

BASIC Programming Language

BASIC stands for Beginners All-purposed Symbolic Instruction Code. It was developed in 1964 by Professor John Kemeny and Thomas Kurtz to teach students at Dartmouth College. It has undergone series of historical development, which has resulted in several forms of the language.

Versions of BASIC

1. BASICA
2. GWBASIC
3. Turbo BASIC
4. Quick BASIC
5. VB.NET

BASIC CHARACTER SET

1. Alphabetic Characters – A to Z
2. Numeric Character – 0 to 9
3. Special Characters – 0 + % ^ # = () etc

BASIC VARIABLES

A variable is a quantity that changes during the execution of a program. It can also be defined as a name that is used to represent some storage location.

Types of Variable

1. Numeric Variables: These are used to store numeric values such as 23, 98, etc. There are two major types of numeric variables, which are; Integer variable and real variable
 - a. Integers are whole numbers without decimal places.
 - b. Real numbers are numbers with decimal places

Hence Integer variable are names which the computer can assign integer values or constants to and real variable are name which the computer can assign real numbers or constant to. Integer variable name are written with a “%” sign as the last character. E.g. . When a numeric variable name is written without the % sign, it can take both integer and real numbers.

2. String Variables: These are used to store alphabetic and alpha-numeric values. A string variable name is always written with a dollar sign (\$) as the last character. E.g. Name\$=“John”

Rules for coding variable

1. In BASIC combining alphabets, numbers and decimal point (a maximum length of 40 characters) may for variable.
2. No reserve word can be used as a variable name.
3. Special characters cannot be used for naming variable.
4. A string variable corresponds to string data whereas a numeric variable corresponds to numeric data,
5. In a program, each variable is referred throughout the program by its name.

CONSTANTS

A constant is data that remains the same as the program runs (executes).

BASIC allows two types of constants:

1. Numeric constant: Numeric constant in BASIC is any signed or unsigned number.
2. Alpha-Numeric or string constant: It consists of the combination of letters, digits, and other symbols that is treated in a manner completely analogous to numeric constant. They are enclosed within inverted comma

Rules for numeric constants

1. A number can have maximum of 8 digits
2. No comma is allowed
3. A decimal point can appear anywhere
4. If the value is quite larger it is expressed in exponent form
5. No blank space, special characters or any other letter is allowed in the number.

Expressions and Operators

In programming, an expression can be defined as the combination of operand and operator which is to be evaluated to produce answer. Operands are the data items involved in an expression. Operators determine the action to be carried out on the operand in the expression. For instance in the statement: $LET C = A + B$, A and B are the operands while "+" is the operator.

There are three major types of expression in BASIC. They are:

- i. Arithmetic expression
- ii. Relational Expression
- iii. Logical expression

Arithmetic Expression and Arithmetical Operator

BASIC arithmetic expression is used to represent mathematical formulae in BASIC programming. Below is a list of BASIC arithmetic operators:

Arithmetic Operator

SYMBOL	NAME	FUNCTION
\wedge	Upper caret	Exponential
/	Slash	Division
*	Asterisk	Multiplication
+	Plus	Addition
-	Minus	Subtraction

Arithmetic Expression

Mathematical formulae	BASIC Expression
lbc	$L*B*C$
$ut+1/2at^2$	$U*T+1/2*A*T^2$
$2(lb + bh + lh)$	$2*(L*b+B*H+L*H)$
$PTR/100$	$P*T*R/100$

Relational Expression

Relational Expression is used for comparison of two or more data items. BASIC relational operators are listed below:

SYMBOL	NAME
<	Less than
>	Greater than
=	Equal to
<>	Not equal to
<=	Less than or equal to
>=	Greater than or equal to

Logical Expression

Logical expression involve is an expression involving two or more relational repression joined by logical expression. BASIC logical operators are:

- i. AND
- ii. NOT
- iii. OR

EVALUATION OF ARITHMETIC EXPRESSION

To evaluate arithmetic expression, the following order is followed:

PRIORITY	OPERATOR
1 ST	Parenthesis I.e. (and)
2 nd	Exponentiation
3 rd	Multiplication and Division
4 th	Mod and Inter Division
5 th	Addition and Subtraction

Example: evaluate $4*A*B^2 + (A^2*B+C)/(A+B)$ if $A=2$; $B=4$ and $c=2$

Solution

Step 1 Substituting we have $4*2*4^2 + (2^2*4+2)/(2+4)$

Step 2 evaluate terms in the parenthesis $4*2*4^2+18/6$

Step 3 evaluate 4^2 $4*2*16+18/6$

Step 4 evaluate $4*2*18$ $128+18/6$

Step 5 evaluate $18/6$ $128 + 3$

Step 6 evaluate 131

BASIC STATEMENT

1. LET Statement

The let statement is used to assign a numeric or string value to a variable.

