Mo. :- 8804321194
telegram Id:- @vksarraf

C By Vikash Sarraf

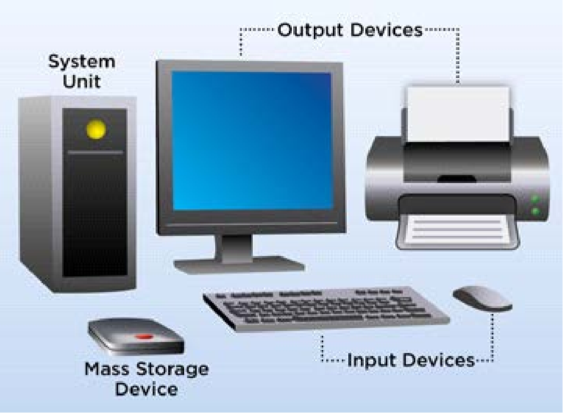
Introduction (I)

Computer

Any Calculating device is called Computer.

Definition - What Does Computer mean?

A computer is a machine or device that performs processes, calculations and operations based on instructions provided by a software or hardware program. It is designed to execute applications and provides a variety of solutions by combining integrated hardware and software components.

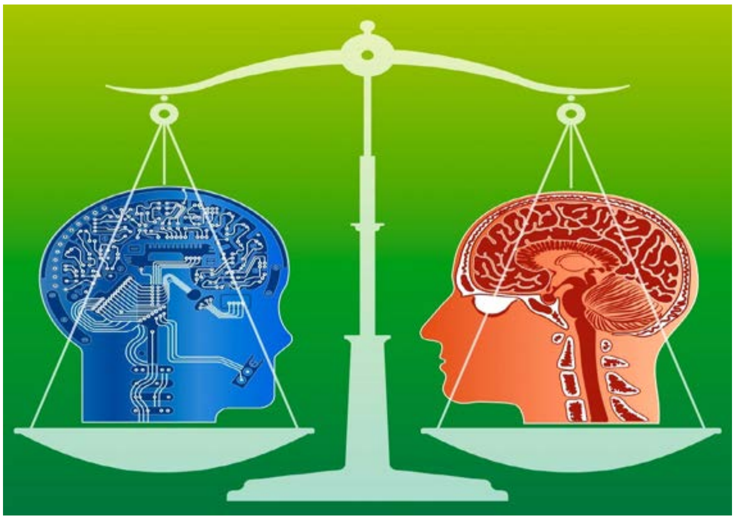


Full Form

C= Common
O= Oriented
M= Machine
P= Particularly
U= United and used under
T= Technical and
E= Educational
R= Research

