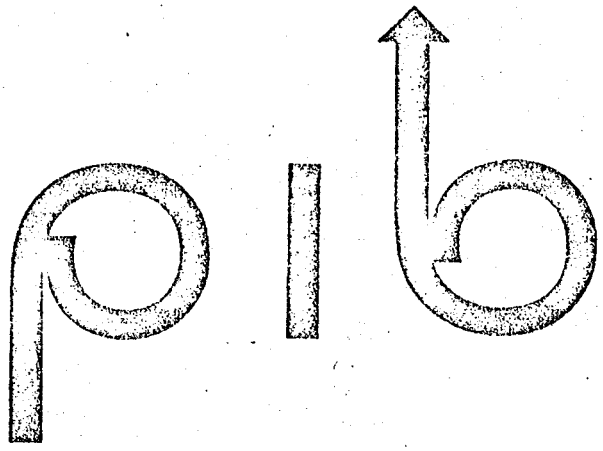
JOEHNSEN

.37

THE COMPUTER  CENTER

POLYTECHNIC INSTITUTE OF BROOKLYN

CALL/360

USERS

MANUAL

OCTOBER, 1970

XCC101, CALL 360

TERMINAL KEYS

Most of the IBM 2741 terminals are equipped with APL balls and APL transliteration charts. The following table indicates instances in which the desired CALL/360 character (right column) is obtained by punching the APL character shown on the left.

APL Symbol	CALL/360 Symbol	APL Symbol	CALL/360 Symbol
⊐	)	⊤	T
←	=	○	@
⊏	(	⍴	⍴
×	*	÷	/
¨	"	⍲	⍲
−	_	⊥	⊥
∨	|	∩	∩
∧	¢	⌊	⌊
→	→	∈	∈
\	%	◁	◁
⌈	⌈	△	△
~	¬	S	$
↓	!		
⍵	⍵		
⊃	⊃		
↑	↑		
⊂	¢		
∘	∘		
⌷	#		

1. Uppercase letters, digits, and special characters represent information that must appear as shown.
2. Boldface type represents the abbreviation of the command.
3. Lowercase letters in italics represent information that is supplied by the user.
4. A series of three periods indicates that a variable number of items may be entered.

TYPING COMMANDS

Command Format: command *entry, entry, . . . , entry*

1. Type the command word(s) and strike the space bar.
2. Type each entry.
3. Type a comma after each entry if more than one entry is typed. Spacing after the comma is optional.
4. Press the RETURN key at the end of each line.

SYSTEM COMMUNICATION

Change User without Disconnecting	LOGON
Disconnect from System	OFF
Test Transmission	ECHO *characters to test*
Request Information	HELP
Print User Status	STATUS
Request CPU and Connect Time (Month to Date)	TIME

USER IDENTIFICATION

User Number

Each user is identified by a user number established by the service center and a password established by the user.

User Number	6 characters—3 alpha followed by 3 numeric
User Group Code	First 4 characters of user number
Password	Up to 8 characters—alpha, numeric, and special characters, except leading blanks

Change Password — PASSWORD *characters*

Pressing RETURN without typing a password causes the old password to be retained.

Name Program	NAME *program name*

Program name—maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break (_) characters.

Identify Language	ENTER *language name*

Language name:
BASIC — short precision
BASICL — long precision

Identify Data File	FILE *filename, length*
List Program	LIST LIST *line number* LIST-NO-HEADER LIST-NO-HEADER *line number*
Store Program	SAVE
Set Page Width	WIDTH *no. chars. per line*

18 characters minimum, 255 characters maximum. Unless otherwise specified, width is 72 characters.

PROGRAM EXECUTION

Execute Program in Work Area	RUN
Execute Program from User Program Library	RUN *program name*
Execute Program from Shared Program Library	RUN **program name* RUN ***program name* RUN ****program name*

PAPER TAPE

Allow Paper Tape Input from Teletype	TAPE
Restore Keyboard Functions	KEY

*LIBRARY

Contains names of programs shared by all users with the same user group code. (Group code first four characters in the user number.)

**LIBRARY

Contains the names of programs supplied by users and shared with all users of the system.

Store Program Name	POOL **program name* POOL ***program name*
Protect Program	PROTECT **program name* PROTECT ***program name*
Remove Protection	ALLOW **program name* ALLOW ***program name*
Delete Program Name	PULL **program name* PULL ***program name*
Retrieve Program	LOAD **program name* LOAD ***program name*
Execute Program	RUN **program name* RUN ***program name*
Print List of Program Names	CATALOG * CATALOG **

***LIBRARY

Contains programs supplied by the installation to be shared by all users of the CALL/360-OS System.

Retrieve Program	LOAD ****program name*
Execute Program	RUN ****program name*
Print List of Program Names	CATALOG ***

NOTE: Programs loaded or run from the Shared Libraries identify the language to be used. If applicable, data files must be provided for the Shared Library Programs. The FILE command should be used to create a data file entry in the user's library which corresponds to the filename to be opened by the Library Program.

USER PROGRAM LIBRARY

Store Program	SAVE
Lock Program	LOCK *program name*
Unlock Program	UNLOCK *program name*
Delete Program	PURGE *program name*
Retrieve Program	LOAD *program name*
Execute Program	RUN *program name*
Print List of Programs	CATALOG
Print List of Programs and Descriptive Data	CATALOG ALL

DATA FILES

Allocate Data File	FILE *filename* FILE *filename, length*

Filename—maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break characters (_).

Length—maximum number of disk storage units to be allocated for the file. Disk storage unit length is 3440 bytes. Maximum file length is 100 units. Default is 4.

Increase Data File	FILE *filename, length*

Filename—must correspond to a valid data file in user's library.

Length—used to increase the number of storage units allocated for storage of a data file.

Lock File	LOCK *filename*
Unlock File	UNLOCK *filename*
Delete File	PURGE *filename*

PROGRAM MODIFICATION

Renumber Statements	RENUMBER *new line no., old line no., increment*
Delete Single Statements	DELETE *line no., line no., . . .*

Line numbers must be entered in ascending sequence.

Delete Group of Statements	DELETE *line no.* THRU *line no., . . .*

Several groups of statements and/or single statements may be deleted with one DELETE command.

Extract Single Statements	EXTRACT *line no., line no., . . .*

Line numbers must be entered in ascending sequence.

Extract Group of Statements	EXTRACT *line no.* THRU *line no., . . .*

Several groups of statements and/or single statements may be extracted with one EXTRACT command.

