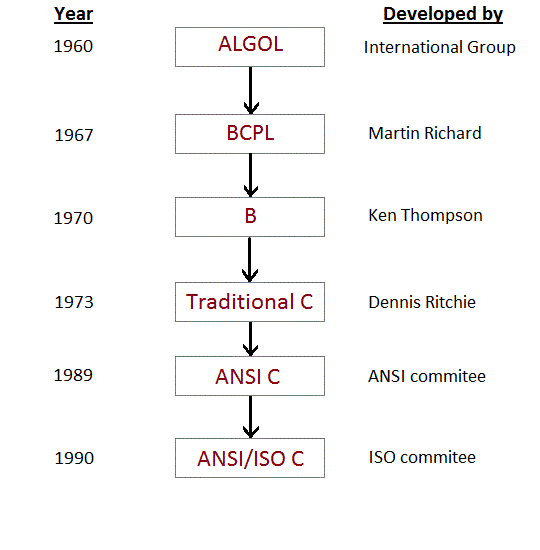
Overview of C Language

C is a structured programming language developed by Dennis Ritchie in 1972 at Bell Laboratories. It is one of the most popular computer languages today because of its structure, high-level abstraction, machine independent feature etc. C language was developed to write the UNIX operating system, hence it is strongly associated with UNIX, which is one of the most popular network operating system in use today and heart of internet data superhighway.

### **History of C language**

C language has evolved from three different structured language ALGOL, BCPL and B Language. It uses many concepts from these languages while introduced many new concepts such as datatypes, struct, pointer etc. In 1988, the language was formalised by **American National Standard Institute**(ANSI). In 1990, a version of C language was approved by the **International Standard Organisation**(ISO) and that version of C is also referred to as C89.



The idea behind creating C language was to create an easy language which requires a simple compiler and enables programmers to efficiently interact with the machine/system, just like machine instructions.

C language compiler converts the readable C language program into machine instructions.

Importances/ Advantages of C

* Robust Language:

It is a robust language with rich set of built-in functions and operators that can be used to write any complex program.

* Efficient and Fast

Programs Written in C are efficient and fast. This is due to its variety of data type and powerful operators.

* Structured Language

C is a structured language as it has a fixed structure. A program can be divided into a number of modules or blocks. A collection of these modules makes a complete program.

* Extendibility

A C program may contain a number of user defined functions. We can add our own user defined functions to the C library if required.

* Portable

C is highly portable this means that programs once written can be run on another machines with little or no modification.

* Rich System Library

There are large numbers of built in functions, keywords and operators in C’s System library organized in different header files. Using these built-in functions saves our time and effort.

First C program and its Structure

Lets see how to write a simple and most basic C program:

#include <stdio.h>

int main()

{

printf("Hello World"); //single line comment

return 0;

/\* multi line comments /\*

}

**Output:**

Hello world

## **Different parts of C program**

* Pre-processor
* Header file
* Function
* Variables
* Statements & expressions
* Comments

All these are essential parts of a C language program.

### **Pre-processor**

#include is the first word of any C program. It is also known as a **pre-processor**. The task of a pre-processor is to initialize the environment of the program, i.e to link the program with the header files required.

So, when we say #include <stdio.h>, it is to inform the compiler to include the **stdio.h** header file to the program before executing it.

### **Header file**

A Header file is a collection of built-in(readymade) functions, which we can directly use in our program. Header files contain definitions of the functions which can be incorporated into any C program by using pre-processor #include statement with the header file.

For example, to use the printf() function in a program, which is used to display anything on the screen, the line #include <stdio.h> is required because the header file **stdio.h** contains the printf() function.

### **Function**

Functions are main building blocks of any C Program. Every C Program will have one or more functions and there is one mandatory function which is called main()function. This function is prefixed with keyword int which means this function returns an integer value when it exits. This integer value is returned using return statement.

The C Programming language provides a set of built-in functions. printf()is a C built-in function which is used to print anything on the screen.

**Variables:**

Variables are used to hold numbers, strings and complex data for manipulation.

**Statements & Expressions:**

Expressions combine variables and constants to create new values. Statements are expressions, assignments, function calls, or control flow statements which make up C programs.

### **Comments**

We can add comments in our program to describe what we are doing in the program. These comments are ignored by the compiler and are not executed.

To add a single line comment, start it by adding two forward slashes // followed by the comment.

To add multiline comment, enclose it between /\* .... \*/, just like in the program above.