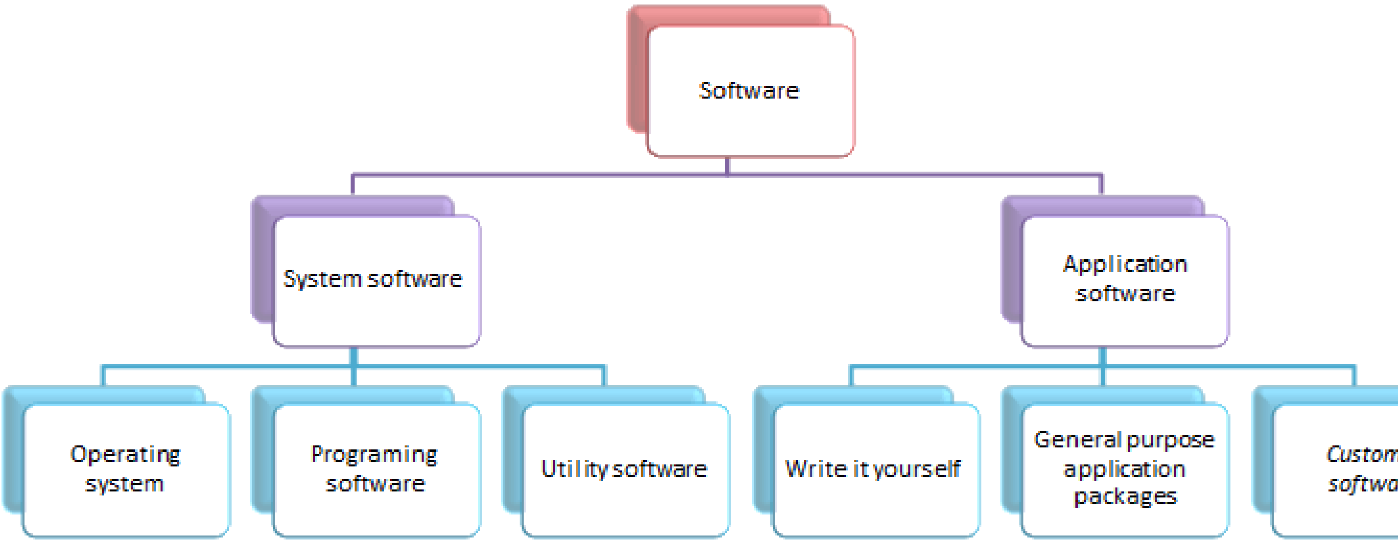
Brain Vs. Computer

Speed
Accuracy
Processing
Size and Complexity
Storage
Control Mechanism
Automation
Versatile
Diligency
Reliability



Software

Collection of programs is called Software.



Program

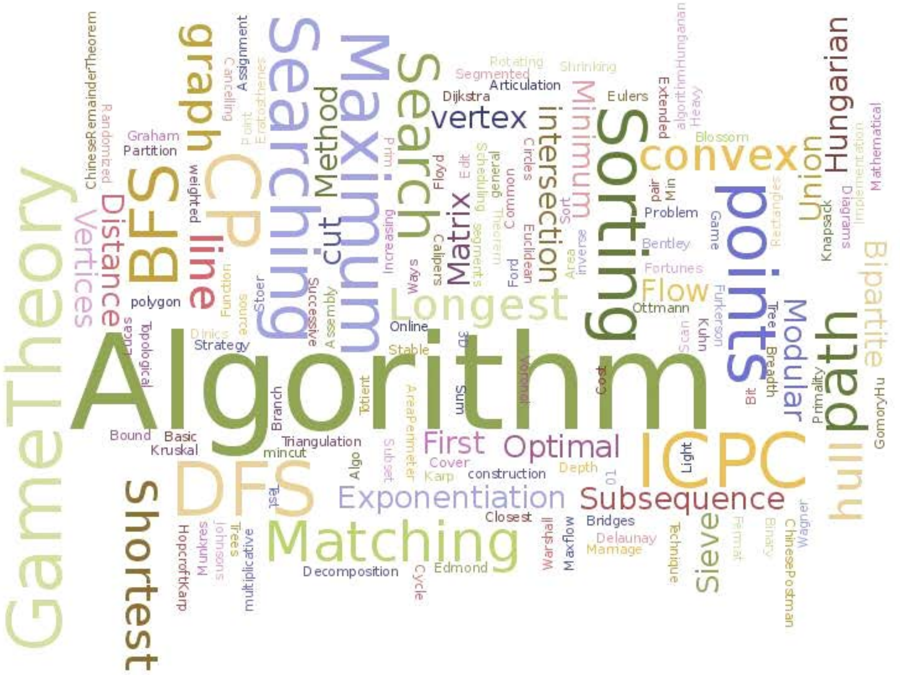
Collection of instructions is called program.

Programing

Solving the problem step by step is called is called Programming.