Merge Programs	MERGE *main program name, subprogram name 1, line no. before insertion 1, subprogram name 2, line no. before insertion 2, . . .*

CORRECTION PROCEDURES

TYPING CORRECTIONS—2741

Correct Current Line

1. Backspace to the point of the error.
2. Press the ATTN key.
3. System prints underscore and spaces down one line.
4. Type the correct character and all characters backspaced over. Example:
```
CLAER
  EAR
```

Delete Current Line

1. Press the SHIFT key and type a degree symbol (°, upshift J key).
2. Press the ATTN key.
3. System prints DELETED and returns the carrier to the next line. The line has been erased. Example:
```
RUN PROG° DELETED
```

TYPING CORRECTIONS – TELETYPE

Correct Current Line

1. Press the SHIFT key and type the back-arrow (←, upshift letter O key) once for the incorrect character and once for each character following it.
2. Type the correct character and retype the remainder of the line. Example: CLAER ←←←EAR

Delete Current Line

1. Hold down the CTRL key and press the X key.
2. The system prints DELETED and returns the carrier to the next line. Example:
```
RUN PROG DELETED
```

PROGRAM STATEMENT CORRECTIONS

Delete Program Statement

1. Type the line number.
2. Press the RETURN key.
3. No system response prints. The line and its number have been deleted. Example:
```
10 X=Y+Z;
10
```

Replace Program Statement

1. Type the line number to be replaced, followed by the new program statement.
2. No system response prints. The last statement typed replaces the previous statement of the same number. Example:
```
10 A=B+C
20 ...
30 ...
10 X=B+C
```

Insert Program Statement

1. Number the statement to be inserted with a number whose value is between the line number preceding the insertion and the line number following the insertion.
2. Type the line number followed by the statement to be inserted. Example:
```
40 A=10.2;
50 B=A+4.5;
60 PUT DATA (A,B,C);
55 C=A2 + B2;
```

LANGUAGE ELEMENTS

PL/I STATEMENT IDENTIFIERS (KEYWORDS)

BEGIN	FORMAT	PROCEDURE
CALL	GET	(PROC)
CLOSE	GO TO (GOTO)	PUT
DECLARE (DCL)	IF	RETURN
DO	ON	REVERT
END	OPEN	STOP

PL/I CHARACTER SET

Characters Used for Identifiers

First character of identifier must be one of these characters:

- A through Z
- $ dollar sign
- @ at sign
- # number or pound sign

Also:

- 0 through 9
- _ break character

Other Characters

- ' quotation mark
- (left parenthesis
-) right parenthesis
- , comma
- ; semicolon
- : colon
- % percent sign
- ? question mark
- blank
- backspace
- carrier return

Characters Used for Arithmetic Expressions

- + addition
- - subtraction
- * multiplication
- / division
- ** or ↑ exponentiation
- . decimal point in constants

Characters Used in Comparison Operations

- > greater than
- >= or ≥ greater than or equal to
- = equal to
- ¬= or ≠ not equal to
- < less than
- <= or ≤ less than or equal to
- ¬> not greater than
- ¬< not less than

Characters Used in Logical Operations

- | or
- & and

valid only in an IF statement and in the WHILE clause of a DO statement

Input/Output Statements

IMPLICIT I/O UNITS	
File Reference Number (j)	Device
1 - 4	Data file
5	Terminal input
6	Terminal output
7 - 99	Data File

END FILE Statement

END FILE i

READ Statement

Formatted READ Statement

READ (i,f[,END=n_1] [,ERR=n_2])[list]

List-Directed READ Statement

READ (i,* [,END=n_1] [,ERR=n_2])[list]

Unformatted READ Statement

READ (i [,END=n_1] [,ERR=n_2])[list]

REWIND Statement

REWIND i

WRITE Statement

Formatted WRITE Statement

WRITE (i,f)[list]

List-Directed WRITE Statement

WRITE (i,*)[list]

Unformatted WRITE Statement

WRITE (i)[list]

INTEGER Type Statement

See *Explicit Specification Statements*

PAUSE Statement

PAUSE [o]
PAUSE ['message']

READ Statement

See *Input/Output Statements*

REAL Type Statement

See *Explicit Specification Statements*

RETURN Statement

RETURN [m]

NOTE: Unit numbers may be j's.

REWIND Statement

See *Input/Output Statements*

STOP Statement

STOP [o]

WRITE Statement

See *Input/Output Statements*

FORTRAN SUBPROGRAMS

Arithmetic Statement Function

name (d_1, d_2, . . .) = a

FUNCTION Subprogram

[type] FUNCTION name [*s] (d_1, [,d_2, . . .])

where type *is* COMPLEX, INTEGER, *or* REAL

SUBROUTINE Subprogram

SUBROUTINE name [(d_1, d_2, . . .)]

Trigonometric Functions

Sine	SIN, DSIN, CSIN, CDSIN
Cosine	COS, DCOS, CCOS, CDCOS
Tangent	TAN, DTAN
Cotangent	COTAN, DCOTAN
Arcsine	ARSIN, DARSIN
Arccosine	ARCOS, DARCOS
Arctangent	ATAN,ATAN2(2),DATAN,DATAN2(2)
Hyperbolic Sine	SINH, DSINH
Hyperbolic Cosine	COSH, DCOSH
Hyperbolic Tangent	TANH, DTANH

Other Mathematical Functions

Absolute Value	IABS, ABS, DABS, CABS, CDABS
Error Function	ERF, DERF
Complemented Error Function	ERFC, DERFC
Exponential	EXP, DEXP, CEXP, CDEXP
Gamma Function	GAMMA, DGAMMA
Log of Gamma Function	ALGAMA, DLGAMA
Logarithm–Common	ALOG10, DLOG10
Logarithm–Natural	ALOG, DLOG, CLOG, CDLOG
Maximum Value	AMAX0(≥2),AMAX1(≥2),MAX0(≥2), MAX1(≥2),DMAX1(≥2)
Minimum Value	AMIN0(≥2),AMIN1(≥2), MIN0(≥2), MIN1(≥2),DMIN1(≥2)
Modular Arithmetic	MOD(2), AMOD(2), DMOD(2)
Square Root	SQRT, DSQRT, CSQRT, CDSQRT
Truncation	INT, AINT, IDINT

Conversion Functions

Floating-Point Conversion	FLOAT, DFLOAT
Fixed-Point Conversion	IFIX, HFIX
Double-Precision Conversion	DBLE
Complex Conversion	CMPLX(2), DCMPLX(2)
Positive Difference	DIM(2), IDIM(2)
Sign Transfer	SIGN(2), ISIGN(2), DSIGN(2)
Obtain Significant Part of Real * 8 Argument	SNGL
Obtain Real Part of Complex * 8 Argument	REAL
Obtain Imaginary Part of Complex * 8 Argument	AIMAG
Obtain Conjugate of Complex Argument	CONJG, DCONJG

NOTE: Number of arguments for function specified in parentheses only if different from one argument.