Syntax

LET [variable] = [constant] for numeric value

LET [variable]\$ = ["value"] for string value

Example

LET X = 12

LET B\$ = "Clementina"

LET AREA = L*B

2. INPUT Statement

The INPUT statement is used to enter data into the computer with a user prompt or a group of variable during program execution.

Syntax for numeric value

INPUT "[prompt]"; [variable]

Syntax for string value

INPUT "[prompt]"; [variable\$]

Example

INPUT "type in the number"; A

INPUT "Type in your name"; N\$

3. READ-DATA statement

READ and DATA are two statements concerned with each other which are used to put data in a line of the program and to read the data when it is needed.

Example

```
READ A, B, C
```

```
DATA 5, 6, 7
```

```
LET SUM = A+B+C
```

```
PRINT SUM
```

```
END
```

4. REM (Remark) Statement

The REM statement is used to insert comments or remarks into a BASIC program. The use of remark statements improves the readability of the program. REM is a non-executable statement.

Syntax

```
REM [remark]
```

Example

```
REM program to add six numbers
```

5. PRINT statement

This statement is used to transmit data from the computer memory to the output device.

Examples

```
PRINT A
```

```
PRINT "I Like Writing Program"
```

Program Terminators (END and STOP)

6. STOP and END statement

The STOP statement is used to terminate the execution of a program at any point in the program. The END statement indicates the actual end of a program. The STOP statement may appear many times and anywhere, whereas an END statement can only appear at the end of a program and only once.

Example

```
REM END statement
```

```
PRINT "Good morning"
```

```
END
```

7. FOR – NEXT

Looping is used to have the computer do repetitive tasks in a fraction of the time that would be otherwise be required. The most common type of loop used in QBASIC programming is the FOR...NEXT and WHILE WEND loop that repeats a series of instructions a specified number of times.

Syntax

```
FOR variable=x TO y [STEP z]
```

```
.
```

```
.
```

```
.
```

```
NEXT [variable][,variable...]
```

x,y, and z are numeric expressions.

STEP z specifies the counter increment for each loop.

Example 1

```
FOR I = 1 TO 5
```

```
PRINT "the dulllest pencil is better than the sharpest memory"
```

```
NEXT I
```

```
END
```

EXAMPLE 2

REM program to print odd numbers from 1 to 20

```
PRINT "odd numbers from 1 to 20 are"
```

```
FOR ODD =1 TO 20 STEP 2
```

```
PRINT ODD
```

```
NEXT ODD
```

```
END
```

SIMPLE BASIC PROGRAMS

Example 1: Program to find the sum and difference two number

```
10 REM this program accepts two numbers and finds their sum and difference
```

```
20 INPUT "Type the first number and press ENTER"; NUM1
```

```
30 INPUT "Type the second number and press ENTER"; NUM2
```

```
40 LET SUM = NUM1 + NUM2
```

```
50 LET DIFF = NUM1 - NUM2
```

```
60 PRINT "first number is "; NUM1
```

```
70 PRINT "second number is "; NUM2
```

```
80 PRINT "=====
```

```
90 PRINT NUM1; "+"; NUM1 "=" ; SUM
```

```
100 PRINT NUM1; "-"; NUM2 "=" DIFF
```

```
110 END
```

Example 2: program to calculate the area and perimeter of a rectangle

```
10 REM program to find the area and perimeter of a rectangle
```

```
20 INPUT "Type the length of the rectangle"; L
```

```
30 INPUT "Type the in the breadth of the rectangle"; B
```

```
40 LET AREA = L*B
```

```
50 LET PERI = 2 * (L + B)
```

```
60 PRINT "The area of the rectangle is " ; AREA
```

```
70 PRINT "The perimeter of the rectangle is" ; PERI
```

```
80 END
```