C/C++ Tokens

A token is the smallest element of a program that is meaningful to the compiler. Tokens can be classified as follows:

1. Keywords
2. Identifiers
3. Constants
4. Strings
5. Special Symbols
6. Operators

**Keyword:**Keywords are pre-defined or reserved words in a programming language. Each keyword is meant to perform a specific function in a program. Since keywords are referred names for a compiler, they can’t be used as variable names because by doing so, we are trying to assign a new meaning to the keyword which is not allowed. You cannot redefine keywords.

**Identifiers:**Identifiers are used as the general terminology for naming of variables, functions and arrays. These are user defined names consisting of arbitrarily long sequence of letters and digits with either a letter or the underscore(\_) as a first character. Identifier names must differ in spelling and case from any keywords. You cannot use keywords as identifiers; they are reserved for special use. Once declared, you can use the identifier in later program statements to refer to the associated value.

**There are certain rules that should be followed while naming c identifiers:**

* They must begin with a letter or underscore(\_).
* They must consist of only letters, digits, or underscore. No other special character is allowed.
* It should not be a keyword.
* It must not contain white space.
* It should be up to 31 characters long as only first 31 characters are significant.

**Constants:**Constants are also like normal variables. But, only difference is, their values can not be modified by the program once they are defined. Constants refer to fixed values. They are also called as literals.  
Constants may belong to any of the data type.**Syntax:**

**const data\_type variable\_name;** (or) **const data\_type \*variable\_name;**

**Strings:**Strings are nothing but an array of characters ended with a null character (‘\0’).This null character indicates the end of the string. Strings are always enclosed in double quotes. Whereas, a character is enclosed in single quotes in C and C++.**Declarations for String:**

* char string[20] = {‘g’, ’e’, ‘e’, ‘k’, ‘s’, ‘f’, ‘o’, ‘r’, ‘g’, ’e’, ‘e’, ‘k’, ‘s’, ‘\0’};
* char string[20] = “geeksforgeeks”;
* char string [] = “geeksforgeeks”;

**Special Symbols:**The following special symbols are used in C having some special meaning and thus, cannot be used for some other purpose.[] () {}, ; \* = #

**Operators:**Operators are symbols that triggers an action when applied to C variables and other objects. The data items on which operators act upon are called operands.  
Depending on the number of operands that an operator can act upon, operators can be classified as follows:

* **Unary Operators:** Those operators that require only single operand to act upon are known as unary operators.For Example increment and decrement operators
* **Binary Operators:** Those operators that require two operands to act upon are called binary operators.**Binary operators are classified into :**
  1. Arithmetic operators
  2. Relational Operators
  3. Logical Operators
  4. Assignment Operators
  5. Conditional Operators
  6. Bitwise Operators

**Ternary Operators:** These operators requires three operands to act upon. For Example Conditional operator(?:).

# Variables and Keywords in C

A **variable**in simple terms is a storage place which has some memory allocated to it. Basically, a variable used to store some form of data. Different types of variables require different amounts of memory, and have some specific set of operations which can be applied on them.

**Variable Declaration:**  
A typical variable declaration is of the form:

type variable\_name;

or for multiple variables:

type variable1\_name, variable2\_name, variable3\_name;

A variable name can consist of alphabets (both upper and lower case), numbers and the underscore ‘\_’ character. However, the name must not start with a number.

**Difference b/w variable declaration and definition**  
Variable declaration refers to the part where a variable is first declared or introduced before its first use. Variable definition is the part where the variable is assigned a memory location and a value. Most of the times, variable declaration and definition are done together.

Keywords: Keywords are pre-defined or reserved words in a programming language. Each keyword in a is meant to perform a specific function in a program. Since keywords are referred names for a compiler, they can’t be used as variable names because by doing so, we are trying to assign a new meaning to the keyword which is not allowed. You cannot redefine keywords

All keywords must be written in lowercase.

Data type keywords

Int: specifies the integer type of value a variable will hold

Char: specifies the character type of value a variable will hold

Loop control structure keywords

For: Loop is used when the number of passes is known in advance

While: Loop is used when the number of passes in not known in advance

Do: loop is used to handle menu-driven programs

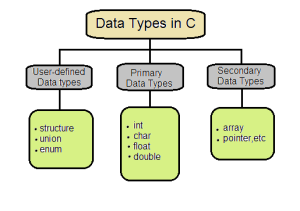
# DATA TYPES : C PROGRAMMING LANGUAGE

[Nishirika](https://www.codingeek.com/author/nishirikakumari/) | December 4, 2016 | [c programming](https://www.codingeek.com/category/tutorials/c-programming/) | [0 Comments](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#disqus_thread)

**Data Type** is the classification of the data that is taken as input, processed and results in an output. It is how we categorize data according to its type. There are three type of Data Types:

* ***Primary data types***
* ***Secondary data types***
* ***User-defined data types***

These data types are further subdivided into several other data types which you can find in the figure below:

[](https://www.codingeek.com/wp-content/uploads/2016/12/data-types-in-c.png)

Now let’s talk about these data types in detail.

Table of Contents

* [Primary Data Types](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#primary_data_types)
  + [Integer Data Type](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#integer_data_type)
  + [Character Data Type](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#character_data_type)
  + [Float(and Double) Data Type](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#floatand_double_data_type)
* [Secondary Data Types](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#secondary_data_types)
* [User-defined data types](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#user-defined_data_types)
* [Recommended -](https://www.codingeek.com/tutorials/c-programming/data-types-c-programming-language/#recommended)

### PRIMARY DATA TYPES

**Primary data type** , also known as **‘primitive data type’** or **‘fundamental data type’**, are the built-in data types that are provided by the programming language. It defines the most basic data like int, char, float,etc.

