

C Programming Tutorial

History of C Language

C programming language was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in U.S.A.

Dennis Ritchie is known as the founder of c language.

It was developed to overcome the problems of previous languages such as B, BCPL etc.

Initially, C language was developed to be used in UNIX operating system. It inherits many features of previous languages such as B and BCPL.

Let's see the programming languages that were developed before C language.

Language	Year	Developed By
Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee

Features of C Language

C is the widely used language. It provides a lot of features that are given below.

- 1. Simple:**
C is a simple language in the sense that it provides structured approach (to break the problem into parts), rich set of library functions, data types etc.
- 2. Machine Independent or Portable:**
Unlike assembly language, c programs can be executed in many machines with little bit or no change. But it is not platform-independent.
- 3. Mid-level programming language:**
C is also used to do low level programming. It is used to develop system applications such as kernel, driver etc. It also supports the feature of high level language. That is why it is known as mid-level language.
- 4. Structured programming language:**
C is a structured programming language in the sense that we can break the program into parts using functions. So, it is easy to understand and modify.
- 5. Rich Library:**
C provides a lot of inbuilt functions that makes the development fast.
- 6. Memory Management:**
It supports the feature of dynamic memory allocation. In C language, we can free the allocated memory at any time by using function.
- 7. Fast Speed:**
The compilation and execution time of C language is fast.
- 8. Pointers:**
C provides the feature of pointers. We can directly interact with the memory by using the pointers. We can use pointers for memory, structures, functions, array etc.
- 9. Recursion:**
In c, we can call the function within the function. It provides code reusability for every function.
- 10. Extensible:**
C language is extensible because it can easily adopt new features.

C Programming Tutorial

The C Compiler

The source code written in source file is the human readable source for your program. It needs to be "compiled", into machine language so that your CPU can actually execute the program as per the instructions given. The compiler compiles the source codes into final executable programs.

First C Program

Before starting the abcd of C language, you need to learn how to write, compile and run the first c program.

To write the first c program, open the C console and write the following code:

```
#include <stdio.h>
#include <conio.h>
void main(){
printf("Hello C Language");

getch();
}
```

Explanation:

- **#include <stdio.h>** includes the standard input output library functions. The printf() function is defined in stdio.h .
- **#include <conio.h>** includes the console input output library functions. The getch() function is defined in conio.h file.
- **void main()** function is the entry point of every program in c language. The void keyword specifies that it returns no value.
- **clrscr()** function to clear the screen.
- **printf()** function is used to print data on the console.
- **getch()** function asks for a single character. Until you press any key, it hold the screen.

Comments in C

Comments in C language are used to provide information about lines of code. It is widely used for documenting code. There are 2 types of comments in C language.

- Single Line Comments
- Multi Line Comments

Single Line Comments

Single line comments are represented by double slash //. Let's see an example of single line comment in C.

Multi Line Comments

Multi line comments are represented by slash asterisk /* ... */. It can occupy many lines of code but it can't be nested.

Escape Sequence in C

An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character.

Escape Sequence	Meaning
\a	Alarm or Beep
\b	Backspace
\t	Tab (Horizontal)
\v	Vertical Tab
\\	Backslash
\'	Single Quote
\"	Double Quote
\?	Question Mark
\0	Null

C Programming Tutorial

Identifiers

A C identifier is a name used to identify a variable, function, or any other user-defined item. An identifier starts with a letter A to Z, a to z, or an underscore '_' followed by zero or more letters, underscores, and digits (0 to 9).

C does not allow punctuation characters such as @, \$, and % within identifiers. C is a case-sensitive programming language. Thus, Man and man are two different identifiers in C.

Data Types

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
int	2 bytes	-32,768 to 32,767
long	4 bytes	-2,147,483,648 to 2,147,483,647
float	4 byte	1.2E-38 to 3.4E+38 (6 decimal places)

Defining Constants:

There are two simple ways in C to define constants –

1. Using #define preprocessor.
2. Using const keyword.

Example:

- #define LENGTH 10
- #define WIDTH 5
- #define NEWLINE '\n'
- const int LENGTH = 10;
- const int WIDTH = 5;
- const char NEWLINE = '\n';

Exercise:

- 1 C Program to Calculate Circumference of Circle
- 2 C Program to Calculate Area of Scalene Triangle
- 3 C Program to Calculate Area of Equilateral Triangle
- 4 C Program to Calculate Area of Right angle Triangle
- 5 C Program to Calculate Area of Circle
- 6 C Program to Calculate Area of Rectangle
- 7 C Program to Calculate Area of Square
- 8 Calculate sum of 5 subjects and Find percentage
- 9 Convert temperature from degree centigrade to Fahrenheit
- 10 Find the simple interest
- 11 Find sum of two numbers
- 12 Solve Quadratic Equation
- 13 Calculate sum of 5 subjects and find percentage
- 14 Swap 2 numbers using 3rd variable.
- 15 Swap 2 numbers without using 3rd variable.

C Programming Tutorial

C Operators

An operator is simply a symbol that is used to perform operations.

There are following types of operators to perform different types of operations in C language.

