

BASIC- Beginners All-Purpose Symbolic Instruction Code

Commands, Statements, and Symbols NOTE: 't' often =up arrow

Command/ Statement	Example	Purpose
CLOSE	10 CLOSE n	Closes logical file'n'.
CLR	CLR	Sets variables to zero or null.
CMD	↓	Keep ieee device 'D' open to monitor bus.
CONT	CONT	Continue program execution after a stop command. No program changes are permitted.
DATA	10 DATA 1, 2,3,4 20 DATA TOM, SUE 30 DATA " DOE, TOM"	Specifies data to be read left to right. Alphabets do not need to be enclosed in quotes. if strings contain spaces, commas, colons, or graphic characters, the string must be enclosed in quotes.
DEF	10 DEF FN R(X)	Defines function 'R'
DIM	10 DIM A(n) 20 DIM A(n,m,o,p) 30 DIM A(n),B(m) 40 DIM A(N) 50 DIM A\$(n)	Specifies maximum number of elements in an array or matrix. Specifies maximum number of dimensions in an array. Number of arrays limited by memory. May be dimensioned dynamically. Strings to be dimensioned.
END	999 END	Terminates program execution.
FOR	10 FOR I = 1 TO10	Begins repetitive loop, specifying loop variable and number of intended iterations (in this example 'I' for 10 iterations).
FRE	PRINT FRE (0)	Returns number of bytes of available memory.
GET	10GETC 20 GET C\$ 30 GET #d, C 40 GET #d, C\$	Accepts single character from keyboard. Accepts single string character from keyboard. Accepts single character from specified logical file. Accepts specified single string character from logical file.
GOSUB	10 GOSUB n	Begins execution of a subroutine which begins on line 'n'.
GOTO	GOTO n	Continue program execution at line n after a stop command. Program changes are permitted.
IF... GOTO	10 IF X=10 GOTO n	Transfers execution to line 'n' if result of condition is true.
IF... THEN	10 IF X=10 THEN Y=3	Code following THEN is executed only if result of condition is true. May also be followed by line number to transfer execution.
INPUT	10 INPUT A 20 INPUT A\$ 30 INPUT A,A\$,B,B\$ 40 INPUT #d, A 50 input #d, a\$ 60 INPUT #d, A,A\$,B,B\$	Accepts value of 'A' from keyboard. Accepts value of string variable 'A' from keyboard. The string does not have to be enclosed in quotes. Accepts specified values from keyboard. Accepts value of 'A' from logical file 'd'. accepts specified string from logical file 'd'. Accepts specified values and string from logical file 'd'. Strings do not have to be enclosed in quotes.
LET	LET X=10	Optional. Assigns variable 'X' the value of 10.
LIST	LIST LIST -n LIST n-m LIST n-	Lists current program. Lists current program through line 'n'. Lists lines 'n' through 'm' of current program. Lists current program from line W to end.
LOAD	10 LOAD 20 LOAD "NAME" 30 LOAD "NAME", d 30 LOAD "NAME", d, c	Loads next encountered program from tape unit into memory. Loads program or file 'NAME' into memory from tape unit. Loads specified file 'NAME' from device 'd'. Loads specified file 'NAME' from device 'd' for command 'c'. (VIC/ C64 only - c = 1 for direct memory load)

