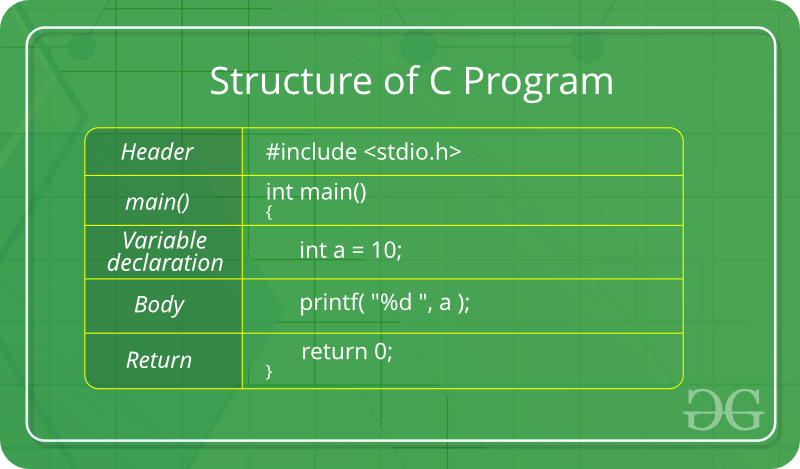
* *Structure Of C*



# -> preprocessor

include -> To add something

stdio -> standrd input and output

\n -> new line

.h -> header file(main file). Header files add

functionality to C programs.

printf -> print formatted

// -> single line comment ( forward slashes)

/\* -> multi line comment( Start with )

\*/ -> multi line comment( End with )

ctrl + / -> comment in vs code

* *What is C*
* C is a general-purpose programming language created by Dennis Ritchie at the Bell Laboratories in 1972 And associated with UNIX, as it was developed to write the UNIX operating system.
* UNIX - an operating system analogous to DOS and Windows, supporting multiple concurrent users.
* Concurrent - also known as a floating user is one individual, machine, program, or device within or outside your organization.

* *Why Learn C*
* C is very fast, compared to other programming languages
* If you know C, you will have no problem learning other popular programming languages as the syntax is similar.
* C is very versatile; it can be used in both applications and technologies
* Versatile - અનેક વિષયોમાં ગતિવાળું
* *Difference between C and C++*
* The main difference between C and C++ is that C++ support classes and objects, while C does not.
* *What is language*
* The computer language is defined as code or syntax which is used to write programs or any specific applications. It is used to communicate with computers.

Broadly the computer language can be classified into three categories:- Assembly language

machine language

high-level language.

* *Difference between program and programming*

Program : set of commands coded to do some specific task.

Programming : means coding a program.

|  |  |
| --- | --- |
| \n | For print in new line |
| \t | Inserts a tab in the text at this point |
| \” | For highlight any word of sentence |
| \0 | Null character |
| \\ | Insert a backslash character |

* *C Variable*

* In case of string we need to assign character length by [].

Ex. char name[20] = “hello world”;

* We also declare a variable without assigning the value, and assign the value later.

Like… int myNum; OR int myNum=15;

myNum = 15;

* Note: If you assign a new value to an existing variable, it will overwrite the previous value.

Ex. int myNum = 15;  // myNum is 15  
 myNum = 10;  // Now myNum is 10

* Format specifiers are used together with the printf() function to tell the compiler what type of data the variable is storing. It is basically a placeholder for the variable value.
* starts with a percentage sign %
* It is surrounded by double quote in printf function

Ex. int myNum = 15;  
 printf("%d", myNum);

* To print other types, use %c for char and %f for float:
* To combine both text and a variable, separate them with a comma inside the printf() function
* To add a variable to another variable, you can use the + operator

|  |  |
| --- | --- |
| **Format specifier** | **Data Type** |
| %d or %i | Integer |
| %f | Float or Real |
| %lf | Double |
| %c | Single character |
| %s | String |
| %u | Unsigned integer |
| %ld | Long integer |
| %Lf | Long double |

***Identifiers***

* All C **variables** must be **identified** with **unique names**.These unique names are called **identifiers**
* **General rules**for naming variables:

1. contain only letters, digits and underscores.
2. must begin with a letter or an underscore (\_)
3. Names are case sensitive
4. cannot contain whitespaces or special characters like !, #, %, etc
5. Standard language keyword can not be used as variable name
6. Should have less than 31 character

***Basic data Type***

* The data type specifies the size and type of information the variable will store.