### **Return statement - return 0;**

A return statement is just meant to define the end of any C program.

**Data Types**

Data types in c refer to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.

There are four data types in C language. They are,

|  |  |
| --- | --- |
| **Types** | **Data Types** |
| Basic Data Type | int, char, float, double |
| Enumerated Data Type | enum |
| Derived data type | pointer, array, structure, union |
| Void data type | void |

Basic Data Type

There are several basic data types in C. They are as discussed below;

1. Integer Data Type :

* Integer data type allows a variable to store numeric values.
* “int” keyword is used to refer integer data type.
* The storage size of int data type is 2 or 4 or 8 byte.
* It varies depend upon the processor in the CPU that we use. If we are using 16 bit processor, 2 byte (16 bit) of memory will be allocated for int data type.
* Like wise, 4 byte (32 bit) of memory for 32 bit processor and 8 byte (64 bit) of memory for 64 bit processor is allocated for int datatype.
* int (2 byte) can store values from -32,768 to +32,767
* int (4 byte) can store values from -2,147,483,648 to +2,147,483,647.
* If you want to use the integer value that crosses the above limit, you can go for “long int” and “long long int” for which the limits are very high.

1. Character Data Type :

* Character data type allows a variable to store only one character.
* Storage size of character data type is 1. We can store only one character using character data type.
* “char” keyword is used to refer character data type.
* For example, ‘A’ can be stored using char datatype. You can’t store more than one character using char data type.

1. Floating Point Data Type :

Floating point data type consists of 2 types. They are,

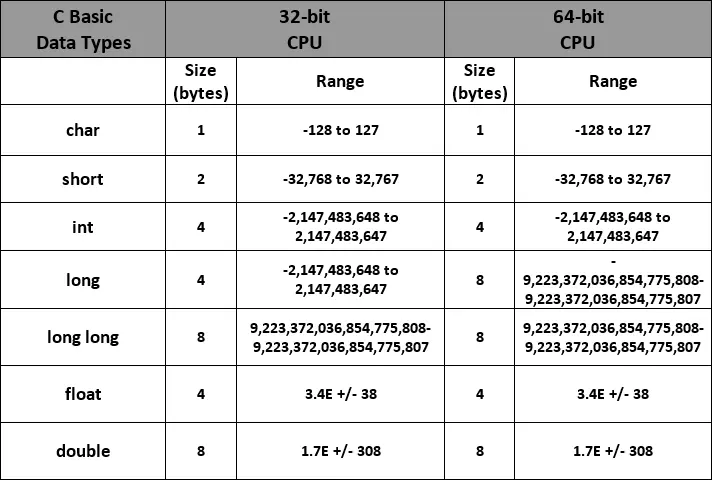
* float
* double

**a) . float:**

* Float data type allows a variable to store decimal values.
* Storage size of float data type is 4 byte.
* We can use up-to 6 digits after decimal using float data type.
* The range for float data type is from 3.4 E-38 to 3.4E +38.
* For example, 10.456789 can be stored in a variable using float data type.

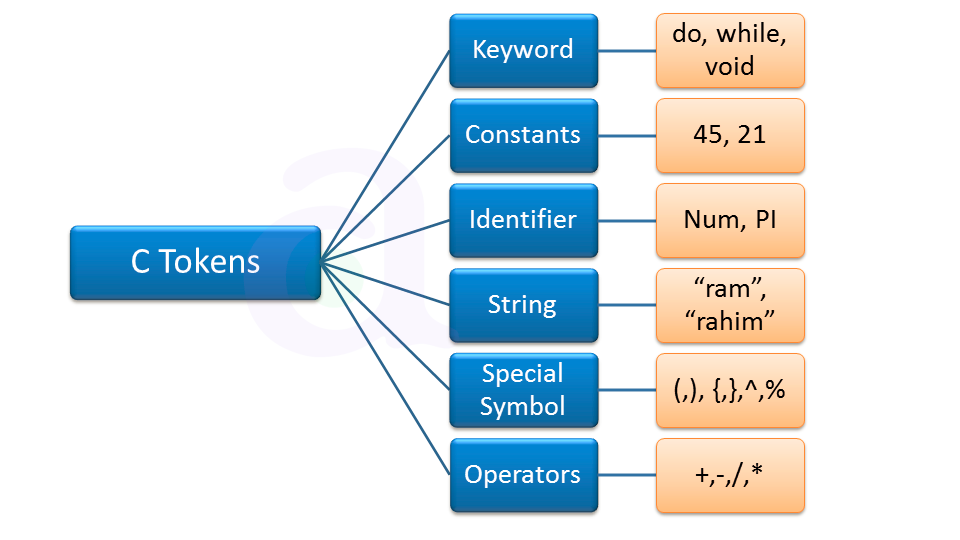
# **b). double:**

* Double data type is also same as float data type which allows up-to 10 digits after decimal.
* Storage size of double float data type is 4 byte.
* The range for double datatype is from 1.7E-308 to 1.7E+308.