- Arithmetic Operators
- Relational Operators
- Shift Operators
- Logical Operators
- Bitwise Operators
- Ternary or Conditional Operators
- Assignment Operator
- Misc Operator

a) Arithmetic Operators

The following table shows all the arithmetic operators supported by the C language. Assume variable **A** holds 10 and variable **B** holds 20 then –

Operator	Description	Example
+	Adds two operands.	$A + B = 30$
-	Subtracts second operand from the first.	$A - B = -10$
*	Multiplies both operands.	$A * B = 200$
/	Divides numerator by de-numerator.	$B / A = 2$
%	Modulus Operator and remainder of after an integer division.	$B \% A = 0$
++	Increment operator increases the integer value by one.	$A++ = 11$
--	Decrement operator decreases the integer value by one.	$A-- = 9$

b) Relational Operators

The following table shows all the relational operators supported by C. Assume variable **A** holds 10 and variable **B** holds 20 then –

Operator	Description	Example
>	Checks if the value of left operand is greater than the value of right operand.	$(A > B)$
<	Checks if the value of left operand is less than the value of right operand.	$(A < B)$
>=	Checks if the value of left operand is greater than or equal to the right operand.	$(A >= B)$
<=	Checks if the value of left operand is less than or equal to the right operand.	$(A <= B)$

C Programming Tutorial

c) Comparison Operators

Following table shows all the logical operators supported by C language. Assume variable **A** holds 1 and variable **B** holds 0, then –

Operator	Description	Example
==	Checks if the values of two operands are equal or not.	(A == B)
!=	Checks if the values of two operands are equal or not.	(A != B)

d) Logical Operators

Following table shows all the logical operators supported by C language. Assume variable **A** holds 1 and variable **B** holds 0, then –

Operator	Description	Example
&&	Called Logical AND operator. If both conditions are true.	(A>B && B>C)
	Called Logical OR Operator. If any of the two conditions is true.	(A>B B>C)
!	Called Logical NOT Operator. It is used to reverse the logical state	(A !=B)

e) Assignment Operators

The following table lists the assignment operators supported by the C language –

Operator	Description	Example
=	Simple assignment operator. Assigns values from right side operands to left side operand	C = A + B will assign the value of A + B to C
+=	Add AND assignment operator. It adds the right operand to the left operand and assign the result to the left operand.	C += A is equivalent to C = C + A
-=	Subtract AND assignment operator. It subtracts the right operand from the left operand and assigns the result to the left operand.	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator. It multiplies the right operand with	C *= A is equivalent to C =

C Programming Tutorial

	the left operand and assigns the result to the left operand.	$C * A$
<code>/=</code>	Divide AND assignment operator. It divides the left operand with the right operand and assigns the result to the left operand.	$C /= A$ is equivalent to $C = C / A$
<code>%=</code>	Modulus AND assignment operator. It takes modulus using two operands and assigns the result to the left operand.	$C \% = A$ is equivalent to $C = C \% A$

f) Increment/Decrement Operators

Operator	Description	Example
<code>++</code>	Increment operator increases the integer value by one.	$A++ = 11$
<code>--</code>	Decrement operator decreases the integer value by one.	$A-- = 9$

g) The `? :` Operator (or Conditional operator)

We have a Conditional Operator which replace if...else statements. It has the following general form
`Exp1 ? Exp2 : Exp3;`

C Math

C Programming allows us to perform mathematical operations through the functions defined in `<math.h>` header file. The `<math.h>` header file contains various methods for performing mathematical operations such as `sqrt()`, `pow()`, `ceil()`, `floor()` etc.

No.	Function	Description
1)	<code>ceil(number)</code>	rounds up the given number. It returns the integer value which is greater than or equal to given number.
2)	<code>floor(number)</code>	rounds down the given number. It returns the integer value which is less than or equal to given number.
3)	<code>sqrt(number)</code>	returns the square root of given number.
4)	<code>pow(base, exponent)</code>	returns the power of given number.
5)	<code>abs(number)</code>	returns the absolute value of given number.

C Programming Tutorial

Example:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main(){
clrscr();
printf("\n%f",ceil(3.6));
printf("\n%f",ceil(3.3));
printf("\n%f",floor(3.6));
printf("\n%f",floor(3.2));
printf("\n%f",sqrt(16));
printf("\n%f",sqrt(7));
printf("\n%f",pow(2,4));
printf("\n%f",pow(3,3));
printf("\n%d",abs(-12));
getch();
}
```

Output:

```
4.000000
4.000000
3.000000
3.000000
4.000000
2.645751
16.000000
27.000000
12
```

if else Statement in C

The if statement in C language is used to perform operation on the basis of condition. By using if-else statement, you can perform operation either condition is true or false.

There are many ways to use if statement in C language:

- If statement
- If-else statement
- If else-if ladder
- Nested if

If Statement:

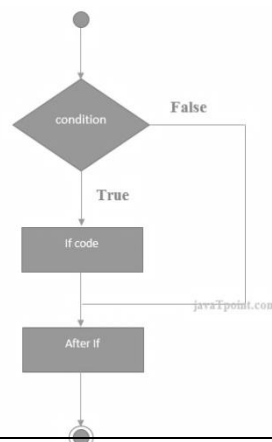
The single if statement in C language is used to execute the code if condition is true. The syntax of if statement is given below:

```
if(expression){
    //code to be executed
}
```

e.g;

```
#include<stdio.h>
#include<conio.h>
void main(){
    int number=0;
    clrscr();
    printf("enter a number:");
```

Flowchart of if statement in C



C Programming Tutorial