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size** | **Description** |
| int | 2 or 4 bytes | Stores whole numbers, without decimals |
| float | 4 bytes | Stores fractional numbers, containing one or more decimals. Sufficient for storing 7 decimal digits |
| double | 8 bytes | Stores fractional numbers, containing one or more decimals. Sufficient for storing 15 decimal digits |
| char | 1 byte | Stores a single character/letter/number, or ASCII values |

***Constant in C***

* Real constant

All number with positive, negative and decimal places.

Ex. -32.33,43.56

* Integer constant

All number with positive, negative and no decimal places.

Ex.1990,373,43

* Character constant

All character with Alphabet, Digit ,special symbol($).

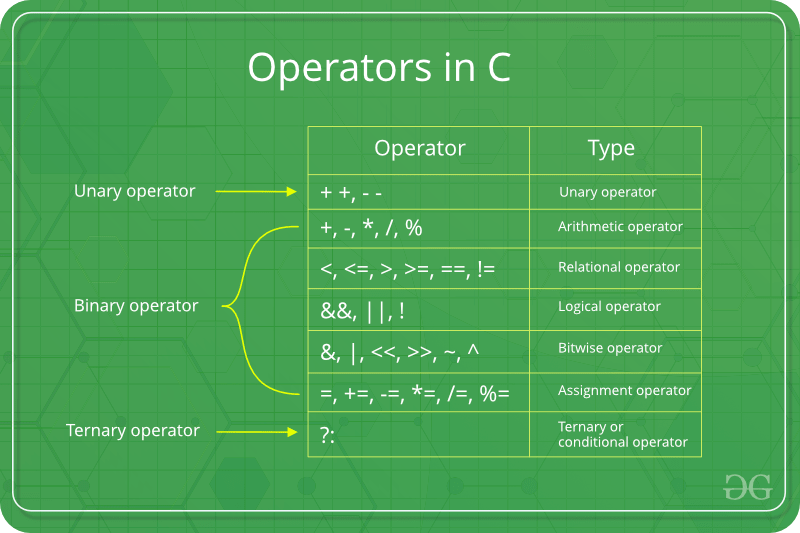
Ex. A-Z or a-z , 0-9 , $ .

* String constant

Collection of character in double quote.

Ex. Full sentence

* Operators are used to perform operations on variables and values.



* Types of Unary Operator in C
* increment (++)
* decrement (- -)
* **Increment**

We use this operator to increment the overall value of any given variable by value of 1. We can perform increment using two major ways:

* Prefix increment.

The operator in this method precedes the given operand (Example, ++p). Thus, the value of the operand gets altered *before* we finally use it.

int p = 1;

int q = ++p; // q = 2

* Postfix increment.

The operator in this method follows the given operand (Example, p++). Thus, the value of the available operand gets altered after we use it.

int p = 1;

int q = p++; // q = 1

int r = p; // r = 2

* **Decrement**

We use this operator to decrement the overall value of any given variable by a value of 1.

* Prefix decrement.

The operator in this method precedes the given operand (Example, --p). Thus, the value of the operand gets altered before we finally use it.

int p = 1;

int q = --p; // q = 0

* Postfix decrement

The operator in this method follows the given operand (Example, p–). Thus, the value of the available operand gets altered after we use it.

int p = 1;

int q = p–; // q = 1

int r = p; // r = 0

* **Arithmetic Operators**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| + | Addition | Adds together two values | x + y |  |
| - | Subtraction | Subtracts one value from another | x – y |  |
| \* | Multiplication | Multiplies two values | x \* y |  |
| / | Division | Divides one value by another | x / y |  |
| % | Modulus | Returns the division remainder | x % y |  |

* **Relational Operators / Comparison operator**

Comparison operators are used to compare two values.

**Note:** The return value of a comparison is either true (1) or false (0).

|  |  |  |  |
| --- | --- | --- | --- |
| == | Equal to | x == y |  |
| != | Not equal | x != y |  |
| > | Greater than | x > y |  |
| < | Less than | x < y |  |
| >= | Greater than or equal to | x >= y |  |
| <= | Less than or equal to | x <= y |  |

* **Logical Operators**

Logical operators are used to determine the logic between variables or values.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| && | Logical and | Returns true if both statements are true | x < 5 &&  x < 10 |  |
| || | Logical or | Returns true if one of the statements is true | x < 5 || x < 4 |  |
| ! | Logical not | Reverse the result, returns false if the result is true | !(x < 5 && x < 10) |  | |