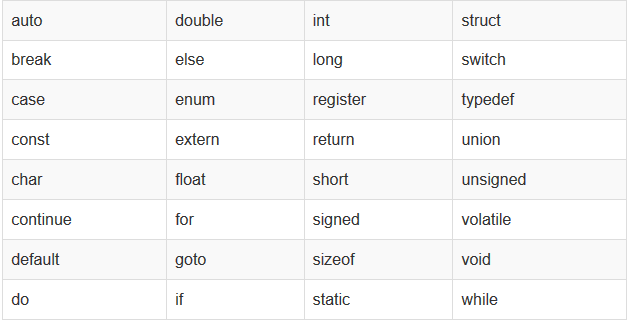
**C Tokens**

C tokensare the basic buildings blocks in C language which are constructed together to write a C program. Each and every smallest individual units in a C program are known as C tokens. C tokens are of six types:



**Keywords**

Keywords are preserved words that have special meaning in C language. The meaning of C language keywords has already been described to the C compiler. These meaning cannot be changed. Thus, keywords cannot be used as variable names because that would try to change the existing meaning of the keyword, which is not allowed. There are a total 32 keywords in C language.



Identifiers:

In C language identifiers are the names given to variables, constants, functions and user-defined data. These identifiers are defined against a set of rules.

#### **Rules for an Identifier**

1. An Identifier can only have alphanumeric characters(a-z , A-Z , 0-9) and underscore(\_).
2. The first character of an identifier can only contain alphabet(a-z , A-Z) or underscore (\_).
3. Identifiers are also case sensitive in C. For example **name** and **Name** are two different identifiers in C.
4. Keywords are not allowed to be used as Identifiers.
5. No special characters, such as semicolon, period, whitespaces, slash or comma are permitted to be used in or as Identifier.

Constants

Constants in C are the fixed values that are used in a program, and its value remains the same during the entire execution of the program. They are also known as literals.

Constants are categorized into two basic types, and each of these types has its subtypes/categories.

1. Numeric Constants
   * Integer Constants
   * Real Constants
2. Character Constants
   * Single Character Constants
   * String Constants
   * Backslash Character Constants

**Integer Constant**

It's referring to a sequence of digits. Integers are of three types viz:

1. Decimal Integer
2. Octal Integer
3. Hexadecimal Integer

Example: 15, -265, 0, 99818, +25, 045, 0X6

**Real Constant**

The numbers containing fractional parts like 99.25 are called real or floating points constant.

**Single Character Constant**

It simply contains a single character enclosed within ' and ' (a pair of single quote). It is to be noted that the character '**8**' is not the same as **8**. Character constants have a specific set of integer values known as ASCII values (American Standard Code for Information Interchange).

Example: 'X', '5', ';'

**String Constant**

These are a sequence of characters enclosed in double quotes, and they may include letters, digits, special characters, and blank spaces. It is again to be noted that "**G**" and '**G**' are different - because "G" represents a string as it is enclosed within a pair of double quotes whereas 'G' represents a single character.

Examples: "Hello!", "2015", "2+1"

**Backslash Character constant**

C supports some character constants having a backslash in front of it. The lists of backslash characters have a specific meaning which is known to the compiler. They are also termed as "Escape Sequence".

Example: \n is used for new line .

**String in C**

In C programming, a string is a sequence of characters terminated with a null character \0. For example:

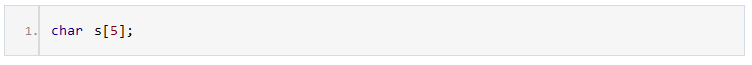


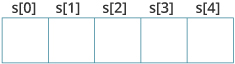
When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default.