```
scanf("%d",&number);
if(number%2==0){
    printf("%d is even number",number);
}
getch();
}
```

o/p: enter a number:4
4 is even number

If-else Statement:

The if-else statement in C language is used to execute the code if condition is true or false. The syntax of if-else statement is given below:

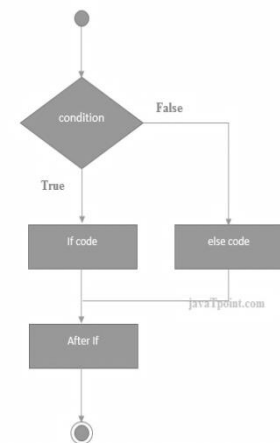
```
if(expression){
    //code to be executed if condition is true
}else{
    //code to be executed if condition is false
}
```

e.g: Let's see the simple example of even and odd number using if-else statement

```
#include<stdio.h>
#include<conio.h>
void main(){
    int number=0;
    clrscr();
    printf("enter a number:");
    scanf("%d",&number);
    if(number%2==0){
        printf("%d is even number",number);
    }
    else{
        printf("%d is odd number",number);
    }
    getch();
}
```

o/p: enter a number:5
5 is odd number

Flowchart of if-else statement in C



If else-if ladder Statement:

The if else-if statement is used to execute one code from multiple conditions. The syntax of if else-if statement is given below:

```
if(condition1){
    //code to be executed if condition1 is true
}else if(condition2){
    //code to be executed if condition2 is true
}
else if(condition3){
    //code to be executed if condition3 is true
}
...
else{
    //code to be executed if all the conditions are false
}
```


C Programming Tutorial

e.g:

```
#include<stdio.h>
#include<conio.h>
void main(){
int number=0;
clrscr();
printf("enter a number:");
scanf("%d",&number);
if(number==10){
printf("number is equals to 10");
}
else if(number==50){
printf("number is equal to 50");
}
else if(number==100){
printf("number is equal to 100");
}
else{
printf("number is not equal to 10, 50 or 100");
}
getch();
}
```

Exercise:

- 1 Find greatest in 2 numbers
- 2 Find greatest in 3 numbers
- 3 Calculate total of 5 subjects and find percentage with grade
- 4 Update the salary of an employee
 - salary less than 10,000, give 5 % increment
 - salary between 10,000 and 20,000, give 10 % increment
 - salary between 20,000 and 50,000, give 15 % increment
 - salary more than 50,000, give 20 % increment
- 5 Calculate the purchase amount after deduction criteria on printed price
 - printed price between 500 and 1000 , allow 5% discount
 - printed price between 1000 and 5000, allow 10% discount
 - printed price between 5000 and 10000, allow 15% discount
 - printed price more than 1000, allow 20% discount

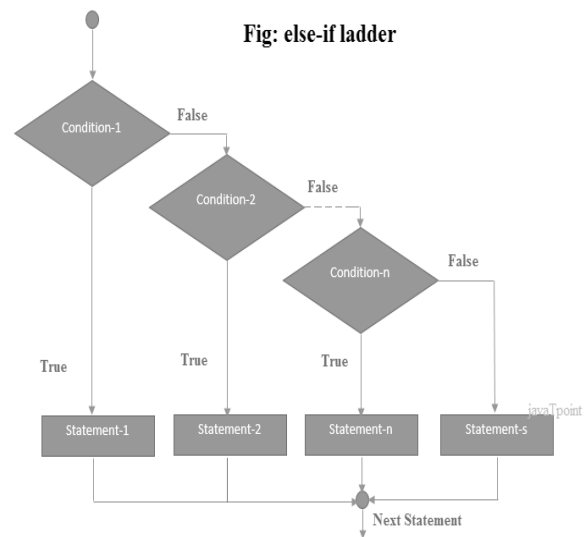
Switch Statement:

The switch statement in C language is used to execute the code from multiple conditions. It is like if else-if ladder statement.

The syntax of switch statement in c language is given below:

```
switch(expression){
case value1:
//code to be executed;
break; //optional
case value2:
//code to be executed;
break; //optional
.....
}
```

Fig: else-if ladder



o/p: enter a number:4
number is not equal to 10, 50 or 100

C Programming Tutorial

default:

```
// code to be executed if all cases are not matched;  
}
```

Rules for switch statement in C language:

- 1) The switch expression must be of integer or character type.
- 2) The case value must be integer or character constant.
- 3) The case value can be used only inside the switch statement.
- 4) The break statement in switch case is not must. It is optional. If there is no break statement found in switch case, all the cases will be executed after matching the case value. It is known as fall through state of C switch statement.