Command/ Statement	Example	Purpose
NEW	NEW	Deletes current program in memory, sets variables to zero.
NEXT	NEXT	Indicates end of code contained in a FOR/NEXT loop.
ON ...GOSUB	10 ON A GOSUB I, m, n	Begins execution of subroutine which begins on specified line (in this example, T, 'm', or 'n') depending on value of index 'A'.
ON ...GOTO	10 ON A GOTO I, m, n	Transfers control to specified line 'I', 'm', or 'n' depending on value of index 'A'.
OPEN	10 OPEN a 20 OPEN a, d 30 OPEN a, d, c 40 OPEN a, d, c, ' NAME"	Opens logical file 'a' for read only from tape unit. Opens logical file 'a' for read only from device 'd'. Opens logical file 'a' for command 'c' from device 'd'. Opens logical file 'a' on device 'd'. If device 'd' accepts formatted files, file name is positioned for command.
PEEK	PEEK(a) PEEK(A)	Returns byte value from address 'a'. Address can be dynamic.
POKE	POKE a, b POKE A, B	Puts byte 'b' into address 'a'. Parameters can be dynamic.
POS	10 PRINT POS(0)	Prints next available print position (position of cursor on screen).
PRINT	10 PRINT A 20 PRINT A\$ 30 PRINT A, A\$ 40 PRINT #d, A 50 PRINT #d, A\$	Prints value 'A' on display screen. Prints specified string on screen. Prints specified values or strings on screen, beginning in next available print position (pre-tabbed positions are in columns 10,20,30,40 etc.). Prints value of 'A' on device 'd'. Prints specified string on device 'd'.
READ	10 READ A\$, B\$	Reads next two data elements into variables A\$ and B\$.
REM	10 REM comment	Remark indicator. Execution skips entire line.
RESTORE	10 RESTORE	Resets data pointer so that next READ receives first element of first DATA statement.
RETURN	9990 RETURN	Subroutine exit; transfers control to the statement following most recent gosub directing transfer to the subroutine.
RUN	RUN RUN n	Begins execution of program at lowest line number. Begins execution of program a line 'n'.
SAVE	SAVE "NAME" SAVE "NAME", d SAVE " NAME" , d, c	Saves current file or program 'NAME' on tape unit Saves current program or file 'NAME' on device 'd'. Saves file 'NAME' on device 'd'. 'c' specifies eof or eot.
STEP	10 FOR I =1 TO 10 STEP 2	Alters loop variable increment.
STOP	STOP	Stops program execution.
SYS	SYS (x)	Complete control of pet is transferred to a subsystem at decimal address contained in the argument. Brackets optional.
USR	USR (x)	Transfers program control to a program whose address is at locations 1 and 2 (VIC/C64 - locations 784,785). 'x' is a parameter passed to and from the machine language program.
VERIFY	VERIFY VERIFY "NAME" VERIFY " NAME", d	Verifies current program against next program on tape unit. Verifies current program against program 'NAME' on tape unit. Verifies current program 'NAME' on device 'd'.
WAIT	WAIT a, b, c	Halts execution of Basic until contents of address 'a', and 'ed with value 'b' and exclusive or 'ed with value 'c'. is not equal to zero. 'c' is optional and defaults to zero.

Arithmetic Operators

Symbol	Example	Purpose
=	10 A = B	Assigns a value to a variable.
t	30 PR INTA ^t 2	Exponentiation
/	40 C=A/8	Division.
*	50 C=A*8	Multiplication.
+	60 C = A + 8	Addition.
-	70 C=A-8	Subtraction.
=	10 IF A=B THEN PRINTC	'A' Equals 'B'.
<>	10 IF A<>B THEN C =4	'A' Does not equal 'B'.
<	10 IF A<B THEN C\$ X"	'A' Is less than 'B'.
>	10 IF A>B THEN C\$= "Y"	'A' Is greater than 'B'.
<=	10 IF A<=B THEN C=20	'A' Is less than or equal to 'B'.
>=	10 IF A>=B THEN C=D-1	'A' Is greater than or equal to 'B'.
AND	10 IF A AND B THEN C =9	'A' and 'B' must both be true for statement 10 to be true.
OR	20 IF A OR B THEN C =9	'A' must be true or 'B' must be true for statement 20 to be true.
NOT	30 IF NOT A THEN PRINT C	Expression is true if 'A' is false.

Note: the numerical values used in the evaluation of logical comparisons are: 'true' is any non-zero number and 'false' is zero.

Special Symbols

Symbols	Example	Purpose
	10 A =1: B= 2:C =3	Allows multiple statements on a line.
	10 PRINTA;B 20 PRINT A\$; B\$	Allows same line printing. Elements are separated by 3 spaces. Allows same line printing. String elements are concatenated.
.	X =10.99	Decimal point
	10 PRINT A, B LOAD "NAME ",d	Allows same line printing. Elements are separated and printed in pre-tab'ed print positions (columns 10,20,30, etc.). Separates parameters in load, save, open, mid\$, on..goto, etc.
?	10 ?A	Abbreviation for 'print'. Stores as one character; lists as word PRINT
\$	10 A\$ "ABCDEFGG "	String identifier.
	10 A% = INT(X)	Integer identifier.
#	10 PRINT#8	Logical file number indicator
'	10 A\$ "ABCDEFGG "	String enclosures.
()	X = (A-2)/(B + 2)	Expression priority evaluation
π	10 C =pie_symbol^D	Value of Pi 3.1415927.