Declaration:

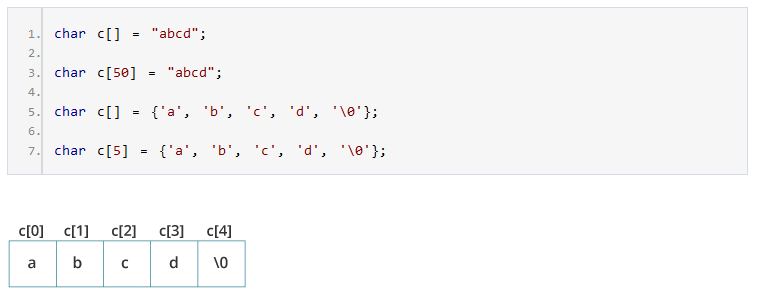
String can be declared as:





Initialization:

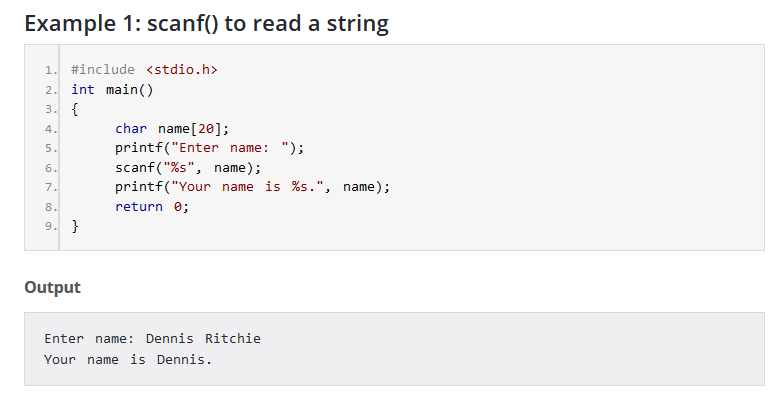
You can initialize strings in a number of ways.



Reading String from User

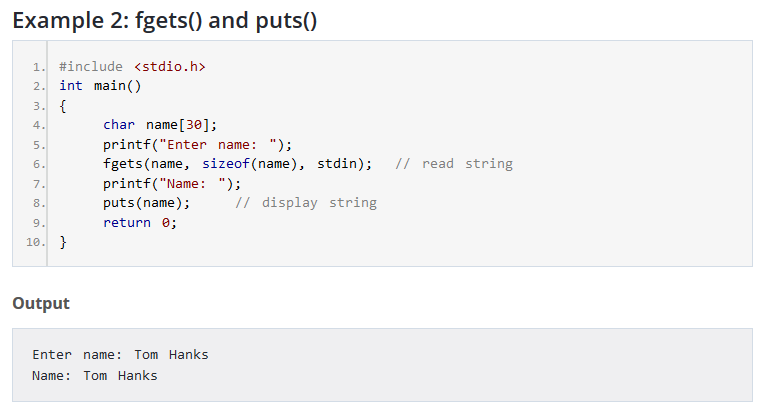
You can use the scanf() function to read a string.

The scanf() function reads the sequence of characters until it encounters [whitespace](https://stackoverflow.com/questions/30033582/what-is-the-symbol-for-whitespace-in-c) (space, newline, tab etc.).



Even though Dennis Ritchie was entered in the above program, only "Dennis" was stored in the name string. It's because there was a space after Dennis.

**Note**:You can use the fgets() function to read a line of string. And, you can use puts() to display the string.



Here, we have used fgets() function to read a string from the user.

fgets(name, sizeof(name), stdlin); // read string

The sizeof(name) results to 30. Hence, we can take a maximum of 30 characters as input which is the size of the name string.

To print the string, we have used puts(name);.

**Note:** The gets() function can also be to take input from the user. However, it is removed from the C standard.

It's because gets() allows you to input any length of characters. Hence, there might be a buffer overflow.

Commonly used C String Functions:

There are several string functions provided by C. These string functions are defined in string.h header file. Therefore, we need to include string.h header file using #include<string.h> at the top of the program.

Commonly used string functions in C are:

* **strlen()**- Gives the length of string
* **strcat()**- Concatenates string at the end of another string.
* **strcmp()**- Compares two strings.
  + Returns 0 if str1 is same as str2.
  + Returns <0 if strl < str2.
  + Returns >0 if str1 > str2
* **strcpy()**- Copies one string to another.
* **strupr()**- Converts string to uppercase.
* **strlwr()**- Converts string to lowercase.