Primary data types could be of several types like an ***int*** can be unsigned int, short int, unsigned long int,etc. With such a wide range of classification and variety, the programmer has got many different data types to choose from as per the requirement and use it in their code along with the advantage of the secondary data types.

Here are the most commonly used primary data types:

* boolean
* byte
* char
* short
* int
* long
* float
* double

**‘Void’** is another primary data type which means **‘no value’**. It is usually used to define the type of return value in a function. A function with void return type does not return any value.

#### INTEGER DATA TYPE

Integer data types are used to define the variables taking integer values with or without constant values given to them. The most commonly used keyword or the data type used to define the integer type data is **‘int’**. There are other data types like **‘short’** and**‘long’** used to define integer values but they have different ranges (we use them as per the requirement of the program i.e. if we are sure that our requirement is small and it will never go beyond the range of small we shall take small and likewise). In the tutorial [constants](https://www.codingeek.com/tutorials/c-programming/constants-c-language/), we saw that the range of the constants differed for different compilers. Similarly, the size of the  integer data type (in bytes) also differs for different compilers as shown in the table below:

[size of int data types](https://www.codingeek.com/wp-content/uploads/2016/12/size-chart.png)

Note that the size of the given data types is in bytes.

The **short** integer can be used in places where small values and little storage space is required. It can boost up the runtime because it uses less space. It is declared by using the keyword **‘short’**or **‘short int’**.

The**long** integer gives us a long range or a bigger size compared to ‘short’ but it can cause our program to take more time for execution because of the storage size it offers. It is declared by using the keyword **‘long’** or **‘long int’**. The range for a long integer is –**2147483648** to +**2147483647**.

This is how we declare the integer variables:

int num,length;  
short breadth;  
short int height;  
long int volume=0;

We have another classification of the integer data type: **signed** and **unsigned int**. In case we need to take positive values only, then we can use unsigned int . Its range is **0** to **4294967295**. This is because the left most bit is free and does not need to store the sign of the number. So, we get more storage space i.e. double on the positive side. By default, signed type is declared and we do not need to use signed. **Signed int** works as an int. It’s range is the same as int .

#### CHARACTER DATA TYPE

Character data types are used to define variables taking one character as its value. The keyword used for character data type is **‘char’**. Here is how we declare character variables:

char ch, ch1=‘A’, ch2=67;

Here, in ch1 variable, we store ‘A’, i.e., the binary equivalent of the ASCII value of A(=binary of decimal 65) gets stored. And ch2 variable stores the value 67(ASCII for ‘C’). So these are actually two ways of initializing a character value.

Like integers, here we have **signed** and **unsigned** character values. The**signed char** is equivalent to **char**. For a signed char the range is **-128** to **+127**. Whereas, for an unsigned char the range is from **0** to **255**. Here’s how we declare signed(declared as **char** above) and unsigned char values:

unsigned char ch;

char ch1=128;

Surprised to see why I put the value of ch1 as 128? Here’s another thing about this data type. As mentioned before **char**has a range of **+127** but we have put the value **128** here. What is going to be the output?

What happens here is that once we reach the end of the range, the other side of the range is accessed. It goes back to -128 again. So in this case, when it comes across 128 it goes back to the beginning and accesses the char at the ASCII -128.

#### FLOAT(AND DOUBLE) DATA TYPE

Float and double data types are used to define variables that take up a decimal value or an exponential value. The keyword used for float and double data type are **‘float’** and **‘double’** respectively.

* **Float** has a range of –**3.4e38** to **+3.4e38** and its size is **4 bytes**.
* **Double** has a range of **-1.7e308** to **+1.7e308**and its size is **8 bytes**.
* Another data type that is offered by programming languages is **‘long double’** which has a range of –**1.7e4932** to **+1.7e4932** and its size is **10 bytes**. Here is how we declare float and double variables:

float length, area=0.0;

double radius, area=0.0;

### SECONDARY DATA TYPES

**Secondary data types** are basically derived from the primary data types. Let’s have a look at a few secondary data types:

* ***Array***: An **array**is a collection of data of the same data type. These are declared under the same variable and are accessed using it. If we declare an integer array, all the values in the array have to be an integer. Similarly, for a character array all the elements of the array are characters and same goes for double and every other data type. An array is declared as follows:  
  **int a[50];** //Declaration  
  This array has an integer data type and can store 50 integer elements.
* **Pointers**: A pointer contains the address of a variable in the program .  
  We declare the pointer as:  
  **int \*ip;** //Declaration  
  A pointer declared as integer type stores the address of the integer type variable. Similarly, a pointer declared as char type stores the address of character type variable and so on.

### USER-DEFINED DATA TYPES

User-defined data type defines the data in the way that the programmer chooses. Let’s have a look at these commonly used user-defined data types:

* ***Structures***: It is a collection of variables of different data types represented by the same name. Unlike array where we had to store all the data of the same type in the variable, here one can store data of different data types under the same variable name. It is mostly used to form records where different specifications needs to be stored under the same name. The **struct** keyword is used to define a structure.

* ***Union***: Another data type which is very similar to structures. It allows the programmer to store data of different data type in the same memory location. A union can have multiple members but it can store only one member at a particular time. The keyword **union** is used to define a Union.
* ***Enum***: Enum or Enumeration is used to declare the variables and consists of integral constants. The keyword **enum**is used to define the enumeration data type.  
  Eg: enum identifier{element1, element2,……., elementn};  It assigns the value from **0** to **n** to the elements present inside the identifier sequentially.