```
void main(){  
int number=0;  
clrscr();  
printf("enter a number:");  
scanf("%d",&number);  
switch(number){  
case 10:  
printf("number is equals to 10");  
break;  
case 50:  
printf("number is equal to 50");  
break;  
case 100:  
printf("number is equal to 100");  
break;  
default:  
printf("number is not equal to 10, 50 or 100");  
}  
getch();  
}
```

Exercise:

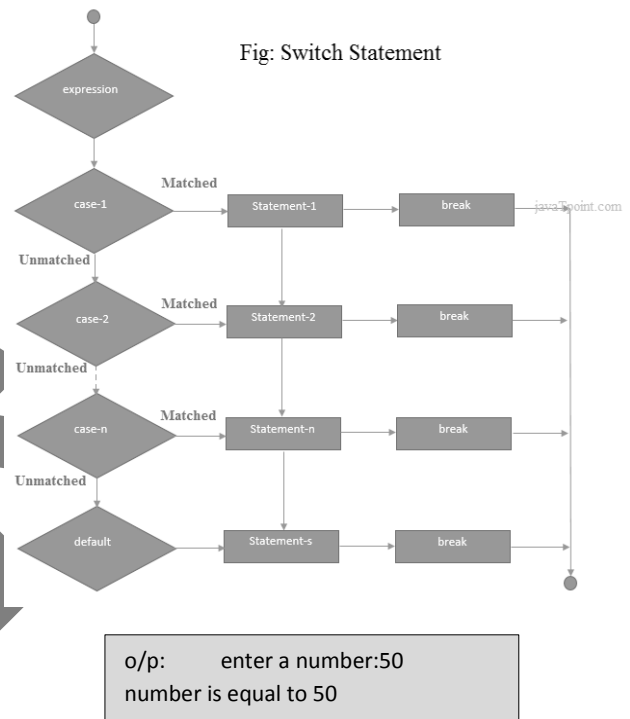
- 1 Switch on grade to display message
- 2 Switch on choice to perform the following operations
- 1 for addition
- 2 for subtraction
- 3 for multiplication
- 4 for division
- 3 Switch on choice to perform the following operations
- a for area of circle
- b for area of rectangle
- c for area of square
- d for area of cylinder

C Loops

The loops in C language are used to execute a block of code or a part of the program several times. In other words, it iterates a code or group of code many times.

Advantage of loops in C:

- 1) It saves code.
- 2) It helps to traverse the elements of array (which is covered in next pages).



C Programming Tutorial

Types of C Loops:

There are three types of loops in C language that is given below:

1. While
2. Do while
3. for

1. while loop in C:

It iterates the code until condition is false. Here, condition is given before the code. So code may be executed 0 or more times.

It is better if number of iteration is not known by the user.

The **syntax** of while loop in c language is given below:

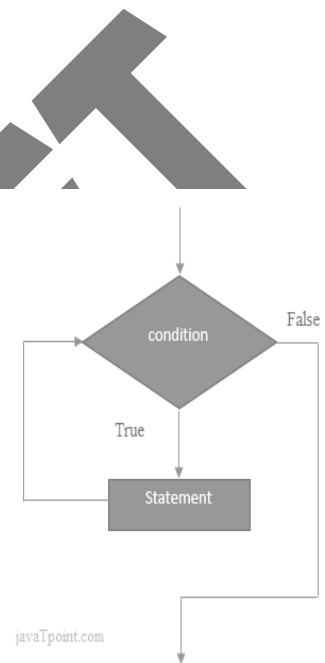
```
while(condition){  
    //code to be executed  
}
```

Flowchart and Example of while loop in C:

E.g; Program to print table for the given number using while loop in C:

```
#include <stdio.h>  
#include <conio.h>  
void main(){  
    int i=1,number=0;  
    clrscr();  
    printf("Enter a number: ");  
    scanf("%d",&number);  
    while(i<=10){  
        printf("%d ",(number*i));  
        i++;  
    }  
    getch();  
}
```

o/p: Enter a number: 5
5 10 15 20 25 30 35 40 45 50



2. do-while loop in C:

It iterates the code until condition is false. Here, condition is given after the code. So at least once, code is executed whether condition is true or false.

It is better if you have to execute the code at least once.

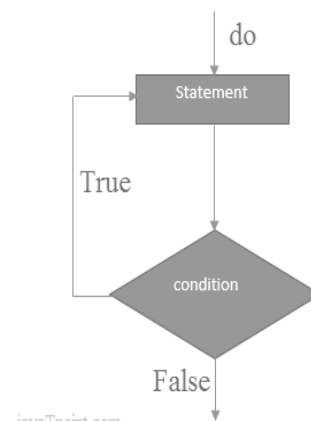
The **syntax** of do-while loop in c language is given below:

```
do{  
    //code to be executed  
}while(condition);
```

Flowchart and Example of do while loop in C:

E.g; Program to print table for the given number using do while loop:

```
#include <stdio.h>  
#include <conio.h>  
void main(){  
    int i=1,number=0;  
    clrscr();  
    printf("Enter a number: ");  
    scanf("%d",&number);  
    do{  
        printf("%d ",(number*i));  
        i++;  
    }while(i<=10);  
    getch();  
}
```



C Programming Tutorial

```
i++;  
}while(i<=10);  
getch();  
}
```

o/p: Enter a number: 5
5 10 15 20 25 30 35 40 45 50

3. for loop in C:

Like while, it iterates the code until condition is false. Here, initialization, condition and increment/decrement is given before the code. So code may be executed 0 or more times.

It is good if number of iteration is known by the user.

The **syntax** of for loop in c language is given below:

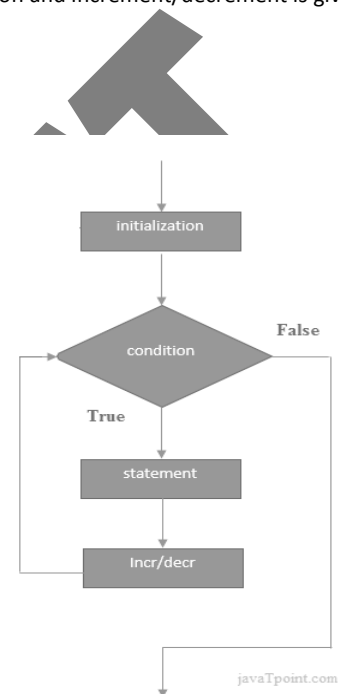
```
for(initialization;condition;incr/decr){  
//code to be executed  
}
```

Flowchart and Example of for loop in C:

E.g: Print table for the given number using C for loop:

```
#include <stdio.h>  
#include <conio.h>  
void main(){  
int i=1,number=0;  
clrscr();  
printf("Enter a number: ");  
scanf("%d",&number);  
for(i=1;i<=10;i++){  
printf("%d ",(number*i));  
}  
getch();  
}
```

o/p: Enter a number: 5
5 10 15 20 25 30 35 40 45 50



Exercise:

- 1 Print First 10 Natural Numbers
- 2 Find sum of N numbers
- 3 Write a program that prints on the screen all the even numbers up to 10.
- 4 Find Factorial of Number
- 5 Print table of n and square of n using pow()
- 6 Reverse a given number !
- 7 Generate the Fibonacci Series starting from any two numbers
- 8 Check Whether Given Number is Palindrome or Not ????
- 9 Check Whether Number is Prime or not
- 10 Check for Armstrong Number in C
- 11 Check Whether Number is Perfect Or Not
- 12 Print All ASCII Value Table in C Programming
- 13 C program to count number of digits in a given number
- 14 C program to add reversed number with Original Number

C Programming Tutorial

C Array

Array in C language is a collection or group of elements (data) of same type.

Advantage of C Array:

- 1) Code Optimization: Less code to access the data.
- 2) Easy to traverse data: By using the for loop, we can retrieve the elements of an array easily.
- 3) Easy to sort data: To sort the elements of array, we need a few lines of code only.
- 4) Random Access: We can access any element randomly using the array.

Disadvantage of C Array:

- 1) Fixed Size: Whatever size, we define at the time of declaration of array, we can't exceed the limit.

Declaration of C Array:

We can declare an array in the C language in the following way.

Syntax: `data_type array_name[array_size];`

E.g: `int marks[5];`

Initialization of C Array:

A simple way to initialize array is by index. Notice that array index starts from 0 and ends with [SIZE - 1].

```
marks[0]=80;           //initialization of array
marks[1]=60;
marks[2]=70;
marks[3]=85;
marks[4]=75;
```

C array example:

```
#include <stdio.h>
#include <conio.h>
void main(){
    int i=0;
    int marks[5]; //declaration of array
    clrscr();
    marks[0]=80; //initialization of array
    marks[1]=60;
    marks[2]=70;
    marks[3]=85;
    marks[4]=75;
    //traversal of array
    for(i=0; i<5; i++){
        printf("%d \n", marks[i]);
    } //end of for loop
    getch();
}
```

80	60	70	85	75
marks[0]	marks[1]	marks[2]	marks[3]	marks[4]

Initialization of Array

C Array: Declaration with Initialization:

We can initialize the C array at the time of declaration.

```
int marks[5]={20,30,40,50,60};
```

In such case, there is no requirement to define size. So it can also be written as the following code.

```
int marks[]={20,30,40,50,60};
```

C Programming Tutorial

Let's see the full program to declare and initialize the array in C.

```
#include <stdio.h>
#include <conio.h>
void main(){
    int i=0;
    int marks[5]={20,30,40,50,60}; //declaration and initialization of array
    clrscr();
    //traversal of array
    for(i=0;i<5;i++){
        printf("%d ",marks[i]);
    }
    getch();
}
```

o/p: 20 30 40 50 60

C Array using scanf:

e.g:

```
#include <stdio.h>
#include <conio.h>
void main(){
    int i=0;
    int marks[5]; //declaration of array
    clrscr();
    //input of array
    for(i=0;i<5;i++){
        scanf("%d",&marks[i]);
    }
    //traversal of array
    for(i=0;i<5;i++){
        printf("%d ",marks[i]);
    }
    getch();
}
```

Exercise:

- 1 C Program to implement the array
- 2 C Program to reversing an Array Elements
- 3 C Program to calculate Addition of All Elements in Array
- 4 C program to find Smallest Element in Array
- 5 C Program to find Largest Element in Array
- 6 C program to find 2nd largest and smallest element in array
- 7 Merging of Two arrays in C Programming
- 8 Searching element in array
- 9 Replace the search element with new element in array
- 10 copy all elements of an array into another array in C
- 11 C Program for deletion of an element from the specified location from Array
- 12 C program for sorting the array.
- 13 C Program to delete duplicate elements in an array

C Programming Tutorial

Two Dimensional Array in C

The two dimensional array in C language is represented in the form of rows and columns, also known as matrix. It is also known as array of arrays or list of arrays.

The two dimensional, three dimensional or other dimensional arrays are also known as multidimensional arrays.

Declaration of two dimensional Array in C:

data_type array_name[size1][size2];

A simple example to declare two dimensional array :

int arr[4][3];

Here, 4 is the row number and 3 is the column number.

Initialization of 2D Array in C:

A way to initialize the two dimensional array at the time of declaration is given below.