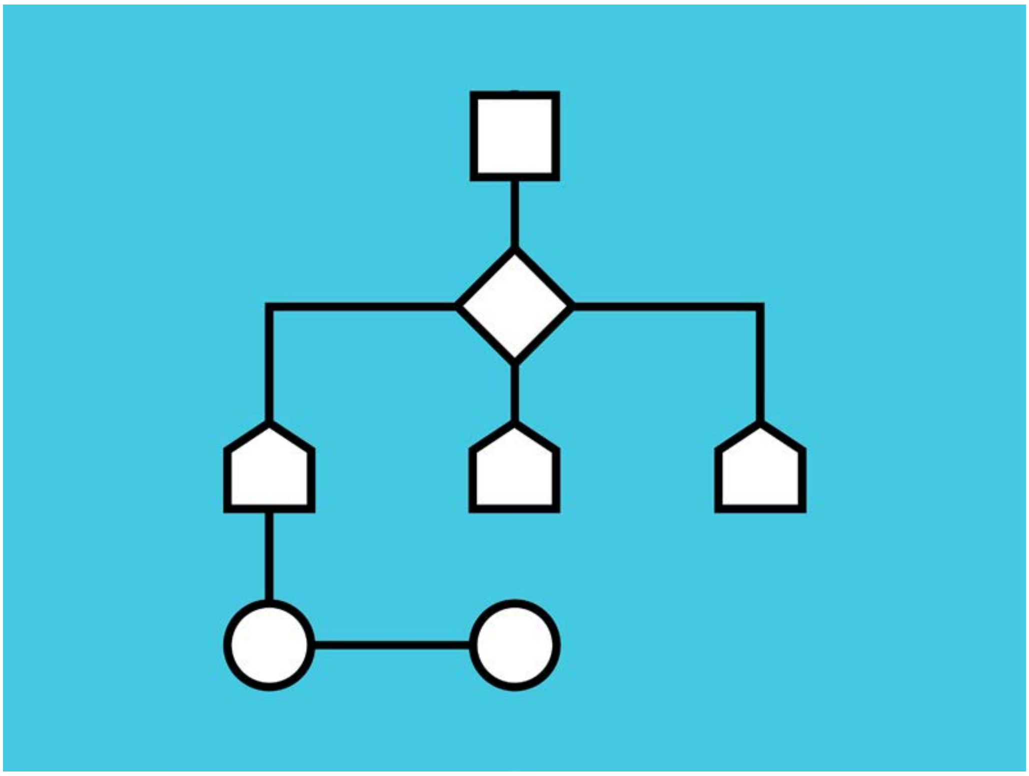
Algorithm

An algorithm (pronounced AL-go-rith-um) is a procedure or formula for solving a problem, based on conducting a sequence of specified actions. A computer program can be viewed as an elaborate algorithm. In mathematics and computer science, an algorithm usually means a small procedure that solves a recurrent problem.



