(P timesI) plus P will be computed.

Statement 40:
The computer will type out

AMOUNT IS (and the value for the amount)

Statement 50:
The computer will then "loop" back to statement 10 and repeat the above operations.

Statement 60: indicates the last instruction.

STATEMENT NUMBERS

For a stored program each statement must be preceded by a statement number (1 through 2047). You may type statement numbers in any order. The program will run in numerical order..

VARIABLES

Variables in CINET-BASIC are denoted by any single letter. Thus A, B, X, and N are variables. A variable may also have a subscript. The subscript is designated by immediately following identifiers with the subscript value enclosed in parentheses.

Examples:

```
A(3)
A(M)
X(39)
M(I)
```

Variations from other basics

Only single subscripts allowed (range $\pm$2046)
Subscript may be computed; however, only the integer value within the accepted range will be used.

Examples:

```
A(I+2)
X(3*4-3)
```

In the following, all examples are variable A(12)

```
A(12)
A(12.3)
A(12.7)
```

NUMBERS

Input

Numbers may be typed in with or without signs. They may be typed in fixed point, floating pcint or exponential form. However, no more than 6 significant digits may be given. Should an error be made while inputting a number, type a back arrow (←) and retype entire number.

FIXED POINT EXAMPLES:

5

-2701

FLOATING POINT EXAMPLES:

0.00025

3.2

EXPONENTIAL EXAMPLES:

15E32 means 15 X 10^{32}

3.25E-6 means 3.25X10^{-6} or .00000325

The limit on exponents is $\pm$ 615

Caution: Should you exceed the limit of the exponents by some iterative process, the exponent generated will reverse its sign.

Output

The printout of numbers will be given in floating points unless the value exceeds the range of 10^{-5} to 10^{9}. Numbers that exceed this range will automatically be printed in exponential form.

Examples:

1.5

-2035.

0.3200000E11

Caution:

Values are accurate to six significant digits only; disregard all digits beyond this range.

INSTRUCTIONS

Following each statement number is an instruction. The instruction name must be followed by a space. Correct spacing in CINET-BASIC is important. However, in general, correct spacing can be achieved if the user will use the normal conventions of spacing between words and not spacing in algebraic expressions.

Instruction	Example	Explanation
A. PRINT	30 PRINT "X=",X	outputs information
B. INPUT	10 INPUT A	allows input of values for variables
C. LET	55 LET B=4*(X+3)	defines variables
D. GO TO	140 GO TO 30	unconditional jump
E. IF	70 IF A = 5 THEN 59	conditional jump
F. FOR	85 FOR I=1 TO 30 STEP 2	automatic looping
G. NEXT	37 NEXT I	end of FOR loop
H. GOSUB	127 GOSUB 300	enter subroutine
I. RETURN	320 RETURN	return from subroutine
J. DIMENSION	12 DIM A(15)	designates upper limit of subscript
K. REMARKS	5 REM I STANDS FOR INTEREST	explanation remarks only
L. STOP	43 STOP	logical stop
M. END	2000 END	last statement in program

A. PRINT

The print instruction commands the computer to Output information.

If print is followed by a letter or series of letters separated by commas or semicolons, the values presently assigned to these variables will be typed. If print is followed by any series of characters enclosed within quotation marks, the enclosed characters will be reproduced exactly.

Examples:

```
100 PRINT A
```

If "A" were first computed to be 5, the Teletype (TTY) would type:

```
5.
```

```
50 PRINT "AMOUNT IS", A
```

The TTY would type

```
AMOUNT IS 5.
```

The print instruction may also be used to evaluate one or more expressions.

Examples:

```
1000 PRINT 2-5, SQR(64)
```

2-5 would be evaluated and output as -3. Then the square root of 64 would be found and output as 8. 14 spaces are reserved for each numerical value typed. The output would appear as

```
-3              8.
```

The print statement will normally give a carriage return and line feed at the completion of typing. This carriage return, line feed, may be suppressed by ending the statement with a comma or semicolon.