Out-of-Line Service Subprograms

CALL OPEN (i, filename, mode)
where mode *is* INPUT, OUTPUT, SYSIN, *or* SYSPRINT
CALL CLOSE (i1, i2, i3, . . .)
CALL DVCHK (j)
CALL OVERFL (j)
CALL EXIT

NOTE: Unit numbers may be j's.

CONVENTIONS USED BELOW IN FORTRAN STATEMENTS AND SUBPROGRAMS

CHARACTER	MEANING
a	an arithmetic expression
b	an initial value
d	a dummy argument or *
e	an executable statement
f	a FORMAT statement number
i	an unsigned integer constant
j	a nonsubscripted integer variable of length 4
jj	a nonsubscripted integer variable of length 2 or 4
k	a constant
l	a relational expression
m	an integer constant or variable of length 4 used to denote the mth statement number in the SUBROUTINE statement argument list
n	an executable statement number
o	a string of one to five digits
*s	a length specification
s*	a replication factor
t	a variable or array name or an array declarator
v	an argument (may be a, k, &n, w, or x)
w	a variable, array, or function name
x	a variable name or an array element
y	a nonsubscripted or integer subscripted variable name or an array name
z	an array name
filename	a data file name
list	an input/output list
name	a symbolic name
...	repetition
[ ]	optional

FORTRAN STATEMENTS

Arithmetic Assignment Statement

x = a

ASSIGN Statement

ASSIGN n TO j

CALL Statement

CALL name [(v_1,v_2,...)]

COMMON Statement

COMMON t_1, t_2, ...

COMPLEX Type Statement

See *Explicit Specification Statements*

CONTINUE Statement

CONTINUE

DATA Initialization Statement

DATA y_1,y_2,.../[s_1*]k_1, [s_2*]k_2,.../,...

DIMENSION Statement

DIMENSION z_1(i_1,...), z_2(i_2,...),...

NOTE: Array dimensions may be j's in a subprogram.

DO Statement

DO n jj = i_1,i_2[,i_3]

NOTE: DO parameters may also be unsigned jjs.

END Statement

END

END FILE Statement

See *Input/Output Statements*

EQUIVALENCE Statement

EQUIVALENCE (x_1, x_2 ,...),...

Explicit Specification Statements

type [*s] w_1[*s_1] [(i_1,...)] [/b_1,.../],w_2 [*s_2] [(i_1,...)] [/b_2,.../],...

where type *is* COMPLEX, INTEGER, *or* REAL

EXTERNAL Statement

EXTERNAL $name_1$, $name_2$,...

FORTRAN STATEMENTS (Continued)

FORMAT Statement

f FORMAT (list)

where list *is composed of format codes*

Format Type	Code	General Form
Integer	I	nIw
Real	F,E,D	nFw.d
		nEw.d
		nDw.d
General	G	nGw.s
Scale Factor	P	pP
Hollerith	A,H	nAw,wH,or '...'
Blank	X	nwX
Tab	T	nTr

integer constants only:
- d = length of decimal field
- n = number of repetitions of code
- p = scale factor
- r = character position in record
- s = number of significant digits
- w = field length

NOTE: Either a comma or a slash may be used as a separator between format codes. A slash indicates the beginning of a new record.

The first character of a print record is treated as a carrier control character. The general form is 1Hx where x may be one of the following:

blank	*Advance one line before printing.*
0	*Advance two lines before printing.*
+	*No advance.*

GO TO Statements

Assigned GO TO Statement
GO TO j[,(n_1 , n_2 ,...)]

Computed GO TO Statement
GO TO (n_1 , n_2 ,...), jj

Unconditional GO TO Statement
GO TO n

IF Statements

Arithmetic IF Statement
IF (a) n_1 , n_2 , n_3

NOTE: a may not be complex.

Logical IF Statement
IF (l) e

NOTE: e may not be a DO or a logical IF statement.

KEY TO SYMBOLS IN COMMAND FORMATS

1. Uppercase letters, digits, and special characters represent information that must appear as shown.
2. Boldface type represents the abbreviation of the command.
3. Lowercase letters in italics represent information that is supplied by the user.
4. A series of three periods indicates that a variable number of items may be entered.

TYPING COMMANDS

Command format: command *entry, entry , . . . , entry*

1. Type the command word(s) and strike the space bar.
2. Type each entry.
3. Type a comma after each entry if more than one entry is typed. Spacing after the comma is optional.
4. Press the RETURN key at the end of each line.

SYSTEM COMMUNICATION

Change User without Disconnecting	**LOGON**
Disconnect from System	**OFF**
Test Transmission	**ECHO** *characters to test*
Request Information	**HELP**
Print User Status	**STATUS**
Request CPU and Connect Time (Month to Date)	**TIME**

USER IDENTIFICATION

User Number

Each user is identified by a user number established by the service center and a password established by the user.

User Number	6 characters—3 alpha followed by 3 numeric
User Group Code	First 4 characters of user number
Password	Up to 8 characters—alpha, numeric, and special characters, except leading blanks

Change Password — **PASSWORD** *characters*

Pressing RETURN without typing a password causes the old password to be retained.

PROGRAM ENTRY

Clear Work Area	**CLEAR**
Name Program	**NAME** *program name*

Program name maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break characters (_).

Identify Language	**ENTER** *language name*

Language name: PL/I
BASIC assumed if no ENTER typed.

Identify Data File	**FILE** *filename, length*
List Program	**LIST** **LIST** *line number* **LIST-NO-HEADER** **LIST-NO-HEADER** *line number*
Store Program	**SAVE**
Set Page Width	**WIDTH** *no. chars. per line*

18 characters minimum, 255 characters maximum. Unless otherwise specified, width is 72 characters.