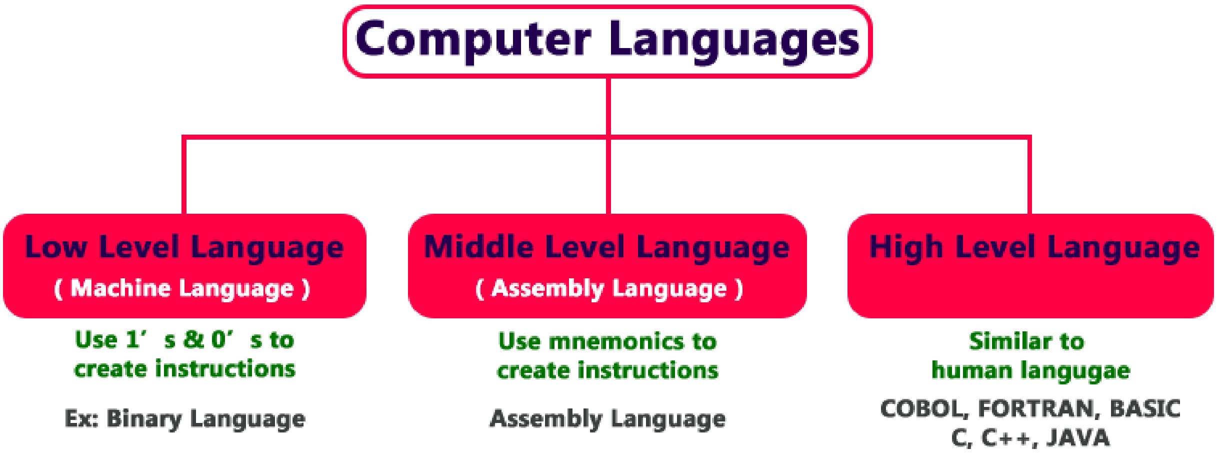
Flow Chart

A flowchart is a formalized graphic representation of a logic sequence, work or manufacturing process, organization chart, or similar formalized structure. The purpose of a flow chart is to provide people with a common language or reference point when dealing with a project or process.



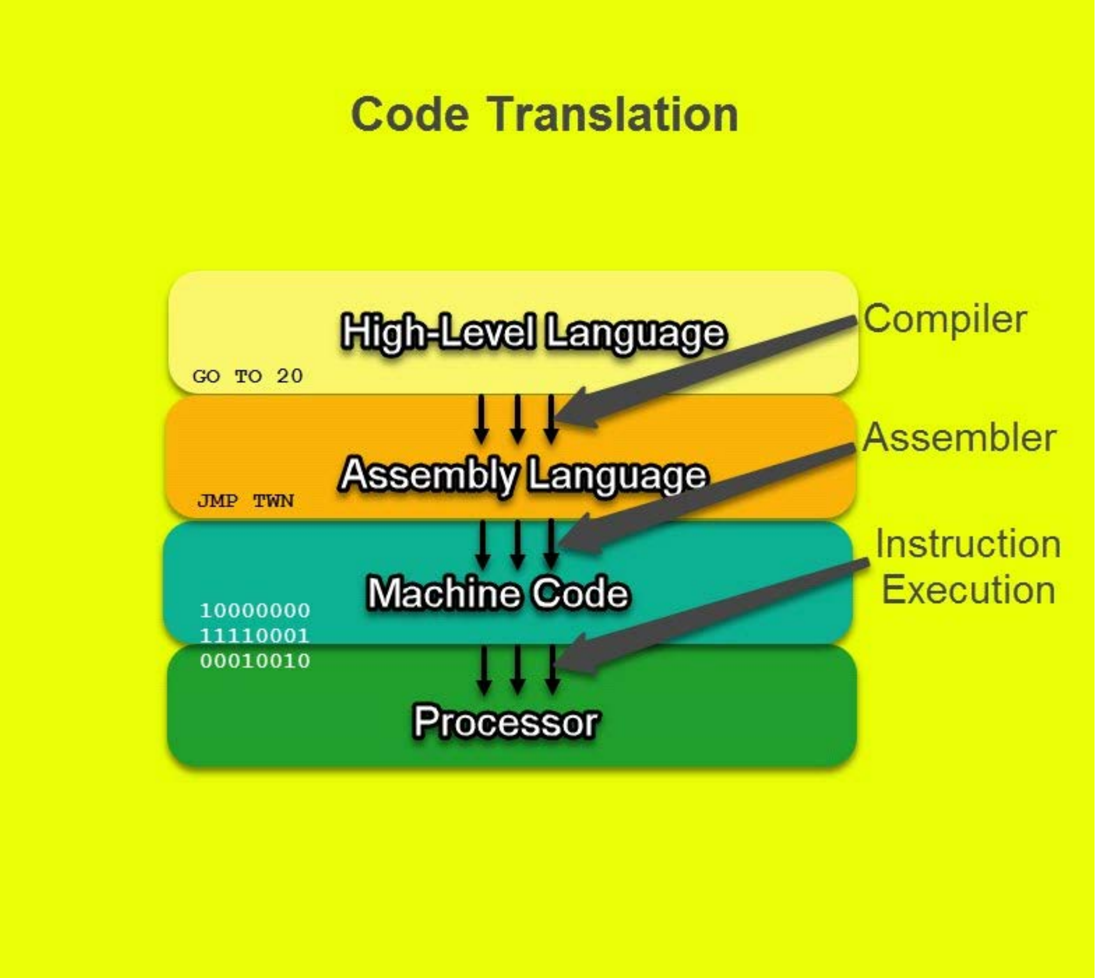
Programming Languages

There are two types of programming language.
1. Low level language (L.L.L)
2. High level language (H.L.L)



