Variables in C

In this C tutorial you will learn all about variables, the basic unit of programming language. We will discuss in detail what is a variable in C, how to name, declare and initialize a variable with the help of examples .Beside these you will also walk through Lvalues and Rvalues in C and the types of variables in C.

What is a variable in C ?

Variable is a reserved space of computer memory where data can be stored and you can assign a specific name to each of these memory locations. In RAM every byte of data gets stored sequentially with a definite address. Its value starts with '0' and continues to increase as the sequence of memory allocation proceeds.

How to name variables in C ?

Though 'C' has provided the freedom of using variable names according to your wish and convenience, there are certain rules you must abide by.

1. Variable names can be a combination of alphabets(A,a,B,b), numbers(0,1,2) and underscore(_) only.
2. Variable names should start with alphabets and some compilers support underscore too. But it can't begin with a number.
3. Since C is a case-sensitive language both uppercase (NUM) and lowercase (num) letters are treated differently.
4. As per ANSI standards, it is safer to use variable names short, simple and meaningful.(not more than 31 characters).
5. Special characters(/,;..)and symbols(#,@) are not allowed while naming a variable.
6. The variable name must be different from C keywords(printf, scanf).
7. No whitespace is allowed.

VALID VARIABLES NAMES	INVALID VARIABLES NAMES
_age	1?age
num,NUM	int, print
Stud1, Stud_2	Stud#,Stud 1,Stud@2

How to declare a variable ?

In C programming all the variables which are going to be used in the program must be declared first. It is because of the two reasons

1. The compiler must know how much space to be reserved for the variable for proper functioning. The allocated space must be sufficient and not extra large causing misuse of memory.
2. The compiler must recognize the data whether it is a number or alphabet and that is why 'type' must be declared. Except this, a meaningful name (called identifier)should be given to it for ease of operation.

Variable declaration syntax involves two important parts as shown below, one is the **data type** and other is the **variable name**. Variable declarations should always end with **semicolon** unless the compiler will raise a termination error. Also C provides the freedom to declare **multiple variables** in a single line if they possess the same datatype.

```
datatype variable_name;
```

Basically, there are three data type declarations used very often. They can be illustrated by this example:

```
int num, no;
char a, b;
float f1,f2;
```

Here, '**int**' means whole numbers, '**char**' means a single character and '**float**' means a fractional number.int, char, float are data types and num, no, a, b, f1,f2 are identifiers(variable names). when a variable is declared with a specific data type then it can only take values of that particular data type. Here **num** can take only integral values , if we try to store any other values like character to num then it will raise an error.

How to initialize a variable ?

After the declaration is done, you are free to use the allocated space (i.e. variable) as per your requirement.This declared variable initially contains garbage values which are the undefined values. If you wish to assign some values initially you can do that with the help of assignment operator '='. Variable initialization can be performed at the time of declaration itself . The syntax is as follows.

```
datatype variable_name = value;
```

Some examples of variable initialization is illustrated below:

```
int num = 10, no =20;
char a = 'A', b = 'B';
float f1 = 2.5 ,f2 = 5.2;
```

What are Lvalues and Rvalues in C ?

Lvalues actually refer to the address location of the variable. It generally appears on the left side of the operator (=, >). So you can say an identifier is a modifiable lvalue which can vary.

```
#include <stdio.h>
main()
{
     int num1,num2,total;
     num1=4;
     num2=5;
     total=num1+num2;
     printf("Total number is %d",total);
}
```

In this program **num1,num2,total** act as lvalues in 5th, 6th and 7th line respectively.

Rvalues are generally assigned values i.e the data stored in the allocated address. It can be any character or number, generally found on the right side of the operator. In the above-mentioned program **4,5** are rvalues in the 5th and 6th line respectively.

So, at last, we can say that lvalue implies the identity of the object and rvalue denotes the value of it.

Types of variables in C

C being a static language contains 5 types of variables. They are listed below:

1. local variable
2. global variable
3. static variable
4. automatic variable
5. external variable

Local Variable

A local variable is a variable which is defined inside a function or block of code. The scope of the variable is limited to that function or code. Hence we can say this type of variables have local scope only.

```
void funct()
{
     int num = 5;   // Local Variable
}
```

Global Variable

A global variable is a variable which is defined outside a function or block of code. The scope of a global variable is valid throughout the program and it can be used by any functions or codeblocks. The value of the global variable can be modified inside a function.

```
int g = 10;   // Global Variable
void funct()
{
     int num = 5;   // Local Variable
}
```

Static Variable

A variable is said to be static if it is prefixed with the keyword static. Static variables are useful during function calls because a variable declared as static preserves the values on each function calls. The below code snippet will illustrate the working of static variable and local variable.

```
#include <stdio.h>
int main()
{
void funct()
{
      static int a =10; // Static Variable
      int num = 5;     // Local Variable
      num = num + 1;
      a = a + 1;
      printf("a = %d\t and Num =%d\n",a,num);
}
  funct();
  funct();
  funct();

}
```

Output:

```
a = 11   and Num =6
a = 12   and Num =6
a = 13   and Num =6
```

Automatic Variable

By default , in C all local variables are automatic variables as they allocate memory when they are inside a code block and deallocate the memory when it exits the code block.Using the keyword "auto" we can declare a variable as automatic variables explicitly.

```
void funct()
{
     int num1 = 5;   // implicitly automatic  Variable
     auto int num2 = 10;   // Explicitly automatic variable
}
```

External Variable

A variable prepended with the keyword 'extern' is said to be an external variable whose scope is valid across source files in C. More specifically, variable scope is not bound to a single file instead its scope is available to external files.External variables are also global variables.

file.h

```
extern int num1 = 10;  //external variable
```

program.h

```
#include <file.h>
#include <stdio.h>
main()
{
      int num2,total;
      num2=5;
      total=num1+num2;
      printf("Total is %d",total);
}
```