```
int arr[4][3]={1,2,3},{2,3,4},{3,4,5},{4,5,6}};
```

E.g:

```
void main(){
int i=0,j=0;
int arr[4][3]={1,2,3},{2,3,4},{3,4,5},{4,5,6}};
clrscr();
//traversing 2D array
for(i=0;i<4;i++){
for(j=0;j<3;j++){
printf("arr[%d] [%d] = %d \n",i,j,arr[i][j]);
} //end of j
} //end of i
getch();
}
```

O/P:

```
arr[0][0] = 1
arr[0][1] = 2
arr[0][2] = 3
arr[1][0] = 2
arr[1][1] = 3
arr[1][2] = 4
arr[2][0] = 3
arr[2][1] = 4
arr[2][2] = 5
arr[3][0] = 4
arr[3][1] = 5
arr[3][2] = 6
```

2D Array: using scanf():

E.g:

```
void main(){
int i=0,j=0;
int arr[4][3];
clrscr();
//Input of 2D array
for(i=0;i<4;i++){
for(j=0;j<3;j++){
scanf("%d",&arr[i][j]);
} //end of j
} //end of i
//traversing 2D array
for(i=0;i<4;i++){
for(j=0;j<3;j++){
printf("arr[%d] [%d] = %d \n",i,j,arr[i][j]);
} //end of j
} //end of i
getch();
}
```

C Programming Tutorial

Exercise:

- 1 C Program to implement 2D Array
- 2 C Program to Print Square of Each Element of 2D Array Matrix
- 3 Program to find Transpose of Given Square Matrix
- 4 C Program to find addition of two matrices
- 5 C Program to evaluate Subtraction of two matrices
- 6 Addition of Diagonal Elements in Matrix
- 7 Addition of All Elements in Matrix
- 8 Accessing 2-D Array Elements In C Programming
- 9 C Program to Multiply Two 3 X 3 Matrices
- 10 C program to find Smallest Element in 2D Array
- 11 C Program to find Largest Element in 2D Array
- 12 C Program to Check whether Matrix is Magic Square or Not ?

Functions in C

The function in C language is also known as procedure or subroutine in other programming languages.

To perform any task, we can create function. A function can be called many times. It provides modularity and code reusability.

Types of Functions:

There are two types of functions in C programming:

1. Library Functions: are the functions which are declared in the C header files such as scanf(), printf(), gets(), puts(), clrscr(), getch() etc.
2. User-defined functions: are the functions which are created by the C programmer, so that he/she can use it many times. It reduces complexity of a big program and optimizes the code.

Declaration of a function:

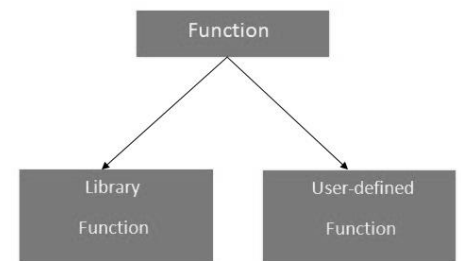
The syntax of creating function in c language is given below:

```
return_type function_name(data_type parameter...)
{
//code to be executed
}
```

Return Value means a C function may or may not return a value from the function.

If you don't have to return any value from the function, use void for the return type.

A c function may have 0 or more **parameters**. You can have any type of parameter in C program such as int, float, char etc. The parameters are also known as formal arguments.



1) Function with no return and no parameter:

E.g:

```
#include <stdio.h>
#include <conio.h>
//defining function
void hello(){
printf("hello c programming");
}
void main(){
clrscr();
hello();
hello();
hello();
getch();
}
```

//calling a function

Output: hello c programming
hello c programming
hello c programming

C Programming Tutorial

2) Function with no return and with parameter:

```
void cube(int n){
    return n*n*n;
}
void main(){
    clrscr();
    int a=3;
    cube(a);                //calling a function
    getch();
}
```

3) Function with return and no parameter:

```
int get(){
    int a=10;
    return a*a;
}
void main(){
    clrscr();
    int a;
    a=get();
    printf("Ans=%d",a);    //calling a function
    getch();
}
```

4) Function with return and with parameter:

```
int add(int a, int b){
    return a+b;
}
void main(){
    clrscr();
    int a=3,b=5,c;
    c=add(a,b);            //calling a function
    printf("Ans=%d",c);
    getch();
}
```

Exercise:

- 1 Create function for add two numbers using
 - a. without return type - without parameter
 - b. without return type - with parameter
 - c. with return type - without parameter
 - d. with return type - with parameter
- 2 Function to Calculate Circumference of Circle
- 3 Function to Calculate Area of Scalene Triangle
- 4 Function to Calculate Area of Equilateral Triangle
- 5 Function to Calculate Area of Right angle Triangle
- 6 Function to Calculate Area of Circle
- 7 Function to Calculate Area of Rectangle
- 8 Function to Calculate Area of Square
- 9 Write a program that ask for two numbers, compare them and show the maximum. Declare a function called max_two() that compares the numbers and returns the maximum.
- 10 Write a program that asks the user for an integer number and find the sum of all natural numbers upto that number.

C Programming Tutorial

C Strings:

String in C language is an array of characters that is terminated by \0 (null character).

Let's see a simple example to declare and print string. The '%s' is used to print string in c language.

```
void main ()
{
    char ch[11]={'h', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', '\0'};
    char ch2[11]="helloworld";
    printf("Char Array Value is: %s\n", ch);
    printf("String Literal Value is: %s\n", ch2);
}

o/p:      Char Array Value is: helloworld
          String Literal Value is: helloworld
```

gets() and puts() functions:

The gets() function reads string from user and puts() function prints the string. Both functions are defined in <stdio.h> header file.

Let's see a simple program to read and write string using gets() and puts() functions.