Translators

There are three types of translators
1. Assembler
2. Interpreter
3. Compiler



Assembler

It Converts the program written in Assembly language to Low Level Language.

Interpreter

It converts the program written in High Level Language into Low Level Language line by line if any error in any line then Execution process will terminate.

Compiler

It converts the program written in High Level Language into Low Level Language at one go. If there is any syntax error in program, program won’t be execute.

(II)Introduction of C

C is the functional programming language developed by Dennis Ritchie in 1970 at AT & T Bell lab USA.

But c was appeared in 1972

In 1982 a committee established by ANST to fix the standard of C Language.

In 1989 ANSI introduce ANSI C

C is derived from B language and B is derived from BCPL (Basic Combine Programming Language)

Application Areas of C Language

We can develop standalone application by using C language.

Desktop App

GUI App

Games

Operating System

Compilers

To develop New Language

Structure of C Program

```
Comments
Global Variables
main()
{
}
func1()
{
}
funct2()
{
}
```

(III) Reserve words

Reserve words

Reserve words are also called keywords. It is predefine words which is used for do specific task in program.
C has 32 Reserve words
All keywords in python are in lower case

Keywords in C Programming			
auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

(IV) Identifier

A name in program which, can be used for identification purpose is called an Identifier.
It can be function name, variable name.

Rules to define identifier in C

1. Identifiers can be combination of uppercase and lowercase letters, digits or an underscore (_)sign so We can't use special any symbols like !,#,@,%,$ etc in our Identifier.

2.An Identifier can not start with digit

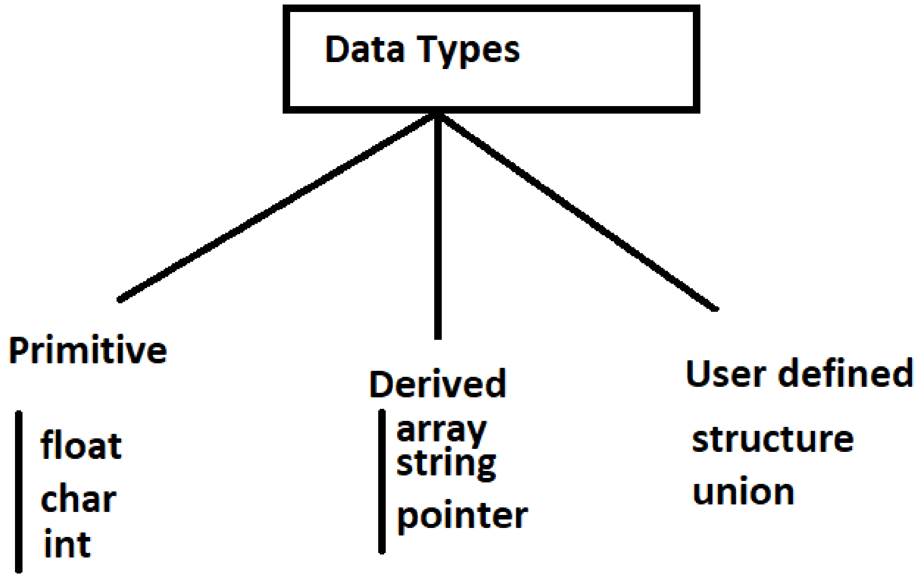
3.We can't use reserve words as identifier.