PROGRAM EXECUTION

Execute Program in Work Area	**RUN**
Execute Program from User Program Library	**RUN** *program name*
Execute Program from Shared Program Library	**RUN** **program name* **RUN** ***program name* **RUN** ****program name*

PAPER TAPE

Allow Paper Tape Input from Teletype	**TAPE**
Restore Keyboard Functions	**KEY**

SHARED PROGRAM LIBRARIES

***Library**

Contains names of programs shared by all users with the same user group code. (Group code—first four characters in the user number.)

****Library**

Contains the names of programs supplied by users and shared with all users of the system.

Store Program Name	**POOL** **program name* **POOL** ***program name*
Protect Program	**PROTECT** **program name* **PROTECT** ***program name*
Remove Protection	**ALLOW** **program name* **ALLOW** ***program name*
Delete Program Name	**PULL** **program name* **PULL** ***program name*
Retrieve Program	**LOAD** **program name* **LOAD** ***program name*
Execute Program	**RUN** **program name* **RUN** ***program name*
Print List of Program Names	**CATALOG** * **CATALOG** **

*****Library**

Contains programs supplied by the installation to be shared by all users of the CALL/360-OS System.

Retrieve Program	**LOAD** ****program name*
Execute Program	**RUN** ****program name*
Print List of Program Names	**CATALOG** ***

NOTE: Programs loaded or run from the Shared Libraries identify the language to be used. If applicable, data files must be provided for the Shared Library Programs. The FILE command should be used to create a data file entry in the user's library which corresponds to the file name to be opened by the Library Program.

USER PROGRAM LIBRARY

Store Program	**SAVE**
Lock Program	**LOCK**
Unlock Program	**UNLOCK**
Delete Program	**PURGE**
Retrieve Program	**LOAD**
Execute Program	**RUN**
Print List of Programs	**CATALOG**
Print List of Programs and Descriptive Data	**CATALOG ALL**

DATA FILES

Allocate Data File	FILE *filename* FILE *filename, length*

Filename—maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break characters (_).

Length—maximum number of disk storage units to be allocated for the file. Disk storage unit length is 3440 bytes. Maximum file length is 100 units. Default is 4.

Increase Data File	FILE *filename, length*

Filename—must correspond to a valid data file in user's library.

Length—used to increase the number of storage units allocated for storage of a data file.

Lock File	LOCK *filename*
Unlock File	UNLOCK *filename*
Delete File	PURGE *filename*

PROGRAM MODIFICATION

Renumber Statements	RENUMBER *new line no., old line no., increment*
Delete Single Statements	DELETE *line no., line no.,...*

Line numbers must be entered in ascending sequence.

Delete Group of Statements	DELETE *line no.* THRU *line no.,...*

Several groups of statements and/or single statements may be deleted with one DELETE command.

Extract Single Statements	EXTRACT *line no., line no.,...*

Line numbers must be entered in ascending sequence.

Extract Group of Statements	EXTRACT *line no.* THRU *line no.,...*

Several groups of statements and/or single statements may be extracted with one EXTRACT command.

Merge Programs	MERGE *main program name, subprogram name 1, line no. before insertion 1, subprogram name 2, line no. before insertion 2,...*

CORRECTION PROCEDURES

TYPING CORRECTIONS—2741

Correct Current Line

1. Backspace to the point of the error.
2. Press the ATTN key.
3. System prints underscore and spaces down one line.
4. Type the correct character and all characters backspaced over. Example:
```
CLAER
 EAR
```

Delete Current Line

1. Press the SHIFT key and type a degree symbol (°, upshift J key).
2. Press the ATTN key.
3. System prints DELETED and returns the carrier to the next line. The line has been erased. Example:
```
RUN PROG° DELETED
```

TYPING CORRECTIONS – TELETYPE

Correct Current Line

1. Press the SHIFT key and type the back-arrow (←, upshift letter O key) once for the incorrect character and once for each character following it.
2. Type the correct character and retype the remainder of the line. Example: CLAER ←←←EAR

Delete Current Line

1. Hold down the CTRL key and press the X key.
2. The system prints DELETED and returns the carrier to the next line. Example:
```
RUN PROG DELETED
```

PROGRAM STATEMENT CORRECTIONS

Delete Program Statement

1. Type the line number.
2. Press the RETURN key.
3. No system response prints. The line and its number have been deleted. Example:
```
10 X=Y+Z;
10
```

Replace Program Statement

1. Type the line number to be replaced, followed by the new program statement.
2. No system response prints. The last statement typed replaces the previous statement of the same number. Example:
```
10 A=B+C
20 ...
30 ...
10 X=B+C
```

Insert Program Statement

1. Number the statement to be inserted with a number whose value is between the line number preceding the insertion and the line number following the insertion.
2. Type the line number followed by the statement to be inserted. Example:
```
40 A=10.2;
50 B=A+4.5;
60 PUT DATA (A,B,C);
55 C=A2 + B2;
```


LANGUAGE ELEMENTS

PL/I STATEMENT IDENTIFIERS (KEYWORDS)

BEGIN	FORMAT	PROCEDURE (PROC)
CALL	GET	PUT
CLOSE	GO TO (GOTO)	RETURN
DECLARE (DCL)	IF	REVERT
DO	ON	STOP
END	OPEN	

PL/I CHARACTER SET

Characters Used for Identifiers

First character of identifier must be one of these characters.

- A through Z
- $ dollar sign
- @ at sign
- # number or pound sign

- 0 through 9
- _ break character

Other Characters

- ' quotation mark
- (left parenthesis
-) right parenthesis
- . comma
- ; semicolon
- : colon
- % percent sign
- ? question mark
- blank
- backspace
- carrier return

Characters Used for Arithmetic Expressions

- + addition
- – subtraction
- * multiplication
- / division
- ** or ↑ exponentiation
- . decimal point in constants

Characters Used in Comparison Operations

- > greater than
- >= or ≥ greater than or equal to
- = equal to
- ¬= or ≠ not equal to
- < less than
- <= or ≤ less than or equal to
- ¬> not greater than
- ¬< not less than

Characters Used in Logical Operations

|	or	valid only in an IF statement and in the WHILE clause of a DO statement
&	and	

PL/I STATEMENTS (Continued)

PROCEDURE Statement

label: PROCEDURE;
label: PROCEDURE *(parameter list) [attributes];*
label: PROCEDURE *[attributes];*

```
SUBR: PROCEDURE(TIME,PLACE,DATE);
```