Examples:

```
20 PRINT "WELL.!"

30 PRINT "HI" ,

40 PRINT "THERE"
```

The TTY type the following, suppressing the carriage return and line feed, after the "HI ".

```
WELL.!
HI THERE
```

The print instruction can be used to give line spacing.

Example:

```
1000 PRINT
```

The TTY would give only a carriage return and line feed.

Variation from other Basics:
There is no distinction made between a comma and a semicolon in a print statement. 14 spaces are reserved for each Variable regardless.

No attempt is made to give an automatic carriage return, line feed, at the end of 72 characters (width of line on TTY). The carriage return line feed is only given when the print statement is complete. It is therefore necessary for the programmer to be aware of the length of his printout.
See Appendix A.

B. INPUT

The input instruction should be followed by a letter or series of letters separated by commas. When the instruction is executed, the computer will type a question mark "?", stop, and wait for you to type the values you wish assigned to each variable.

Example:

```
21  INPUT A, B
```

On executing this instruction, the following would result:
The computer prints a "?" and waits for input from the keyboard. If you type a 5 terminated with a comma, the computer assigns the value 5 to the variable A, types a "?" and waits again. If you type 70.2 terminated by a comma, the machine sets B = 70.2 and types a carriage return. The result on paper looks like this:

```
?5,?70.2,
```

A value may also be terminated with a space, carriage return, or semicolon.
Variations from other Basics.
See Appendix B.

C. LET

The LET instruction is used to define a value for a variable. This assigned value may be a single number or an algebraic expression involving some arithmetic operations. The arithmetic operations possible are:

sign	operation	example
+	addition	A + Z
-	subtraction	3 - 5.03
*	multiplication	B * C
/	division	12/3
↑	exponentiation	A↑2 (means A^2) only integer portion of an exponent will be used
( )	parentheses may be used in pairs to clarify any algebraic expression.	

Order of priority of operations:

1. Values inside parentheses
2. Exponentiation
3. Multiplication
4. Division
5. Addition and subtraction

Examples:

```
32 LET S = 5
```

Defines the variable S as equal to 5.

```
40 LET Y = 4*A*X↑2+X
```

Defines Y to equal $4AX^2+X$

```
55 LET Y = 2*(2*A-B)/3
```

Defines Y to equal $\frac{2(2A-B)}{3}$

Functions

In addition to the five arithmetic operations, the computer can evaluate several mathematical functions. These functions are given special 3-letter names.

Functions:	Interpretation:
SIN(X)	Find the sine of X
COS(X)	Find the cosine of X
ATN(X)	Find the arctangent of X

X in the above functions is assumed to be measured in radians. Should you wish to find the sine of an angle in degrees, the conversion factor $\pi/180$, may be used.

Example:

```
10 PRINT SIN(15*3.14159/180)
```

This will give the sine of 15 degrees

EXP(X)	Find e^X or $(2.718281)^X$
LOG(X)	Find the natural log of X
ABS(X)	Find the absolute Value of X
SQR(X)	Find the square root of X
INT(X)	Find integer part of number in the range ± 2046
RND(X)	Find a nonstatistical pseudo-random number between ± 1

Caution: The last letter of the function name must be followed immediately by a left parenthesis.

Variations from other Basics

Multiplication and division are given different priorities; therefore, A/B*C gives a value for A/(B*C).
The tangent function is not available; however, it may be created by SIN(X)/COS(X).
See Appendix C.

D. GO TO

GOTO

A GO TO instruction is always followed by a statement number, directing the computer to that statement. The computer will execute instructions in numerical order unless re-directed by a GO TO statement of an IF, THEN statement.

Example:

```
50 GO TO 25
```

E. IF, THEN

This statement allows the computer to make a decision. Each statement must be of the form:

IF (variable) (relation) (expression) THEN (line number)

The relation may be made up of a <, >, =, or any combination of the three.