* **Shortend Operators**

|  |  |  |  |
| --- | --- | --- | --- |
| += | x += 3 | x = x + 3 |  |
| -= | x -= 3 | x = x – 3 |  |
| \*= | x \*= 3 | x = x \* 3 |  |
| /= | x /= 3 | x = x / 3 |  |
| %= | x %= 3 | x = x % 3 |  | |

* **Ternary Operators**

The conditional operator is of the form

*(condition) ? TRUE : FALSE ;*

Or the syntax will also be in this form

*variable = (condition) ? Expression2 : Expression3*

* **Decision control statement**

C language provides statements that can alter the flow of a sequence of instructions.

These statements are called control statements.

These statements help to jump from one part of the program to another.

* Control Statement

1. If statement
2. If else statement.
3. Nested if statement.
4. Else if ladder statement.
5. Switch statement.
6. ***If statement***

It is very frequently used in decision making and allowing the flow of program execution.

**if (condition)**

**{  
   Statement *// block of code to be executed if the condition is true*  
 }**

The condition is logical operator which are used in the condition statement.

The condition part should not end with a semicolon, since the condition and statement should be put together as a single statement.

If condition is true the block will be execute.

1. ***If else statement***

Use the else statement to specify a block of code to be executed if the condition is false.

**if (condition)**

**{  
  Statement 1 *//block of code to be executed if the condition is true*  
}**

**else**

**{  
  Statement 2*//block of code to be executed if the condition is false*  
}**

1. ***Nested if statement***

The if statement may itself contain another if statement is known as nested if statement.

If (condition1)

If (condition2)

Statement-1;

Else

Statement-2;

Else

Statement-3;

Here if one condition is true then inner condition will be executed. So one block of code will only be executed if two conditions are true.

Condition 1 is tested first and then condition 2 is tested. The second if condition is nested in the first.

The second if condition is tested only when the first condition is true else the program flow will skip to the corresponding else statement.

1. ***else if ladder***

When many conditions have to be checked we may use the ladder else if statement.

As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the C else-if ladder is bypassed.

If none of the conditions is true, then the final else statement will be executed.

if (condition)

statement 1;

else if (condition)

statement 2;

….

else

default statement x;



As soon on the true condition is found, the statement associated with it is executed and the control is transferred to the statement – x (skipping the rest of the ladder.

When all the condition becomes false, the final else containing the default statement will be executed.

1. ***Switch statement***

The switch statement allows a program to select one statement for execution out of a set of alternatives.

During the execution of the switch statement only one of the possible statements will be executed the remaining statements will be skipped.

we can’t use float datatype in switch and also not use relational and logical operator.

Switch (expression)

{

case label-1:

statements;

break;

case label-2:

statements;

break;

case label-n:

statements;

break;

………………

case default:

break;

}

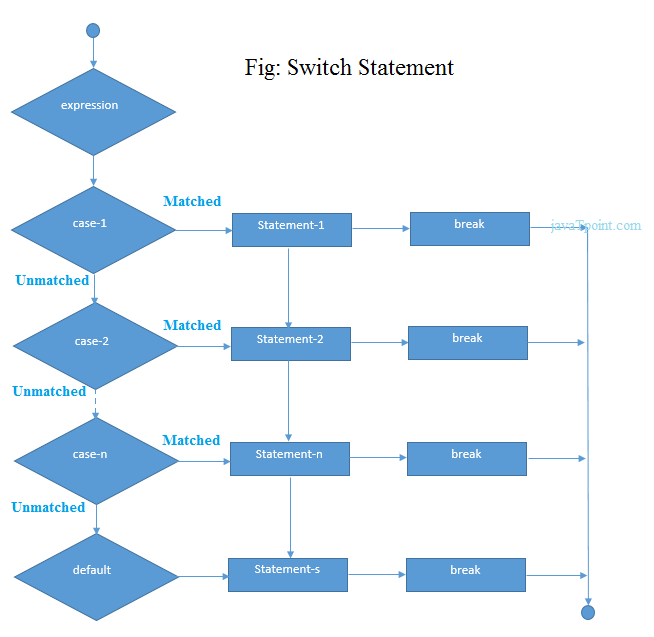
In a switch statement, the “case value” can be of “char” and “int” type

Following are some of the rules while using the switch statement:  
1. There can be one or N numbers of cases.  
2. The values in the case must be unique.  
3. Each statement of the case can have a break statement. It is optional.