PUT Statement

PUT [FILE *(filename)*] [*data specification*] [SKIP [*(expression)*]]

where *data specification* is any one of

- LIST *(data list)*
- DATA *(data list)*
- EDIT *(data list) (format list)*

NOTE: If FILE (filename) is not specified, SYSPRINT is assumed.

```
PUT LIST (C,D);
PUT DATA (X, ARB);
PUT FILE (OUTFILE) EDIT (PROFIT,
  LOSS)(F(5), X(3), F(4,2));
```

RETURN Statement

RETURN;
RETURN *(expression);*

```
RETURN (X**2 + Y**2);
```

REVERT Statement

REVERT *condition;*

```
REVERT ZERODIVIDE;
```

STOP Statement

STOP;

PL/I ON-CONDITIONS

FIXEDOVERFLOW	raised when the total number of decimal digits in the result exceeds approximately 9
OVERFLOW	raised when exponent of floating-point number exceeds allowable maximum of 10^{75}
UNDERFLOW	raised when exponent of floating-point number is less than allowable minimum of 10^{-78}
ZERODIVIDE	raised when an attempt is made to divide by zero
ENDFILE *(filename)*	raised on EOF
ERROR	raised when program is forced to terminate because of some error condition. Alternate action may be specified.

PL/I DATA SPECIFICATION

The data specification of a GET or PUT statement consists of the transmission mode, the data list and, for edit-directed I/O, the format list.

Data List *(element [,element, . . .])*

Input element can be one of the following:

1. scalar name (must be unsubscripted for DATA (data list) input)
2. array name (must be unsubscripted for DATA (data.list) input)
3. pseudo-variable
4. pseudo-array
5. repetitive specification of any of the above items

Output element can be one of the following:

1. scalar name or scalar expression (scalar expression invalid for DATA (data list) output)
2. array name
3. repetitive specification of any of the above items

Format List *(format items)*

Editing format items:

F(w[,d] [,p])	fixed-point numeric
E(w,d[,s])	floating-point numeric
A[(w)]	character-string
C(real format item [,real format item])	complex numeric

Control format items:

X(w)	spacing between characters
SKIP[(w)]	if no w, default is 1; if w ≤ 0, space suppression (permits overprinting line); if w > 0, creates w-1 blank lines
COLUMN(w)	specifies that the wth column is to be next print position

Remote format item:

R (statement-label constant [,statement-label variable])	statement-label designator

KEY TO SYMBOLS

w	Length of field in characters
d	Number of positions after decimal point
s	Number of significant digits to appear (float)
p	Scale factor (fixed)
real format item	Either an E or F format item

PL/I PRIORITY OF OPERATIONS

prefix operator exponentiation	+ or − ** or ↑	highest priority
division multiplication	/ *	second priority
addition subtraction	+ −	third priority
comparison	>,>= or ≥, =, ≠ or ¬=, <, <= or ≤, ¬>, ¬<	fourth priority
and	&	fifth priority
or	|	sixth priority

NOTE: Parentheses modify priorities; innermost parentheses are evaluated first. Expressions involving prefix operators and exponentiation are evaluated from right to left; all other operations are performed from left to right.

PL/I BUILT-IN FUNCTIONS & LIBRARY SUBROUTINES

Character-String Built-In Functions

CHAR *(expression [,size])*
SUBSTR *(string, i [,j])*

Arithmetic Built-In Functions

ABS(x)	MAX($x_1, x_2, \ldots, x_n$)
CEIL(x)	MIN($x_1, x_2, \ldots, x_n$)
COMPLEX(x, y)	MOD(x_1, x_2)
CONJG(x)	REAL(x)
FLOOR(x)	SIGN(x)
IMAG(x)	TRUNC(x)

Mathematical Built-In Functions

ATAN(x[,y])	EXP(x)	SINH(x)
ATANH(x)	LOG(x)	SQRT(x)
COS(x)	LOG2(x)	TAN(x)
COSH(x)	LOG10(x)	TANH(x)
ERF(x)	SIN(x)	

Array Manipulation Built-In Functions

DIM(x,n)	LBOUND(x,n)	PROD(x)
HBOUND(x,n)	POLY(a,x)	SUM(x)

Miscellaneous Built-In Function

DATE

Library Subroutine

$RESET *(filename)*

PL/I ATTRIBUTES (KEYWORDS)

AUTOMATIC(AUTO)	FILE	OUTPUT
CHARACTER (CHAR)	FIXED	PRINT
COMPLEX (CPLX)	FLOAT	REAL
ENTRY	INPUT	RETURNS
ENVIRONMENT(ENV)	LABEL	STATIC

PL/I ENVIRONMENT OPTIONS (KEYWORDS)

DISK	EXTERNAL	INTERNAL

1. Semicolon terminates statement.
2. Colon separates labels from statement body.
3. Comma separates items in a list.
4. /* */ character pairs indicate beginning and end of comments.
5. Single quote marks enclose string constants.
6. Identifiers, except for certain keywords, are limited to eight characters maximum.
7. If default attributes of an identifier are not desired, the identifier must be declared before its first use.
8. Carrier return is treated as a blank except in character strings.
9. At least one blank must separate a line number from the first character of a statement.

KEY TO SYMBOLS IN PL/I STATEMENT FORMATS

1. Uppercase letters, quotation marks, semicolons, colons, commas, and apostrophes represent information that must appear exactly as shown.
2. Lowercase letters in italics represent information that is supplied by the user.
3. A series of three periods indicates that a variable number of items may be included in a list.
4. The appearance of one or more items in sequence indicates that the items, or their replacements, should appear in the specified order.
5. Items shown in { } represent alternatives to be selected.
6. Items shown in [ ] represent options that may be omitted.

PL/I STATEMENTS

Assignment Statement

Option 1. Scalar Assignment

{*scalar-variable* / *pseudo-variable*} = *scalar-expression;*

Option 2. Array Assignment

{*array* / *pseudo-array*} = {*array-expression;* / *scalar-expression;*}

BEGIN Statement

[label:] BEGIN;

CALL Statement

CALL *entry name [(argument list)];*

```
CALL TOTAL (AB, 20, 20, TAD);
```

CLOSE Statement

CLOSE FILE *(filename);*

DECLARE Statement

DCL *entry name* ENTRY *[(attribute list)];*
DCL *entry name* ENTRY *[(attribute list)]* [RETURNS *(attribute list)*];
DCL *name attributes;*
DCL *(name, name, . . .) attributes;*
DCL *name (dimension) other attributes;*

```
DCL A (10,10);
DCL A (-5:5);
DCL A (*);
DCL (I,J) STATIC;
DCL B CHAR (5), D CHAR (*);
DCL AX LABEL;
DCL SUB ENTRY (FLOAT (6), REAL (5,2));
DCL FCN RETURNS (FLOAT);
DCL TAX FILE OUTPUT;
DCL INFILE FILE ENV (INTERNAL);
```

DO Statement

Option 1.