Examples:

```
20 IF X = 0. THEN 85
```

The computer will go to statement 85 if X = 0.; otherwise, it will execute the next statement after 20.

```
34 IF X< = 3+5*A THEN 97
```

If X is less than or equal to 3+5*A, statement 97 will be executed. If not, the next statement after 34 will be executed.

Variations from other Basics:

The left member may only be a variable.

Caution: Do not space between relation signs.
Space before and after THEN.
If Zero is used in an IF statement, use it with a decimal point.

F. FOR

This instruction is used for convenience in setting up program loops and iterations. It must be used in conjunction with a NEXT statement. The general format is:

```
100 FOR A = X TO Y STEP Z
```

The variable A is initialized to the value of X, then those statements following the FOR are executed until a NEXT statement is encountered. When a NEXT statement is found, control is sent back to the preceding FOR. The Value A is then incremented by Z and compared to the value Y. If A is less than or equal to Y, then those statements following the FOR will once more be executed. This process will be repeated until A is greater than Y. At that time the statements following the NEXT will be executed. X, Y, and Z may be algebraic expressions. Z must have a positive value. If STEP is omitted, then the increment will be one.

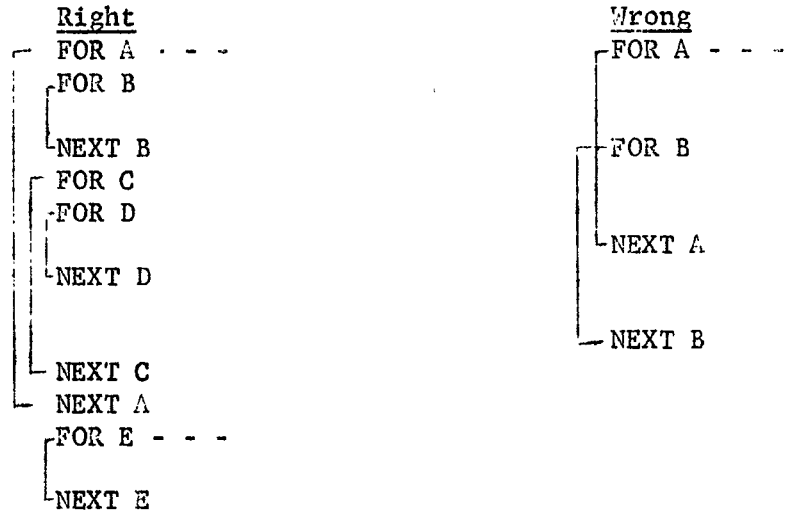
Example:

```
10 FOR A = 1 TO 2↑2 STEP 2.5

20 PRINT A

30 NEXT A
```

When RUN is typed, this program would give:

```
1.

3.5
```

FOR, NEXT statements may be nested as diagrammed on the left.



When using a FOR loop, the loop may be terminated prematurely with an IF statement. However, when exiting from nested FOR statements prematurely, the entire nest should be terminated.

Caution: Space before and after "TO" and "STEP".

G. NEXT

Used only with FOR statements. The general form is:

```
20 NEXT A
        (See FOR)
```

H. GOSUB

When a particular part of a program is to be performed more than one time, or possibly at several different places in the program, it is most efficiently programmed as a subroutine. The subroutine is entered with a GOSUB statement, where the number is the first line number in the subroutine. The subroutine must logically end with a RETURN instruction. This transfers control back to the statement following GOSUB.

Example:

```
10 PRINT "A",

20 GOSUB 100

30 PRINT "B",

40 GOSUB 100

50 PRINT "C"

60 STOP

100 PRINT "XXX"

110 RETURN
```

This program would give:

```
AXXX
BXXX
C

READY
```

I. RETURN

The RETURN must be the last statement in a subroutine.

Example:

```
300 RETURN
```

See GOSUB.

J. DIMENSION (DIM)

DIM is used to reserve space for subscripted variables greater than 10.

Example:

```
20 DIM A(20), B(15)
```

Variations from other Basics: See Appendix J.