**Some important keywords:**

**1) Break:**This keyword is used to stop the execution inside a switch block. It helps to terminate the switch block and break out of it.

**2)** **Default:** This keyword is used to specify the set of statements to execute if there is no case match



* ***Decision control statement***

The process of repeatedly executing a collection of statements is called looping.

But if the condition is given in such logic that the repetition continues any number of times with no fixed condition to stop looping those statements, then this type of looping is called infinite looping*.*

C supports the following types of loops:

* [while loops](https://www.w3schools.in/c-programming/loops/while)

while loops statement allows to repeatedly run the same block of code until a condition is met.

while loop has one control condition, and executes as long the condition is true.

The condition of the loop is tested before the body of the loop is executed, hence it is called an **entry-controlled** loop.

Syntex:

While (condition)

{

statement(s);

Incrementation;

}

* [do-while loops](https://www.w3schools.in/c-programming/loops/do-while)

do-while loop is very similar to the while loops, but it always executes the code block at least once and as long as the condition remains true.

This is an **exit-controlled loop.**

Syntex:

do

{

statement(s);

}

while( condition );

* [for loops](https://www.w3schools.in/c-programming/loops/for)

for loops is very similar to a while loops in that it continues to process a block of code until a statement becomes false, and everything is defined in a single line.

for loop is also **entry-controlled** loop.

Syntex:

for ( init; condition; increment )

{

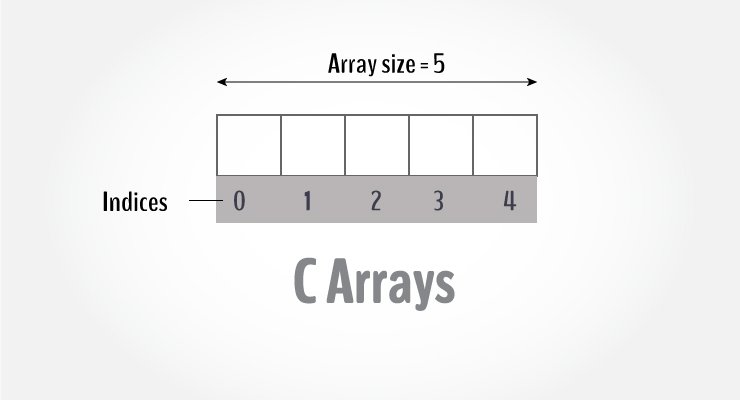
statement(s);

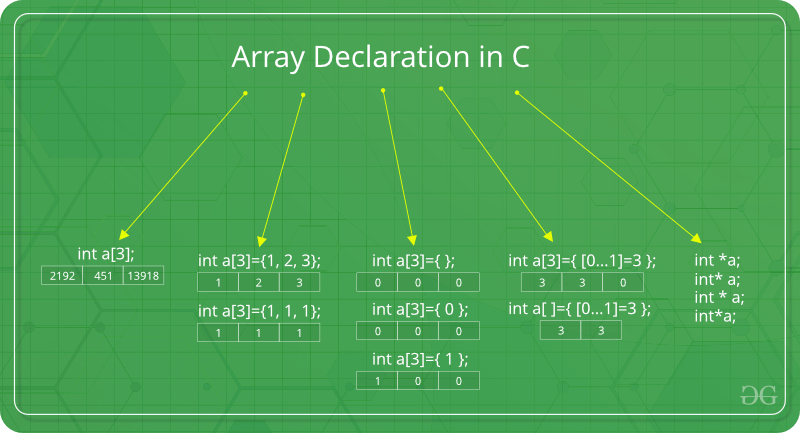
}

* Array is variable that can store multiple values of similar data type.
* In the C programming language, an array can be **One-Dimensional**, **Two-Dimensional,** and **Multidimensional**.
* Declaration of variable:

Datatype arrayName [ size ];

* array size must be an integer constant greater than zero.





* Swap Method

1. c=a

a=b

b=c

1. a=a+b

b=a-b

a=a-b

* LCM

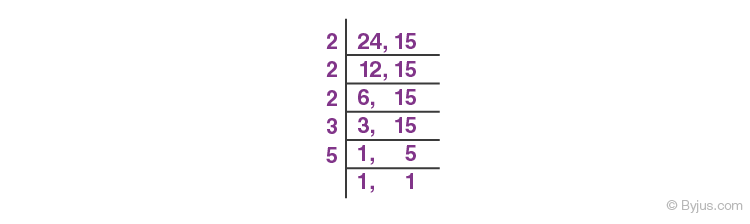
### LCM by Division Method

For example, let us take two positive integers 24 and 15.