DO;

Option 2.

DO WHILE *(scalar expression);*
The scalar expression must be either Boolean or logical.

Option 3.

DO {*pseudo-variable* / *variable*} = *specification;*

A specification has the following format:

expression-1 [TO *expression-2* [BY *expression-3*]] / [BY *expression-3* [TO *expression-2*]]
[WHILE *expression-4*]

```
DO WHILE (S**2<60);
DO I = 1 TO 15 WHILE (A=B);
DO J = 10 BY -1 WHILE (DED<EST);
DO A = 20 BY 3 TO 60;
```

END Statement

END *[label];*

FORMAT Statement

label: FORMAT *(format list);*

```
PUT EDIT (A, J, K)(R (COMMON));
COMMON: FORMAT(A(5),F(5,2),X(3),F(10,0));
```


GET Statement

GET [FILE *(filename)*] LIST *(data list);*
GET [FILE *(filename)*] DATA [*(data list)*];
GET [FILE *(filename)*] EDIT *(data list) (format list);*
NOTE: If FILE (filename) is not specified, SYSIN is assumed.

```
GET LIST (X,Y,Z);
GET DATA (A,B);
GET FILE(INFILE) EDIT(X,Y,Z)(A(5),
  F(5,2),A(10));
```

GO TO Statement

GO TO *statement-label-constant;*
GO TO *statement-label-variable;*

IF Statement

IF *condition* THEN *statement;*
IF *condition* THEN *statement 1;* ELSE *statement 2;*
IF *condition 1* THEN *statement 1;* ELSE IF *condition 2* THEN *statement 2;* ELSE IF *condition 3* THEN *statement 3;* etc.
NOTE: DO group or begin block may be substituted for statement.

```
IF PROFIT<0 THEN GO TO LOSS;
IF A>B THEN A=C; ELSE A=D;
IF A=1 THEN GO TO LAB1;
  ELSE IF A=2 THEN GO TO LAB2;
  ELSE IF A=3 THEN GO TO LAB3;
IF A+B<7 THEN DO; A=2; B=7; GO TO L;
  END;
```

NULL Statement

[label:];

ON Statement

ON *condition action-specification;*
ON *condition* SYSTEM;
ON *condition;*
NOTE: The DO statement may not be substituted for action-specification.

```
ON ZERODIVIDE CALL ANALYSIS;
ON OVERFLOW SYSTEM;
ON UNDERFLOW;
```

OPEN Statement

OPEN FILE *(filename)* [TITLE *(character expression)* [INPUT | OUTPUT];

```
OPEN FILE(TAX) TITLE('TWEEK')OUTPUT;
```

KEY TO SYMBOLS IN COMMAND FORMATS

1. Uppercase letters, digits, and special characters represent information that must appear as shown.
2. Boldface type represents the abbreviation of the command.
3. Lowercase letters in italics represent information that is supplied by the user.
4. A series of three periods indicates that a variable number of items may be entered.

TYPING COMMANDS

Command Format: command *entry, entry, . . . , entry*

1. Type the command word(s) and strike the space bar.
2. Type each entry.
3. Type a comma after each entry if more than one entry is typed. Spacing after the comma is optional.
4. Press the RETURN key at the end of each line.

SYSTEM COMMUNICATION

Change User without Disconnecting	**LOGON**
Disconnect from System	**OFF**
Test Transmission	**ECHO** *characters to test*
Request Information	**HELP**
Print User Status	**STATUS**
Request CPU and Connect Time (Month to Date)	**TIME**

USER IDENTIFICATION

User Number

Each user is identified by a user number established by the service center and a password established by the user.

User Number	6 characters—3 alpha followed by 3 numeric
User Group Code	First 4 characters of user number
Password	Up to 8 characters—alpha, numeric, and special characters, except leading blanks

Change Password — **PASSWORD** *characters*

Pressing RETURN without typing a password causes the old password to be retained.

Clear Work Area	**CLEAR**
Name Program	**NAME** *program name*

Program name—maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break (_) characters.

Identify Language	**ENTER** *language name*

Language name:
BASIC — short precision
BASICL — long precision

Identify Data File	**FILE** *filename, length*
List Program	**LIST** **LIST** *line number* **LIST-NO-HEADER** **LIST-NO-HEADER** *line number*
Store Program	**SAVE**
Set Page Width	**WIDTH** *no. chars. per line*

18 characters minimum, 255 characters maximum. Unless otherwise specified, width is 72 characters.

PROGRAM EXECUTION

Execute Program in Work Area	**RUN**
Execute Program from User Program Library	**RUN** *program name*
Execute Program from Shared Program Library	**RUN** **program name* **RUN** ***program name* **RUN** ****program name*

PAPER TAPE

Allow Paper Tape Input from Teletype	**TAPE**
Restore Keyboard Functions	**KEY**

*LIBRARY

Contains names of programs shared by all users with the same user group code. (Group code first four characters in the user number.)

**LIBRARY

Contains the names of programs supplied by users and shared with all users of the system

Store Program Name	**POOL** **program name* **POOL** ***program name*
Protect Program	**PROTECT** **program name* **PROTECT** ***program name*
Remove Protection	**ALLOW** **program name* **ALLOW** ***program name*
Delete Program Name	**PULL** **program name* **PULL** ***program name*
Retrieve Program	**LOAD** **program name* **LOAD** ***program name*
Execute Program	**RUN** **program name* **RUN** ***program name*
Print List of Program Names	**CATALOG** * **CATALOG** **

***LIBRARY

Contains programs supplied by the installation to be shared by all users of the CALL/360 OS System.

Retrieve Program	**LOAD** ****program name*
Execute Program	**RUN** ****program name*
Print List of Program Names	**CATALOG** ***

NOTE: Programs loaded or run from the Shared Libraries identify the language to be used. If applicable, data files must be provided for the Shared Library Programs. The FILE command should be used to create a data file entry in the user's library which corresponds to the filename to be opened by the Library Program.