K. REMARKS (REM)

This statement is used to allow the programmer to insert explanatory remarks in a program. The computer will completely ignore the line when RUN is typed.

Example:

```
100 REM STATEMENT 110 MAY BE CHANGED TO GIVE NEGATIVE ROOTS.
```

L. STOP

The stop instruction is used to designate a logical stop in the middle of a program.

Example:

```
30 STOP
```

M. END

The END instruction is used to designate the last statement in a program.

Example:

```
99 END
```

Variations from other Basics. See Appendix M.

Direct Mode

If you wish to run a program that is only one step long such as:

```
PRINT SQR(49)
```

you may use direct mode in CINET-BASIC. This is accomplished by simply typing the above instruction (without a statement number) followed by a carriage return. In this example, "7." will be printed immediately. The instruction is executed but not stored for future use.

Commands

There are certain commands that are intended for direct mode only. These are:

	Commands	Example	Explanation
A.	RUN	RUN cr	run stored program
B.	LIST	LIST cr	LIST out stored program
C.	EDIT	EDIT 20 cr	Set up statement 20 for EDIT
D.	ERASE	ERASE ALL cr	ERASE entire stored program
E.	INTERUPT	(hold CTRL key and depress C key)	This interupts CINET-BASIC and returns control to user

A. RUN

RUN will direct the computer to execute the stored program.

Examples:

RUN This executes a stored program starting with the lowest-numbered instruction.

RUN 50 This executes a stored program starting with statement 50 and ignoring all lower-numbered instructions.

B. LIST

The LIST command will direct the computer to list out the stored program. LIST followed by a statement number would direct the computer to list from that statement number on.

Examples:

LIST	Lists the complete stored program
LIST 37	All instructions with statement numbers equal to or greater than 37 will be listed.

Saving Programs:

To save on tape a CINET-BASIC : program that is storedin the computer, the user should:

1. Type LIST (DO NOT depress the RETURN key).
2. Turn on tape punch.
3. Type several @ signs to get leader tape (press the Shift, REPEAT, and P keys in that order; release in reverse order).
4. Depress the RETURN key.

When user's program has been typed and punched out:

5. Type several more @ signs to get trailer tape.
6. Turn off tape punch.

C. EDIT

This command must be followed by a statement number. The statement designated will be set up for edit upon receiving a carriage return. The user then types a search character. The computer will type out the contents of that line until the search character is typed. The user may then perform any of the following optional operations:

1. Type in new characters. They will be added to the line at the point of insertion.
2. Type a FORM (CTRL/L). This directs the computer to proceed to the next occurance of the search character.
3. Type a BELL (CTRL/G). After this the user may change the search character.
4. Type a RUBOUT. This deletes characters to the left. One character is deleted for each time that the user strikes the RUBOUT key.

5. Type ← (SHIFT O). This deletes all the line to the left margin (not the line number).

6. Type RETURN. This terminates the line, deleting characters to the right margin.

7. Type LINE FEED. This saves the remainder of the line from the point LINE FEED is typed.

A completely new statement will replace an existing one by typing the old statement number followed by the new statement.

D. ERASE

Erase must be followed by some parameter.

Examples:

ERASE 30

The above command will erase from the stored program statement number 30.

ERASE ALL

The above command will erase the entire stored program.

Note: Typing a statement number alone followed by a carriage return will delete from storage the contents of that line. It will, however, leave the statement number in the stored program. This line number is ignored when RUN is typed.

E. INTERUPT

You may interupt CINET-BASIC at any time to return control to the user by typing a CTRL/C.

Error and Control Messages

These messages are given as a four-digit number followed in turn by a "#" and an additional four-digit number. The last number where applicable designates the statement number where the error occured. Number 2080 designates a direct mode error.

Example:

0206 #0020

This implies an illegal command in statement 20.

Below is a list of messages.

