Introduction to C Language

Variables:

- ➲ Variables are the identifiers, the named memory location whose value changes on program execution.

Declaration of variables:

- ➲ The declaration tells the computer which storage locations or variables to use in the program.
- ➲ The variables which will be used in the program should be declared at the beginning of the program.
- ➲ The variable declaration indicates the type of constant that the variable can hold and the amount of memory allocated to the variable.
- ➲ The name of the variables should be declared as per the rules of declaring identifiers or variables.
- ➲ **General Syntax:** datatype variable;

  (or)

  datatype variable1, variable2,……..variable_n;

  Example: 1. int a; 2. float x,y; 3. double sum, midun_06;
- ➲ From the examples we will come to know that
  - ✓ "a" is a variable of type integer and allocates 2 bytes of memory.
  - ✓ "x" and "y" are two variable of type float which will be allocated 4 bytes of memory for each variable.
  - ✓ "sum" and "midun_06" are two double type variables which will be allocated with 8 bytes of memory for each.

Variable Initialization:

- ➲ Variables can be initialized at two instances -
  - ✓ Compile Time (using Assignment Statement)
  - ✓ Run Time (using Standard Input function : scanf( ))

Compile Time Initialization: Assignment statement -

- ➲ Declaration indicates only creating space for variables but doesn't assigns any value.
- ➲ **General Syntax:** Var_name = expr;

  Where,

  Var_name is the name of the variable ,

  expr is the value of the expression or the constant value.
- ➲ The "expr" on the right hand side is evaluated and stored in the variable name (Var_name) on left hand side.

- The expression on the right hand side may be a constant, variable or a larger formula built from simple expressions by arithmetic operators.

  Examples:

  1. int a=10; // assigns the value 10 to the integer variable a
  2. float x;

     x=20; // creates a variable y of float type and assigns value 20 to it.
  3. int a=10,b;b=a; // creates two variables a and b. "a" is assigned with value 10, the value of "a" is assigned to variable "b". Now the value of b will be 10.

Other Assignment Statements:

- price = cost*3; //assigns the product of cost and 3 to price.
- Square = num*num; // assigns the product of num with num to square.

Run Time Initialization :scanf( ) statement -

- To enter the input through the standard input devices like keyboard we make use of scanf statement.
- In this initialization of the variables is done at run time.
- **General Syntax:**scanf("format string",list of address of variables);

  Where: Format string consists of the access specifiers.

  List of addresses of variables consist of the variable name preceded with & symbol.

  Example:

  ```
  int a;
  float b;
  scanf("%d%f",&a,&b);
  ```

Constants

Constants are the named memory location; it refers to fixed values that do not change during the execution of a program.

We have different types of constants

1. Integer constant
2. Real constant/Floating Pointing
3. Character constant Ex: ‘a’, ‘9’, ‘\n’
4. String constant Ex: “INDIA”, “8”

Integer constant:

- It refers to sequence of digits
- Embedded spaces, commas, characters should not be included.
- Must not contain a decimal point.
- May be signed or unsigned. (default is +)

Three types of integers

- **Decimal integers:** Consist of digits 0 to 9
  Ex: 123 -345 0 5436 +79
- **Octal integers:** Digits from 0 to 7 but it has to start with 0
  Ex: 027 0657 0777645
- **Hexadecimal integers:**Digits from 0 to 9 and characters from a to f, it has to start with 0X or 0x
  Ex: 0X2 0x56 0X5fd 0xbdae

Real constants/Floating point:

The numbers that are represented by fractional parts are called real constants.

Two forms

1. Fractional form:
   - Digits from 0 to 9, sign (+ or -), dot(.)
   - Ex: 0.0083 215. -71. +0.56 etc..
2. Exponential form:
   - Syntax: mantissa **e** exponent
   - Mantissa is either real number expressed in decimal notation or integer.
   - Exponent is integer with optional + or –
   - Ex: 0.65e4 3.18e-2 73e+3 12e+5

Character Constant:

- Character Constant Can hold Single character at a time.
- Contains Single Character Closed within a pair of Single Quote Marks
- Single Character is smallest Character Data Type in C.
- Integer Representation : Character Constant have Integer Value known as ‘ASCII’ value
- It is Possible to Perform Arithmetic Operations on Character Constants

  Examples: ‘a’ ,‘1’ , ‘#’ , ‘<‘ , ‘X’

String Constant

- A character string, a string constant consists of a sequence of characters enclosed in double quotes.
- A string constant may consist of any combination of digits, letters, escaped sequences and spaces.
- Note that a character constant‘A’ and the corresponding single character string constant "A" are not equivalent.
- The string constant "A" consists of character A and \0. However, a single character string constant does not have an equivalent integer value. It occupies two bytes, one for the ASCII code of A and another for the NULL character with a value 0, which is used to terminate all strings.
- Valid String Constants: -

  "W"
  "100"
  "24, Kaja Street"

- Invalid String Constants: -

  "W the closing double quotes missing

  Raja" the beginning double quotes missing

- Rules for Constructing String constants
  - A string constant may consist of any combination of digits, letters, escaped sequences and spaces enclosed in double quotes.
  - Every string constant ends up with a NULL character which is automatically assigned (before the closing double quotation mark) by the compiler.

➲ Difference between single character constant and string constant

Character Constant	String Constant
A character constant is enclosed within single inverted commas.	A sequence of characters enclosed in double quotes.
The maximum length of a character constant can be one character.	A string constant can be any length.
A single character string constant has an equivalent integer value.	A single character string constant does not have an equivalent integer value.
The character constant 'A' consists of only character A.	The string constant "A" consists of character A and \0.
A single character constant occupies one byte.	A single string constant occupies two bytes.
Every character constant does not end up with a NULL character	Every string constant ends up with a NULL character which is automatically assigned (before the closing double quotation mark) by the compiler.