Reserved Variables

Variable	Purpose
DS	Disk Status number (except 2.0)
DS\$	Disk Status string (except 2.0)
EL	Error Line (B Series only)
ER	Error number (B only)
ERR\$(Error String array. See table for messages. (B only)
TI	Time in Jiffies (1/60th's sec.) since power up or TI\$ reset (except B Series)
TI\$	Time in HHMMSS
ST	The Status variable. See table for functions.

Hierarchy of Operations

Operator	Description
()	Brackets always dictate priority
^	Exponentiation
-	Negation (unary minus)
! /	Multiplication & Division
+ -	Addition & Subtraction
< _ >	Relational Operations
NOT	Logical NOT (Integer two's complement)
AND	Logical AND
OR	Logical OR

Basic 4.0 Disk Commands

Function	Example	Purpose
APPEND	10 APPEND#d, "NAME"	Open file 'NAME' on device 'd' for appending. New data is added to end of existing data.
BACKUP	BACKUP DO TO D1	Duplicate disk in drive 0 onto disk in drive 1
CATALOG	CATALOG DO	Displays list of filenames in specified drive.
COLLECT	COLLECT D1	Purges disk in specified drive of any improperly closed files (indicated by * beside file type).
CONCAT	CONCAT "NAME1 " TO "NAME2", D1	Concatenates file "NAME1 " to file "NAME2 ". I.e. NAME2 NAME2+NAME1
COPY	COPY NAME ",DO TO NAME ",D1 COPY NAME ",DO TO DUP ",DO COPY DO TO D1	Copies file " NAME " from drive 0 to drive 1 Makes duplicate of file "NAME" Copies entire contents from DO to D1
DCLOSE	DCLOSE#n	Closes disk logical file'n'
DIRECTORY	DIRECTORY DO	Exact same as Catalog. Use preference.
DLOAD	DLOAD "NAME" ,Dd,Uu	Loads program "NAME" from drive `d' on unit 'u'
DOPEN	DOPEN#n, "NAME ",Dd,Uu DOPEN#n, "NAME" ,Dd,Uu,W	Opens file "NAME" for reading from drive V, unit V. Default values: d=0, u=8. Data is retrieved through file number 'n'. Opens file " NAME" for writing to drive V, unit 'u'. Not necessary for RELative files.
DSAVE	DSAVE "NAME" ,Dd,Uu	Saves current program to drive 'd' on unit 'u' as file ' NAME
HEADER	HEADER"DISKNAME ",Dd,lid,Uu	Formats disk in drive V unit 'u' assigning it a ' DISKNAME and 'id'.
RECORD	0 RECORD#n, a	Positions relative file open on logical file number 'n' to record number 'a', 'a' may be dynamic but must be enclosed in brackets.
RENAME	RENAME "NAME" TO NEWNAME",DO	Changes a file name.
SCRATCH	SCRATCH "NAME ",D1	Eliminates file " NAME " from disk.

Additional B Series Commands

Function	Example	Purpose
BANK	BANK b	Sets bank number to 'b'.
BLOAD	BLOAD "NAME" ,Dd,Uu,ONBb,Pp	Loads file " NAME" from drive 'd' unit 'u' into bank 'b' at position 'p'
BSAVE	BSAVE "NAME" ON Bb,Pp1 to Pp2	Saves current memory in bank 'b' from address 'p1' to 'p2' as file NAME " to drive 0 unit 8. Addresses are in decimal.
DCLEAR	DCLEAR D1	Initialize disk in drive 1
DELETE	DELETE 10-30	Deletes lines from current program. Specify line range same as LIST.
DISPOSE	DISPOSE GOSUB	Purges stack of unwanted return addresses (like 'POP')
ELSE	IF ST THEN E= 1 ELSE E =0	Alternate condition following IF..THEN. May also be used to transfer execution
INSTR	PRINT INSTR (A\$, B\$)	Returns position of string B\$ within A\$. Returns 0 if not found.
KEY	KEY KEYn, " CATALOG DO +CHR\$(13)	Displays list of function key definitions Defines function key 'n'.
PUDEF	PUDEF " -X	Re-defines Print Using format characters. Default is In this example, space is changed to '-', comma to period, period to comma, and dollars to pounds.
RESUME	RESUME RESUME n RESUME NEXT	Continues execution after program error or editing Resumes execution at linen' Resumes *execution at start of current active FOR/NEXT
TRAP	TRAP 50000	Specifies routine at line 50000 as an ON ERROR routine.
USING	PRINT USING "-\$##,###";X	Specifies format to be used for numerical output.