USER PROGRAM LIBRARY

Store Program	**SAVE**
Lock Program	**LOCK** *program name*
Unlock Program	**UNLOCK** *program name*
Delete Program	**PURGE** *program name*
Retrieve Program	**LOAD** *program name*
Execute Program	**RUN** *program name*
Print List of Programs	**CATALOG**
Print List of Programs and Descriptive Data	**CATALOG ALL**

DATA FILES

Allocate Data File	FILE *filename* FILE *filename, length*

Filename—maximum 8 characters. First character must be alphabetic or #, $, or @. Others may be any combination of these characters, numerals, and break (_) characters.

Length—the number of disk storage units to be allocated for the file. The length of a disk storage unit is 3440 bytes. Maximum file length is 100 units. Default is 4.

Increase Data File	FILE *filename, length*

Filename—must correspond to a valid data file in user's library.

Length—used to increase the number of storage units allocated for storage of a data file.

Lock File	LOCK *filename*
Unlock File	UNLOCK *filename*
Delete File	PURGE *filename*

PROGRAM MODIFICATION

Renumber Statements	RENUMBER *new line no., old line no., increment*
Delete Single Statements	DELETE *line no., line no., . . .*

Line numbers must be entered in ascending sequence.

Delete Group of Statements	DELETE *line no.* THRU *line no., . . .*

Several groups of statements and/or single statements may be deleted with one DELETE command.

Extract Single Statements	EXTRACT *line no., line no., . . .*

Line numbers must be entered in ascending sequence.

Extract Group of Statements	EXTRACT *line no.* THRU *line no., . . .*

Several groups of statements and/or single statements may be extracted with one EXTRACT command.

Merge Programs	MERGE *main program name, subprogram name 1, line no. before insertion 1, subprogram name 2, line no. before insertion 2, . . .*

CORRECTION PROCEDURES

TYPING CORRECTIONS – 2741

Correct Current Line

1. Backspace to the point of the error.
2. Press the ATTN key.
3. System prints underscore and spaces down one line.
4. Type the correct character and all characters backspaced over. Example: CLAER
   EAR

Delete Current Line

1. Press the SHIFT key and type a degree symbol (°, upshift J key).
2. Press the ATTN key.
3. System prints DELETED and returns the carrier to the next line. The line has been erased. Example:
   RUN PROG° DELETED

TYPING CORRECTIONS – TELETYPE

Correct Current Line

1. Press the SHIFT key and type the back-arrow (←, upshift letter O key) once for the incorrect character and once for each character following it.
2. Type the correct character and retype the remainder of the line. Example: CLAER←←EAR

Delete Current Line

1. Hold down the CTRL key and press the X key.
2. The system prints DELETED and returns the carrier to the next line. Example:
   RUN PROG DELETED

PROGRAM STATEMENT CORRECTIONS

Delete Program Statement

1. Type the line number.
2. Press the RETURN key.
3. No system response prints. The line and its number have been deleted. Example:
```
10 LET A=B+C
10
```

Replace Program Statement

1. Type the line number to be replaced, followed by the new program statement.
2. No system response prints. The last statement typed replaces the previous statement of the same number. Example:
```
10 LET A=B+C
20 ...
30 ...
10 LET X=B+C
```

Insert Program Statement

1. Number the statement to be inserted with a number whose value is between the line number preceding the insertion and the line number following the insertion.
2. Type the line number followed by the statement to be inserted. Example:
```
40 LET A=10.2
50 LET B=A+4.5
60 PRINT B
```

LANGUAGE ELEMENTS

UNARY OPERATORS

+	The value of
–	The negative value of

ARITHMETIC OPERATORS

+	Addition
–	Subtraction
*	Multiplication
/	Division
↑ or **	Exponentiation

RELATIONAL OPERATORS

<	Less than
⩽	Less than or equal to (2741 only)
<=	Less than or equal to
>	Greater than
⩾	Greater than or equal to (2741 only)
>=	Greater than or equal to
=	Equal to
≠	Not equal to (2741 only)
<>	Not equal to

INTRINSIC FUNCTIONS

NAME	DESCRIPTION
SIN(x)	Sine of x radians
COS(x)	Cosine of x radians
TAN(x)	Tangent of x radians
COT(x)	Cotangent of x radians
SEC(x)	Secant of x radians
CSC(x)	Cosecant of x radians
ASN(x)	Angle (in radians) whose sine is x
ACS(x)	Angle (in radians) whose cosine is x
ATN(x)	Angle (in radians) whose tangent is x
HSN(x)	Hyperbolic sine of x radians
HCS(x)	Hyperbolic cosine of x radians
HTN(x)	Hyperbolic tangent of x radians
DEG(x)	Convert x from radians to degrees
RAD(x)	Convert x from degrees to radians
EXP(x)	Natural exponentiation of x
ABS(x)	Absolute value of x
LOG(x)	Logarithm of x to the base e
LTW(x)	Logarithm of x to the base 2
LGT(x)	Logarithm of x to the base 10
SQR(x)	Positive square root of x
RND(x)	A random number between 0 and 1
INT(x)	Integral part of x
SGN(x)	Sign of x defined as: If x < 0, SGN(x) = -1 If x = 0, SGN(x) = 0 If x > 0, SGN(x) = +1

1. Uppercase letters, quotation marks, semicolons, colons, commas, and apostrophes represent information that must appear exactly as shown.
2. Lowercase letters in italics represent information that is supplied by the user.
3. A series of three periods indicates that a variable number of items may be included in a list.
4. The appearance of one or more items in sequence indicates that the items, or their replacements, should appear in the specified order.
5. Items shown in { } represent alternatives to be selected.
6. Items shown in [ ] represent options that may be omitted.