**Examples:**

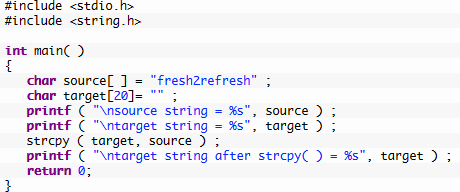
1. Example of strlen() function

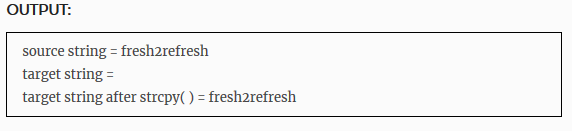


1. Example of strcat() function



1. Example of strcpy() function





1. Example of strcmp() function



1. Example of strupr() function
2. Example of strlwr() function

**Format Specifier in C**

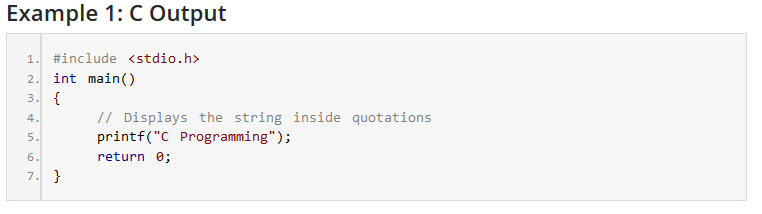
The format specifier is used during input and output. It is a way to tell the compiler what type of data is in a variable during taking input using scanf() or printing using printf().

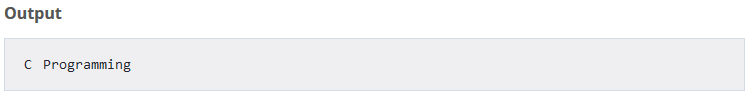
List of Format Specifiers in C:

|  |  |
| --- | --- |
| Data Type | Format Specifier |
| int | %d |
| char | %c |
| float | %f |
| double | %lf |
| long int | %li |
| long long int | %lli |
| string | %s |

**C output using printf()**

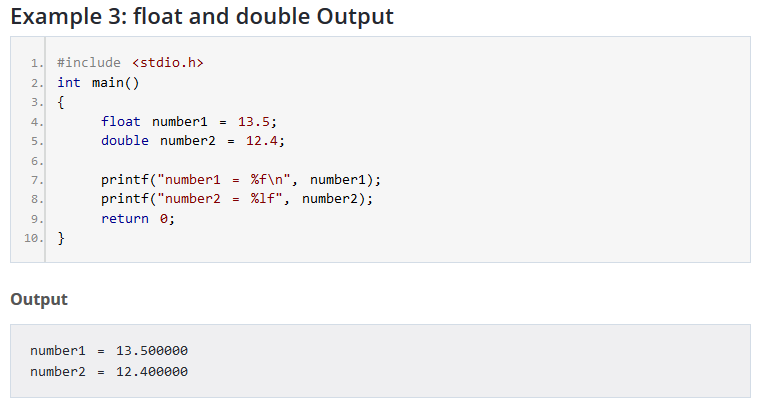
In C programming, printf() is one of the main output function. The function sends formatted output to the screen. For example,



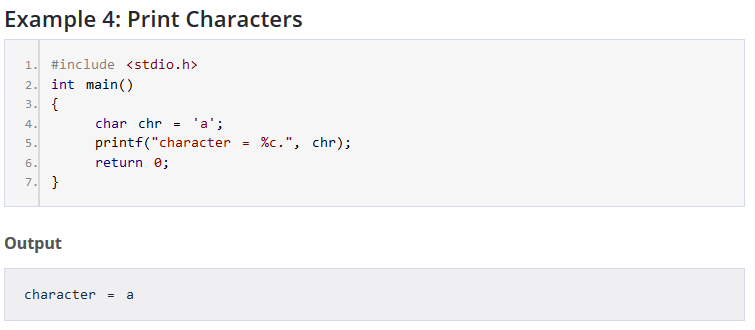




We use %d format specifier to print int types. Here, the %d inside the quotations will be replaced by the value of testInteger.



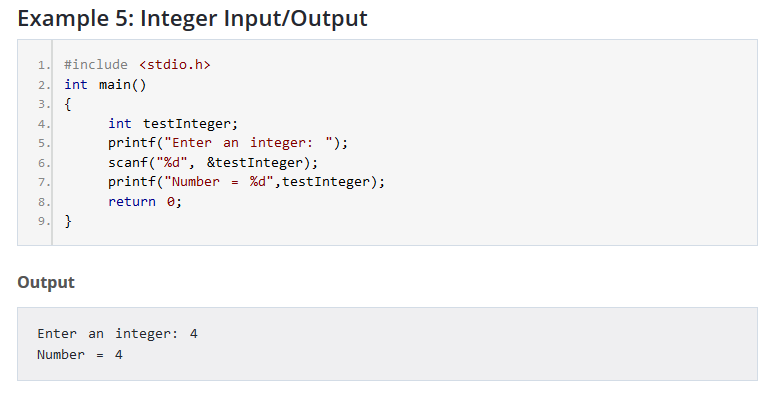
To print float, we use %f format specifier. Similarly, we use %lf to print double values.