String Functions

Function	Example	Purpose
ASC	10 A=ASC("XYZ")	Returns the integer value corresponding to ASCII code of the first character in string.
CHR\$	10 A\$= CHR\$(n)	Returns character corresponding to ASCII code number.
LEFT\$	10 PRINT LEFT\$(X\$, a)	Returns leftmost 'a' characters from string.
LEN	10 PRINT LEN(X\$)	Returns length of string.
MID\$	10 PRINT MID\$(X\$, a, b)	Returns 'b' characters from string, starting with the 'a'th character.
RIGHT\$	10 PRINT RIGHT\$(X\$, a)	Returns rightmost'a' characters from string.
STR\$	10 A\$=STR\$(A)	Returns string representation of variable 'A'
VAL	10 A=VAL(A\$) 20 A=VAL("A")	Returns numeric representation of string. If string not numeric, returns "0".

ASC, LEN and VAL functions return numeric results. They must be used as part of any expression. Assignment statements are used here for examples only; other statement types may be used.

Arithmetic Functions

Function	Example	Purpose
ABS	10 C = ABS(A)	Returns magnitude of argument without regard to sign.
ATN	10 C = ATN(A)	Returns arctangent of argument. 'c' will be expressed in radians.
COS	10 C = COS(A)	Returns cosine of argument. 'A' must be expressed in radians.
DEF FN	10 DEF FNA(B) =C *D	Allows user to define a function. Function label 'a' must be a single letter; argument 'b' is a dummy.
EXP	10 C = EXP(A)	Returns constant 'e' raised to the power of the argument.
INT	10 C = INT(A)	Returns largest integer less than or equal to argument.
LOG	10 C = LOG(A)	Returns natural logarithm of argument. Argument must be greater than or equal to zero.
RND	10 C = RND(A)	Generates a random number between zero and one. If 'a' is less than 0, the same random number is produced in each call to rnd. If 'a' = 0, the same sequence of random number is generated each time rnd is called. If 'a' is greater than 0, anew sequence is produced for each call to rnd
SGN	10 C = SGN(A)	Returns -1 if argument is negative, returns 0 if argument is zero, and returns + 1 if argument is positive.
SIN	10 C = SIN(A)	Returns sin of argument. 'A' must be expressed in radians.
SQR	10 C = SQR(A)	Returns the square root of argument.
TAN	10 C = TAN(A)	Returns tangent of argument. 'A' must be expressed in radians

Mathematical Functions

Function	Basic Equivalent
Secant Cosecant Cotangent	SEC(X) = 1 / COS(X) CSC(X) = 1 / SIN(X) COT(X) = 1 / TAN(X)
Inverse Sine Inverse Cosine Inverse Secant Inverse Cosecant Inverse Cotangent	ARCSIN(X) = ATN(X / SQR(-X*X + 1)) ARCCOS(X) = ATN(X / SQR(-X*X + 1)) + pi_symbol/2 ARCSEC(X) = ATN(X / SQR(X*X-1)) ARCCSC(X) = ATN(X / SQR(X*X-1)) + (SGN(X) * -1 * pi_symbol/2) ARCCOT(X) = ATN(X) - pi_symbol/2
Hyperbolic Sine Hyperbolic Cosine Hyperbolic Tangent Hyperbolic Secant Hyperbolic Cosecant Hyperbolic Cotangent	SINH(X) = (EXP(X) - EXP(-X)) / 2 COSH(X) = (EXP(X) + EXP(-X)) / 2 TANH(X) = EXP(-X) / (EXP(X) + EXP(-X)) * 2 + 1 SECH(X) = 2 / (EXP(X) + EXP(-X)) CSCH(X) = 2 / (EXP(X) - EXP(-X)) COTH(X) = EXP(-X) / (EXP(X) - EXP(-X)) * 2 + 1
Inverse Hyperbolic Sine Inverse Hyperbolic Cosine Inverse Hyperbolic Tangent Inverse Hyperbolic Secant Inverse Hyperbolic Cosecant Inverse Hyperbolic Cotangent	ARCSINH(X) = LOG(X + SQR(X*X + 1)) ARCCOSH(X) = LOG(X / SQR(X*X-1)) ARCTANH(X) = LOG((1 + X) / (1-X)) / 2 ARCSECH(X) = LOG(SQR(-X*X + 1) + 1 / X) ARCCSCH(X) = LOG(X / SQR(X*X-1)) + (SGN(X) * -1 * pi_symbol/2) ARCCOTH(X) = LOG(X) - pi_symbol/2