BASIC CONSTANTS

Internal Constants

Name	Short-Form Value (BASIC)	Long-Form Value (BASICL)
&E	2.718282	2.718281828459045
&PI	3.141593	3.141592653589793
&SQR2	1.414214	1.414213562373095

Literal Constants

Two general forms of the literal constant are:

'[c. . .]'
"[c. . .]"

where c is any character

Numeric Constants

Examples are:

100, 22.45, .3007

VARIABLE NAMES

Alphameric variables are represented by a letter followed by a dollar sign ($). Variables thus defined may contain up to 18 characters. For example:

```
LET A$ = 'PLANS'
IF B$ = 'YES' GOTO 270
```

Numeric variables are represented by a letter or a letter followed by a digit. For example:

A, B1, @, #4, S9

[LET] *variable, . . . , variable = expression*

```
10 LET A, B = 56.4
20 LET A = B+C/D-E
```

DEF FN *letter (variable) = expression*

```
20 DEF FNA(S) S^2+4*S
```

INTERNAL SPECIFICATION STATEMENTS

Enter Constants

READ *variable, variable, . . . , variable*

DATA *constant, constant, . . . , constant*

RESTORE

```
10 READ A, B, C, D$
20 DATA 100,200,300,'XYZ'
30 RESTORE
```

DATA FILE I/O STATEMENTS

Attach a Data File

OPEN *unit, filename, mode*

```
100 OPEN 1, 'OLDMAS', INPUT
200 OPEN 2, A$, OUTPUT
```

Read Data Values from a Data File

GET *[unit:]variable, . . . , variable*

```
100 GET A, B, C$
200 GET 1: X, Y, A$
```

If unit is omitted, 1 is assumed.

Store Data Values in a Data File

PUT *[unit:] field, . . . , field*

```
100 PUT A, B, C$
200 PUT 2: X, Y, A$
```

If unit is omitted, 2 is assumed.

NOTE: The term "field" represents an expression, an alphameric variable, or a literal constant.

Close a Data File

CLOSE *unit, unit, . . . , unit*

```
900 CLOSE 1, 2
```

Reset a Data File

RESET *unit, unit, . . . , unit*

```
900 RESET 1, 2
```

INPUT *variable, variable, . . . , variable*

```
20 INPUT A, B, C, D$
```

NOTE: The term "field" in the PRINT statement format represents an expression, an alphameric variable, a literal constant, or null.

Print Fields Using Full 18-Character Print Zones (uniform spacing, left justification)

PRINT *field, field, . . . , field*

```
50 PRINT A, B, C, D$
60 PRINT "VALUE OF X IS",X
70 PRINT D, E, SQR(F)+3,
```

Commas are used to print one field left-justified in a print zone allowing for a sign, then spacing to the next zone. A blank at the end of the list of fields spaces the carrier to the next line.

Print Fields Using Packed Zones

Print Field	Zone Size
Numeric Field	
2-4 characters	6 positions
5-7 characters	9 positions
8-10 characters	12 positions
11-13 characters	15 positions
14-16 characters	18 positions
Alphameric Field	
Alphameric variable	18 - number of trailing blanks
Literal constant	Size of the converted field

NOTE: A numeric print field begins with + or −, or + is assumed. The first position in the packed zone is reserved for this sign.

PRINT *field; field; . . . ; field*

```
80 PRINT A;B;C;D$
90 PRINT D; E; F;
```

Semicolons are used to print the field in a packed print zone and space to the next packed zone. A blank at the end of the list of fields spaces the carrier to the next line.

Print Literal

PRINT *literal-constant*

```
100 PRINT "THIS IS A LITERAL"
```

Print Fields Using Image Statement (spacing specified, punctuation inserted, and right justification specified)

PRINT USING *line number, variable, . . . , variable*

where *line number* refers to an image statement of the form:

: { *literal or* ******** } . . . { *literal or* ******** }

```
30 PRINT USING 40, A, B
40 :VALUE OF A IS###, VALUE OF B IS###.
```

Insert a Blank Line Between Printed Lines

PRINT

One-Dimensional Array

DIM *array name (integer), . . . , array name (integer)*

```
40 DIM D(20),E(20)
```

Two-Dimensional Array

DIM *array name (integer, integer), . . . , array name (integer, integer)*

```
50 DIM A(12,16),B(20,30)
```

Implicit Declaration

No DIM statement in program; variable referenced in program statement.

One-dimensional array assumes dimension of 10:

```
60 X=A(5)+N+Y-2.9
```

Two-dimensional array assumes dimensions of 10,10:

```
70 X=B(5,5) + A(7)
```

MATRIX STATEMENTS

Matrices may be operated on with the MATRIX statements shown below:

Matrix Addition: MAT $m_1 = m_2 + m_3$

Matrix Constant Function: MAT m = CON[(d_1,d_2)]

Matrix Get: MAT GET *[unit:]* $m_1[(d_{11},d_{12})],m_2[(d_{21},d_{22})],\ldots,m_n[(d_{n1},d_{n2})]$

Matrix Identity Function: MAT m = IDN[(d_1,d_2)]

Matrix Inversion: MAT m_1 = INV(m_2)

Matrix Multiplication: MAT $m_1 = m_2 * m_3$

Matrix Multiplication (Scalar): MAT m_1 = (x) * m_2

Matrix Print: MAT PRINT $m_1t_1m_2t_2m_3t_3\ldots m_n[t_n]$

Matrix Put: MAT PUT *[unit:]* $m_1,m_2,m_3,\ldots,m_n$

Matrix Read: MAT READ $m_1[(d_{11},d_{12})],m_2[(d_{21},d_{22})]\ldots,m_n[(d_{n1},d_{n2})]$

Matrix Subtraction: MAT $m_1 = m_2 - m_3$

Matrix Transposition: MAT m_1 = TRN(m_2)

Matrix Zero Function: MAT m = ZER[(d_1,d_2)]

LOOP STATEMENTS

Defining a Loop
(omit STEP and expression if increment = 1)

FOR *variable = expression* TO *expression* [STEP *expression*]

NEXT *variable*

```
120 FOR N=1 TO 10 STEP 2
    ...
    ...
150 NEXT N
```

BRANCH STATEMENTS

Unconditional Branch — Simple GOTO

GOTO *line number*

```
100 GOTO 10
```

Conditional Branch — Computed GOTO

GOTO *line number, line number, . . . , line number* ON *expression*

```
120 GOTO 10,40,60,80 ON X
```

Conditional Branch — IF Statement

IF *expression relational-operator expression* {GOTO / THEN} *line number*

```
130 IF X+Y=Z THEN 40
140 IF A<=B GOTO 50
```

Branch to Subroutine

GOSUB *line number*

```
40 GOSUB 80
```

Return from Subroutine

RETURN

```
40 GOSUB 80
50 ...
60 ...
70 ...
80 ...
90 ...
100 RETURN
```

REMARKS STATEMENT

Insert Comments

REM *characters*

```
REM THIS PROGRAM CALCULATES INTEREST
```

PROGRAM PAUSE AND TERMINATION STATEMENTS

Stop Program Execution
(resume by pressing the RETURN key)

PAUSE

Terminate Program Execution

STOP

Terminate Compilation and Program Execution

END