To print char, we use %c format specifier.

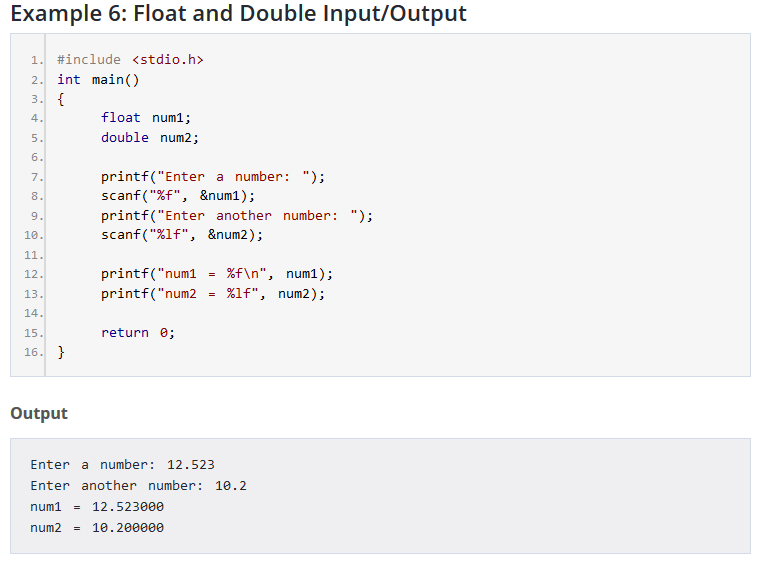
## **C Input using scanf()**

In C programming, scanf() is one of the commonly used function to take input from the user. The scanf() function reads formatted input from the standard input such as keyboards.

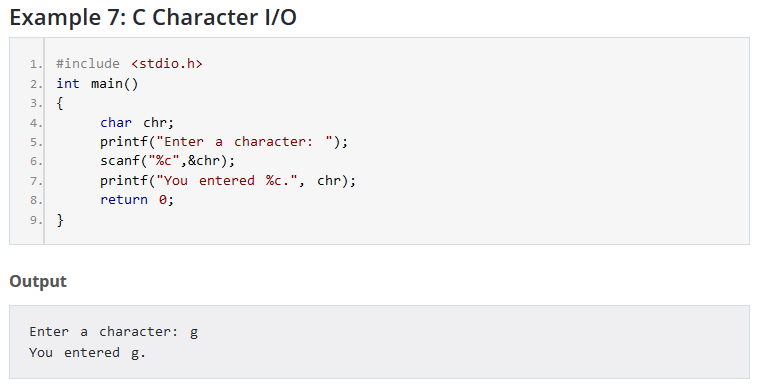


Here, we have used %d format specifier inside the scanf() function to take int input from the user. When the user enters an integer, it is stored in the testInteger variable.

Notice, that we have used &testInteger inside scanf(). It is because &testInteger gets the address of testInteger, and the value entered by the user is stored in that address.



We use %f and %lf format specifier for float and double respectively.



## **I/O Multiple Values**

Here's how you can take multiple inputs from the user and display them.



Escape Characters (Escape Sequence)

An escape sequence is a sequence of characters that does not represent itself ,but is translated into a sequence of characters that may be difficult or impossible to represent directly. Here is the list of such escape codes:

|  |  |  |
| --- | --- | --- |
| **Escape Sequence** | **Meaning** | **Elucidation** |
| \n | New Line | Used to shift the cursor control to the new line |
| \t | Horizontal Tab | Used to shift the cursor to a couple of spaces to the right in the same line. |
| \a | Audible bell | A beep is generated indicating the execution of the program to alert the user. |
| \r | Carriage return | Used to position the cursor to the beginning of the current line. |
| \\ | Backslash | Used to display the backslash character |
| \? | Question Mark | Used to print the question mark |
| \’ | Single Quote | Used to print single quote |
| \’’ | Double Quote | Used to print double quote |
| \0 | Null Character | Used for Null Characters |

https://www.w3schools.in/c-tutorial/operators/