0000 #X	Manual start or interupt
0031 #X	Function not available
0109 #2080	Line number too large
0167 #X	Storage filled by statement X
0195 #X	GO TO Y, and there is no Y
0206 #X	Illegal command in statement X, or next without for
0263 #X	If without then
0275 #X	Left side of equal sign is illegal in statement X
0286 #X	Too many right parentheses in LET statement
0353 #2080	EDIT X and there is no X
0453 #X	Illegal variables or illegal spacing in FOR, or IF statement
0464 #X	Operator missing in statement X or space in Subscripted Variable
0506 #X	Variable illegal; illegal spacing in FOR, IF or PRINT statements
0507 #X	Operator error in statement X
0534 #X	Parentheses error or Parentheses NOT immediately following function
0581 #2080	ERASE with no parameter
0598 #2080	Line number too large
0716 #2080	Input buffer error
0811 #2080	Line too long or program too long
1301 #X	Attempt to take LOG of zero
1486 #X	Constant or input with too many significant figures
1641 #X	STEP used incorrectly
1642 #X	THEN used incorrectly
1657 #X	TO used incorrectly
1784 #X	Argument of function or subscript exceeds acceptable range
1832 #X	Attempt to divide by zero
1943 #X	Attempt to take square root of negative number

If someone else is using the computer, you may punch a program tape by typing your program on an off-line Teletype with the TAPE PUNCH ON. This is accomplished by first typing several @ signs for leader tape. Then type your program. Finally type several more @ signs for trailer tape. Should you make an error while typing, a "back-arrow" (←) will erase the present line when the tape is read in the computer. Or, you may have the computer delete one or more characters from the tape by following bad characters with RUBOUT's, one RUBOUT for each bad character. (Caution: DO NOT BACKSPACE TAPE.) Now take your program tape to the computer's Teletype and insert the tape in the tape reader. Type "ERASE ALL", a Carriage Return, and place the tape reader switch in START position. The program will be quickly loaded into memory. After the tape has loaded, place the tape reader switch in STOP position.

When loading a long program (tape), it is possible to receive the error message 0716 #2080. This may be corrected by simply stopping the tape reader periodically to allow the Input Buffer to clear itself.

APPENDIX

CINET-BASIC Extensions

A. PRINT

1. It is possible to get a carriage return without a line feed. This is accomplished by placing a # (number sign) in the print statement. It must be separated from other output by commas, but not in quotes. This will cause the TTY to give a carriage return only.

   Example:

```
10 PRINT "IS    RIGHT?",#

20 PRINT "  ",A
```

   If A is 5, then the TTY will type

```
IS 5. RIGHT?
```

   This is very useful in plotting or graphing.

2. A carriage return may be substituted for a " sign if the quote is the last character in a line.

B. INPUT

The control characters " and # of the print statement are also operative on the input statement.

Example:

```
150 INPUT "TYPE THE AMOUNT OF INCOME", A
```

The TTY would type the following and then wait for you to assign a value to 'A' as shown below.

```
TYPE THE AMOUNT OF INCOME?
```

C. LET

It is possible in CINET-BASIC to allow one LET statement define a series of variables by separating each equation with a comma or semicolon.

Example:

```
10 LET A = 2, B = 3, C = B*A

20 PRINT A, B, C
```

This program would give

```
2.          3.          6.
```

J. DIMENSION (DIM)

The DIM statement may be omitted, as it is not necessary to reserve space in CINET-BASIC.

M. END

In CINET-BASIC the END instruction is not necessary. However, the computer will not acknowledge the completion of the program with READY, unless END or STOP is used.

attached to 8-15-9 Cinet Basic.

Machine Language Package for PDP-8

These five programs are designed to assist in an instructional situation where octal programming is being used.

They are located in storage so that they do not harm the CINET-BASIC program.

ROLL: Readable Octal Listing
This program dumps any sized section of storage on TTY paper and/or tape. Each line starts with an address and has the content of eight locations. The paper tape produced is suitable for reloading via MALL.

MALL: Machine Language Loader
This program accepts octal codes as addresses or contents of storage locations. Programs may be loaded into storage via paper tape or keyboard.