```
void main(){
    char name[50];
    clrscr();
    printf("Enter your name: ");
    gets(name); //reads string from user
    printf("Your name is: ");
    puts(name); //displays string
    getch();
}
```

Output:

Enter your name: Sonoo Jaiswal
Your name is: Sonoo Jaiswal

String Functions:

No.	Function	Description
1)	strlen(string_name)	returns the length of string name.
2)	strcpy(destination, source)	copies the contents of source string to destination string.
3)	strcat(first_string, second_string)	concatenates or joins first string with second string. The result of the string is stored in first string.
4)	strcmp(first_string, second_string)	compares the 1st string with 2nd string. If both strings are same, it returns 0.
5)	strrev(string)	returns reverse string.
6)	strlwr(string)	returns string characters in lowercase.
7)	strupr(string)	returns string characters in uppercase.
8)	strstr()	It represents the full string from where substring will be searched.

Exercise:

- 1 C Program to input String
- 2 C Program to Sort set of strings in alphabetical order using strcmp()
- 3 Program to convert String into Uppercase Using Library Function
- 4 Program to convert String into Lowercase Using Library Function
- 5 Program to copy one string into other with using library function strcpy()
- 6 Program to Concatenate Two Strings with using Library Function : Strcat()

C Programming Tutorial

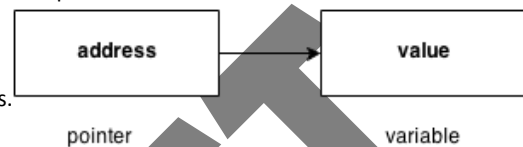
- 7 Find Length of String Using Library Function
- 8 C program to reverse the given String.
- 9 Check whether given string is palindrome or not
- 10 Search occurrence of Character in String :
- 11 Replace a character with the specific characters in string.
- 12 C Program to count number of words digits and vowels.

C Pointers

The pointer in C language is a variable, it is also known as locator or indicator that points to an address of a value.

Advantage of pointer :

- 1) Pointer reduces the code and improves the performance, it is used to retrieving strings, trees etc. and used with arrays, structures and functions.
- 2) We can return multiple values from function using pointer.
- 3) It makes you able to access any memory location in the computer's memory.



Declaring a pointer:

The pointer in c language can be declared using * (asterisk symbol).

```
int *a;           //pointer to int
char *c;          //pointer to char
```

Example:

```
void main(){
int number=50;
int *p;
clrscr();
p=&number;//stores the address of number variable
printf("Address of number variable is %x \n",&number);
printf("Address of p variable is %x \n",p);
printf("Value of p variable is %d \n",*p);
getch();
}
```

Output:

Address of number variable is fff4
Address of p variable is fff4
Value of p variable is 50

Example of Addition using Pointer:

```
void main(){
int number=50;
int *p;//pointer to int
p=&number;//stores the address of number variable
printf("Address of p variable is %u \n",p);
p=p+3; //adding 3 to pointer variable
printf("After adding 3: Address of p variable is %u \n",p);
}
```

Output:

Address of p variable is 3214864300
After adding 3: Address of p variable is 3214864312

Exercise:

- 1 Pointer Program: Declaring Pointer
- 2 Pointer Program: Address operator in C
- 3 C Program to Add Two Numbers Using Pointer!
- 4 C program to Print Value and Address of Variable of Pointer
- 5 C Program to Swap Two Numbers!
- 6 C Program to compute sum of the array elements using pointers !
- 7 C Program to read integers into an array and reversing them using pointers
- 8 C Program to Calculate Size of Pointer in C Programming
- 9 C Program to calculate Area of Circle using Pointer

C Programming Tutorial

Structure in C

Structure in c language is a user defined datatype that allows you to hold different type of elements.

Each element of a structure is called a member.

It works like a template in C++ and class in Java. You can have different type of elements in it.

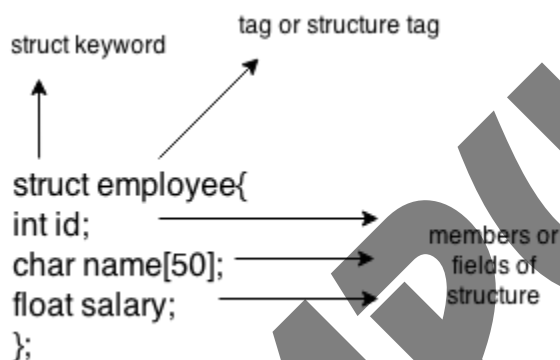
It is widely used to store student information, employee information, product information, book information etc.

Defining structure:

The struct keyword is used to define structure. Let's see the syntax to define structure in c.

```
struct structure_name
{
    data_type member1;
    data_type member2;
    .
    .
    data_type memberN;
} structure-object;
```

Let's see the example to define structure for employee in c.



The diagram illustrates the syntax of a C structure definition for an employee. It shows the code: `struct employee{ int id; char name[50]; float salary; };`. Annotations with arrows point to specific parts: 'struct keyword' points to 'struct', 'tag or structure tag' points to 'employee', and 'members or fields of structure' points to the list of variables 'id', 'name', and 'salary'.

Declaring structure variable:

We can declare variable for the structure, so that we can access the member of structure easily. There are two ways to declare structure variable:

1. By struct keyword within main() function
2. By declaring variable at the time of defining structure.

1st way:

Let's see the example to declare structure variable by struct keyword. It should be declared within the main function.