24 = 24,48,72,96,120,144, …

15 = 30, 45, 60, 75, 90, 105,120,135,…

You can see, the least common multiple or the smallest common multiple of two numbers, 24 and 15 is 120.



LCM of 24 and 15 = 2 × 2 × 2 × 3 × 5 = 120

LCM by Prime Factorization Method

* List the **prime factors** of each number first.  
  24 = 2 × 2 x 2 × 3  
  15 = 3 × 5
* Then multiply each factor the **most number of times** it occurs in any number.

If the same multiple occurs more than once in both the given numbers, then multiply the factor the most number of times it occurs.

The occurrence of Numbers in the above example:  
**2**: three times  
**3**: one times  
**5**: one time  
LCM = 2 × 2 x 2 × 3 × 5 = 120

* HCF

HCF by Prime Factorization Method

Take an example of finding the highest common factor of 144, 104 and 160.

Now let us write the prime factors of 144, 104 and 160.

144 = 2 × 2 × 2 × 2 × 3 × 3

104 = 2 × 2 × 2 × 13

160 = 2 × 2 × 2 × 2 × 2 × 5

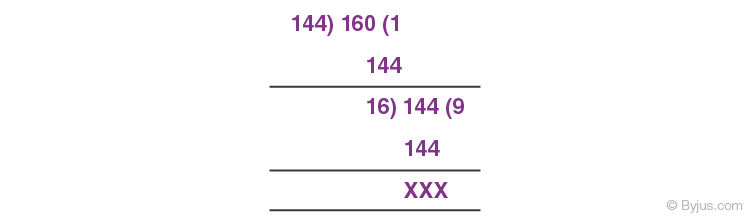
The common factors of 144, 104 and 160 are 2 × 2 × 2 = 8

Therefore, HCF (144, 104, 160) = 8

HCF by Division Method

Find the HCF of 144 and 160 by division method.

Since 160>144, so the dividend will be 160 and the divisor will be 144.  
By using the division method, we get:



Hence, we can see here 16 is the highest number which divides 160 and 144.  
Therefore, HCF (144, 160) = 16

## HCF and LCM Formula

The formula which involves both HCF and LCM is:

|  |
| --- |
| Product of Two numbers = (HCF of the two numbers) x (LCM of the two numbers)  A x B = H.C.F.(A,B) x L.C.M.(A,B) |

In C programming, the one-dimensional array of characters are called strings, which is terminated by a null character '\0'.

OR

String is a sequence of character terminated with a null character ‘\0’.

* String Declaration :

There are two ways to declare a string in C programming:

1. Through an array of characters.

char name[6];

1. Through pointers.

char \*name;

* String operation(string.h)

strlen() : No. of character in string

strcat() : Adding two or more string

strcmp() : Compare two string

strcpy() : Copies one string over another

strlwr() : Convert string to lowercase

strupr() : Convert string to uppercase

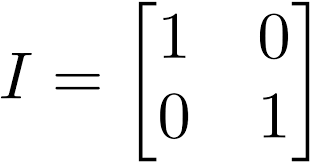
strrev() : Reverse a string

* Multiplication of two matrix

Condition for multiplication :

* No of Column in first matrix = No of Row in second matrix
* The result or product matrix became = No of row of first matrix & No of column of second matrix
* If AB and BA are both defined, it is not necessary that AB = BA.
* The multiplication of two matrices is not commutative. (AB ≠BA)
* If A, B and C are the three matrices, the associative property of matrix multiplication states that, (AB) C = A(BC)
* I. A = A

Where I is identity matrix



* If a matrix is multiplied by a zero matrix, the result matrix is a zero matrix. A\*0=0.

A function is a set of statement that take input ,do some specific computation and produce output.

A function consist of two parts:

Declaration: the function's name, return type, and parameters (if any)

Definition: the body of the function (code to be executed)

Types of function :

1.built-in function

- Also called as library function.

- Provided by system

- Stored in library files

Ex. Scanf() , printf() , strcpy() etc..

2.user defined function

-Created by user for program

-Note that when you are working with multiple parameters, the function call must have the same number of arguments as there are parameters, and the arguments must be passed in the same order.

Syntax :

Int bio (int a) -> Return type : Function name : Parameter