BITAP: Binary Tape Punch
This program produces binary tapes in proper format to be reloaded via DEC's Binary Loader. It is very similar to DEC's Binary Punch (8-5-U).

IDIS: Input Decimal Conversion Subroutine

ODIS: Output Decimal Conversion Subroutine
These are single precision binary to signed decimal conversion subroutines for Input and Output via ASR-33.

Materials available:

Documentation includes: Listing and pamphlet explaining in detail how to use these routines.

Tapes: Binary Tape with programs stored from 4000-4577 and 200-252
Source Tape with no origin.

The PDP-8 Package
A Micro Manual
From C. I. NETWORK

Introduction: The package contains programs designed to aid the programmer in his use of machine language. He can type o· tape his program into the computer's storage. He can have his program listed in a readable form or dumped in a compact form. Also Subroutines are available that enable the computer to input or output integers in base ten. While the Package is being used, CINET-BASIC is temporarily disabled. If someone tries to start it at 200, a program protection message is typed out. The package is located in storage locations 4000-4570. The CINET-BASIC Protector is located in 200-252.

MALL: Machine Language Loader

The MALL program is used for direct loading of machine language programs via the TTY. The starting address is 4000. The TTY will nod its head and it is ready to accept your program in octal machine language either from the keyboard or from tape.

Format: First type the four digit octal address of the first location of your program, followed immediately by a right parenthesis. This sets up a simulated address register in the MALL program, that keeps track of where your program is. Next type the instruction or data that is to go into that address, followed by a space. Keep typing computer words following each one by a space. These will be stored sequentially in succeeding locations.

Simulated Address Register: The SAR may be reset at any time by typing a new address and following it by a right parenthesis. If a parenthesis does not immediately follow the first word typed, then that word will be stored in location 20. The SAR is initialized at 20 each time the MALL Program is started.

Errors: Whenever you are typing or running a tape into the computer, and an error is detected, the computer will type the letter E immediately.

The most common sources of errors are the following:

A four-digit octal word is followed by something other than a space or right parenthesis.

A illegal character is typed. Illegal characters include:

The numbers 8 and 9

All letters

All printed symbols except space and right parentheses.

A computer word has more or less than four-octal digits.

Ignored Characters: The Rub-Out character is ignored completely by the Package. It can be used to correct mistakes made while punching a tape off-line. When a mistake is made, push the backspace lever, then the Rub-Out key and the mistake will be punched full of holes. The Carriage Return and Line Feed characters may be used freely at any time. They are completely ignored by the Package.

ROLL: Readable Octal Listing

The ROLL program is used for listing the contents of a section of storage in the computer on paper or on tape. The starting address is 4200. The TTY will punch some leader if the tape punch is on, do a carriage return and line feed, and then Halt, waiting for section addresses on the switch register.

Format: When the TTY stops, the AC will be clear, place the first address of a section in the switches and press the continue key. This address will now be in the AC, and the computer has again Halted. Now place the last address of the section in the switches and press continue. The computer will type the first address and eight contents per line until the entire section is typed out. When it stops, if another section is wished, repeat the instructions in this paragraph. If trailer is wished, restart the program at 4200.

BITAP: BInary TApe Punch

The BITAP program is used to dump a program from the computer's storage onto tape in a compact form. The starting address is 4300. Make sure that the tape punch is on, for the BITAP starts by punching tape leader.

Format: When the computer stops punching leader, it Halts to receive the number of sections of storage to be dumped. Place this number in the Switch Register (Base 8), and press continue. Now enter the first and last address of each section, following each entry by a press on the continue key. After the last address of each section is received, the computer will punch out that section. When the appropriate number of sections has been punched out, the computer will automatically punch a group of trailer on the tape.

Bin Loader: These tapes may be loaded later using the Bin Loader. Put the leader of the tape in the reader, set SR = 7777 push LOAD ADD, and START on the computer, then push the tape reader lever to START on the TTY. The tape should read in. If it does not, read pages 451-453 in the Digital Small Computer Handbook, 1967 Edition.