```
struct employee
{ int id;
  char name[50];
  float salary;
};
```

Now write given code inside the main() function.

```
struct employee e1, e2;
```

C Programming Tutorial

2nd way:

Let's see another way to declare variable at the time of defining structure.

```
struct employee
{
    int id;
    char name[50];
    float salary;
}e1,e2;
```

Let's see a simple example of structure in C language.

```
#include <stdio.h>
#include <string.h>
struct employee
{
    int id;
    char name[50];
}e1; //declaring e1 variable for structure
void main( )
{
    //store first employee information
    e1.id=101;
    strcpy(e1.name, "Sonoo Jaiswal");//copying string into char array
    //printing first employee information
    printf( "employee 1 id : %d\n", e1.id);
    printf( "employee 1 name : %s\n", e1.name);
    getch();
}
```

Array of Structures in C:

There can be array of structures in C programming to store many information of different data types. The array of structures is also known as collection of structures.

Let's see an example of structure with array that stores information of 5 students and prints it.

```
struct student{
int rollno;
char name[10];
};
void main(){
int i;
struct student st[5];
clrscr();
printf("Enter Records of 5 students");
for(i=0;i<5;i++){
printf("\nEnter Rollno:");
scanf("%d",&st[i].rollno);
printf("\nEnter Name:");
scanf("%s",&st[i].name);
}
printf("\nStudent Information List:");
for(i=0;i<5;i++){
printf("\nRollno:%d, Name:%s",st[i].rollno,st[i].name);
}
getch();
}
```

Output:

```
Enter Records of 5 students
Enter Rollno:1
Enter Name:Sonoo
Enter Rollno:2
Enter Name:Ratan
Enter Rollno:3
Enter Name:Vimal
Enter Rollno:4
Enter Name:James
Enter Rollno:5
Enter Name:Sarfraz
```

Student Information List:

```
Rollno:1, Name:Sonoo
Rollno:2, Name:Ratan
Rollno:3, Name:Vimal
Rollno:4, Name:James
Rollno:5, Name:Sarfraz
```

C Programming Tutorial

C Union

Like structure, Union in c language is a user defined datatype that is used to hold different type of elements. But it doesn't occupy sum of all members size. It occupies the memory of largest member only. It shares memory of largest member.

Structure	Union
<pre>struct Employee{ char x; // size 1 byte int y; //size 2 byte float z; //size 4 byte }e1; //size of e1 = 7 byte</pre>	<pre>union Employee{ char x; // size 1 byte int y; //size 2 byte float z; //size 4 byte }e1; //size of e1 = 4 byte</pre>
size of e1= 1 + 2 + 4 = 7	size of e1= 4 (maximum size of 1 element)

Advantage of union over structure:

It occupies less memory because it occupies the memory of largest member only.

Disadvantage of union over structure:

It can store data in one member only.

Exercise:

- 1 C Program to read and print name and other details of a students using Structure
- 2 C program to use array of structure & display the contents of structure elements
- 3 C program to use array of structure & display the contents of structure elements using Switch

File Handling in C

File Handling in c language is used to open, read, write, search or close file. It is used for permanent storage.

Advantage of File:

It will contain the data even after program exit. Normally we use variable or array to store data, but data is lost after program exit. Variables and arrays are non-permanent storage medium whereas file is permanent storage medium.

Functions for file handling:

There are many functions in C library to open, read, write, search and close file. A list of file functions are given below:

No.	Function	Description
1	fopen()	opens new or existing file
2	fprintf()	write data into file
3	fscanf()	reads data from file
4	putc()	writes a character into file
5	getc()	reads a character from file
6	fclose()	closes the file
7	remove()	remove the file
8	rename()	rename the file
9	putchar()	prints a character on output screen
10	putc()	writes a character into file

C Programming Tutorial

Example:

```
void main()
{
    FILE *fp;                //create file pointer
    char ch;
    fp = fopen("one.txt", "w"); //open the file
    printf("Enter data");
    while( (ch = getchar()) != '@') { //write data till @
        putc(ch,fp);
    }
    fclose(fp);              //close the file

    //Read the text file
    fp = fopen("one.txt", "r");
    while( (ch = getc(fp)) != EOF) //loop runs till end of the file i.e. EOF
        printf("%c",ch);
    fclose(fp);
}
```

Reading and Writing from File using fprintf() and fscanf() using structure:

```
struct emp
{
    char name[10];
    int age;
};
void main()
{
    struct emp e;
    FILE *p,*q;
    p = fopen("one.txt", "a");
    q = fopen("one.txt", "r");
    printf("Enter Name and Age");
    scanf("%s %d", e.name, &e.age);
    fprintf(p,"%s %d", e.name, e.age);
    fclose(p);
    do
    {
        fscanf(q,"%s %d", e.name, e.age);
        printf("%s %d", e.name, e.age);
    }
    while( !feof(q) );
    getch();
}
```

Exercise:

- 1 C Program to Write on Data File
- 2 C program to Read from Data File
- 3 C Program to write on more than one Data File
- 4 C program to Read From more than one Data File
- 5 C program to convert the file contents in Upper-case & Write Contents in a output file
- 6 C Program to Copy Text from One File to Other File
- 7 C program to merge two data files