C- Variables

- A **variable** is a name of the memory location
- Variable in C Programming is also called as container to store the data
- Its value can be changed, and it can be reused many times..
- Variable may have different data types to identify the type of value stored. Suppose we declare variable of type integer then it can store only integer values.
- A Variable is a name given to the memory location where the actual data is stored.
- Variable Name always **holds a single Value**.
- Variable Name is **user defined name given to Memory Address**.
- During Second Run , **Address** of Variable **may change**.
- Example:

```
int a;
float b;
char c;
```

Rules For Constructing Variable Name

- **Characters Allowed :**
  - Underscore(_)
  - Capital Letters (A – Z)
  - Small Letters (a – z)
  - Digits (0 – 9)

- **Blanks & Commas** are not allowed

- No Special Symbols other than **underscore(_) are allowed**

- **First Character** should be **alphabet or Underscore**

- Variable name Should not be **Reserved Word**

Types of Variables

There are many types of variables in c:

- local variable
- global variable
- static variable
- automatic variable
- external variable

Types of Variables

1. Local Variable

- Local Variable is Variable having Local Scope.
- Local Variable is accessible only from function or block in which it is declared .
- Local variable is given Higher Priority than the Global Variable.

Example

```
void function1()
{
int x=10; //local variable
}
```

- You must have to initialize the local variable before it is used

2. Global Variable :

- **Global Variable** is Variable that is Globally available.
- A variable that is declared outside the function or block is called a global variable
- Scope of Global variable is throughout the program [**i.e in all functions including main()**]
- Global variable is also visible inside function , provided that it should not be re-declared with same name inside function because … "**High Priority is given to Local Variable than Global**"
- Global variable can be accessed from any function. i.e Any function can change the value of the global variable
- It must be declared at the start of the block.

Example:

```
int value=20;  //global variable
void function1()
{
int x=10;  //local variable
}
```

3. Static Variable :

- A variable that is declared with the static keyword is called static variable.
- It retains its value between multiple function calls.

Example:

```
void function1()
{
        int x=10;//local variable
        static int y=10;//static variable
        x=x+1;
        y=y+1;
        printf("%d,%d",x,y);
}
```

If you call this function many times, the **local variable will print the same value** for each function call, e.g, 11,11,11 and so on. But the **static variable will print the incremented value** in each function call, e.g. 11, 12, 13 and so on.

4. Automatic Variable :

All variables in C that are declared inside the block, are automatic variables by default. We can explicitly declare an automatic variable using **auto keyword**.

Example:

```
void main()
{
        int x=10;  //local variable (also automatic)
        auto int y=20;  //automatic variable
}
```

5. External Variable

We can share a variable in multiple C source files by using an external variable. To declare an external variable, you need to use **extern keyword**.

```
        extern int x=10; //external variable (also global)
```

Identifiers

- C identifiers represent the name in the C program, for example, variables, functions, arrays, structures, unions, labels, etc.
- An identifier can be composed of letters such as uppercase, lowercase letters, underscore, digits, but the starting letter should be either an alphabet or an underscore.
- an identifier is a collection of alphanumeric characters that begins either with an alphabetical character or an underscore, which are used to represent various programming elements such as variables, functions, arrays, structures, unions, labels, etc.

An identifier is a collection of alphanumeric characters that begins either with an alphabetical character or an underscore, which are used to represent various programming elements such as variables, functions, arrays, structures, unions, labels, etc.

Rules for constructing C Identifiers

- The first character of an identifier should be either an alphabet or an underscore, and then it can be followed by any of the character, digit, or underscore.
- It should not begin with any numerical digit.
- In identifiers, both uppercase and lowercase letters are distinct. Therefore, we can say that identifiers are case sensitive.
- Commas or blank spaces cannot be specified within an identifier.
- Keywords cannot be represented as an identifier.
- The length of the identifiers should not be more than 31 characters.
- Identifiers should be written in such a way that it is meaningful, short, and easy to read.

Differences between Keyword and Identifier

Keyword	Identifier
Keyword is a pre-defined word.	The identifier is a user-defined word
It must be written in a lowercase letter.	It can be written in both lowercase and uppercase letters.
Its meaning is pre-defined in the c compiler.	Its meaning is not defined in the c compiler.
It is a combination of alphabetical characters.	It is a combination of alphanumeric characters.
It does not contain the underscore character.	It can contain the underscore character.