IDIS and ODIS: Input/Output Decimal Integer Subroutines

The IDIS and ODIS programs are used to input and output integers in base ten via the TTY. IDIS is called at address 4100, ODIS at 4400. The AC is used as the only means to transferring data between these subroutines and the main program.

Format: The IDIS program will accept from the TTY, any number of preceding spaces, either a plus or minus sign, and digits that make up a base ten number in the range of $\pm$ 2047. After a number is typed in, it must be terminated by a space or comma or any other non-numeric printed symbol. Leader, Rub-Out Carriage Return and Line Feed characters are completely ignored by the Package. These may be typed in at any time preceding the terminating character. ODIS prints out in five

places every time. This leaves room for a negative four-digit number. If all this room is not needed, spaces are printed on the left.

AC Results: Any data in the AC upon entering IDIS is destroyed. The AC is clear at the return from ODIS.

Errors: If a number larger in absolute value than 2047 is typed in, the computer types out the letter E and returns with 2047 in the AC. This is the only error that the computer can detect. Other error symptoms may be noticed in running the program as follows:

No Output: Be sure to have a TLS instruction at the beginning of your main program. Caution: Do not loop back to this instruction.

Incorrect Output: Be sure that immediately following each use of IDIS you take proper care of the data which is in the AC. Also, be sure that the data to be printed out is in the AC before each use of ODIS.

Subroutines work first time, but that's all: Be sure not to have the addresses of IDIS or ODIS in the auto-index registers (locations 10-17).

USE OF MALL: MAchine Language Loader

1) Place 4000 in SR.

2) Press LOAD ADD on the computer.

3) Press START on the computer.

4) Enter your program via tape or keyboard (see Format section, page 1)

5) Be sure to space at the end of each line.

6) When your program is completely loaded, type a CR and LF.

7) Press STOP on the computer.

8) Your program is now ready to be run.

EXAMPLE of MALL Format:

On the keyboard:

```
0032) 1041 1042 7402 0041)0020 0030
```

Result in Storage:

Addr.	Machine Code
0032	1041
0033	1042
0034	7402
.	.
.	.
.	.
0041	0020
0042	0030

SAMPLE Program using IDIS and ODIS:

```
0010    6046    START   TLS
  11    4420    RERUN   JMS I IDIS
  12    3022            DCA X
  13    4420            JMS I IDIS
  14    1022            TAD X
  15    4421            JMS I ODIS
  16    7402            HLT
  17    5010            JMP RERUN
  20    4100     IDIS   Addr.
  21    4400     ODIS   Addr.
  22       0        X   Temp data storage
```

USE OF ROLL: Readable OctaL Listing

1) Place 4200 in SR.
2) If a tape is desired, press the Lock-On lever on the TTY punch.
3) Press LOAD ADD on the computer.
4) Press START on the computer.
5) When the computer stops, place first address of a section in SR.
6) Press CONT.
7) Place last address of the section in SR.
8) Press CONT.
9) If another section is desired, continue doing Steps 5-8.
10) If a tape is being punched, re-do Steps 1, 3, 4 only.
11) Press the Unlock lever on the TTY punch.

USE OF BITAP: BInary TApe Punch

1) Place 4300 in SR.
2) Press the Lock-On lever on the TTy punch.
3) Press LOAD ADD on the computer.
4) Press START on the computer.
5) When the computer stops, place the number of sections desired in SR (base 8).
6) Press CONT on the computer.
7) Place first address of a section in SR.
8) Press CONT.
9) Place last address of the section in SR.
10) Press CONT.
11) Continue doing Steps 7-10 until the computer punches trailer.
12) Press the Unlock lever on the TTY punch.

CORRECTIONS TO DECUS NO. 8-159

LOCATION	CONTENTS	
		*2654
2654	7000	7000
		*5515
5515	4763	4763
5516	2367	2367
5517	5714	5714
5520	5600	5600
		/ THESE ARE CORRECTIONS FOR CINET-BASIC VERSION 1