4.Identifier can be only 31 Character

5.C is case sensitive so
var1 & Var both are two different identifier (variable)

```
i.e. var    ------ valid
     var123   ------ valid
     Var121   ------ valid
     var_123  ------ valid
     _Var245  ------ valid
     2121Var245  ------ invalid
     if=20   invalid
     _Var&45  ------ invalid
     _Var$45  ------ invalid
```

(V) Data types



Integer data type

short memory 2 byte range

int memory 2/4 byte range

long memory 4 byte range

format specifier %d and %u

float data type

float memory 4 byte

double memory 8 byte

format specifier %f

char data type

char memory 1 byte

format specifier %c

(VI) Operators

Arithmetic

Increment/Decrement

Relational (< ,> ,<=,>=,== , !=)

Assignment (=)

Logical

Bitwise (& ,|,~, <<,>>)

Ternary

Arithmetic Operators

Operators are symbols used to perform certain operations
There are various operators in C

+ -----------> for addition
- -------------> for subtraction
* -----------> for multiplication
/ -----------> for division
% ----------------> for remainder (modulo)

Assignment (=)

Increment/Decrement

Relational (<,>,<=,>=,==,!=)

Logical (&&, || and !)

(VII) Input/output statements

Reading data from keyboard

scanf(“Control String”,address of var)

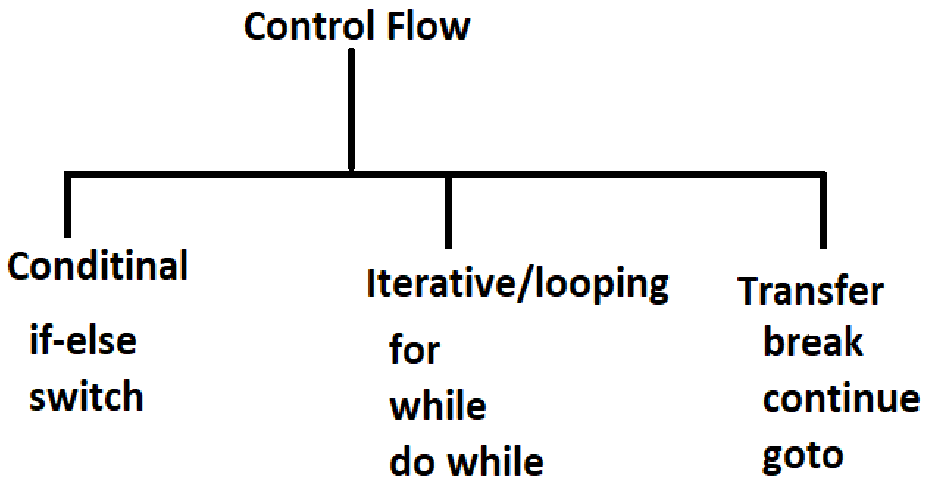
ie. scanf(“%d,%d”,&a,&b)

printing data to consol

printf(“Control String”, variables)

ie. printf(“%d,%d”,a,b)

(VIII) Control Statements



if

Syntax

```
If (condition)
{
 statements
 }
```

If-else

Syntax

```
    If (condition)
    {
     statements
     }
   else
    {
     statements
     }
```

```
/* wap to input a number & check number is 'negative or not */

#include<stdio.h>
#include<conio.h>
void main()
{
int n;
printf("Enter a number:");
scanf("%d",&n);
if (n<0)
printf("%d is negative number",n);
}
```

/* wap to input a number & check number is even or odd

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

/* wap to input Year & check year is leap or not

s

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
int year;
printf("Enter a year:");
scanf("%d",&year);
if ((year%4==0  && year%100!=0) || (year%400==0) )
printf("%d is leap",year);
else
printf("%d is not leap",year);
}

/* Wap to input two numbers & print greater number

#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2;
printf("Enter  numbers:\n");
scanf("%d%d",&n1,&n2);
if (n1>n2)
printf("%d is greater number",n1);
else
printf("%d is greater number",n2);
}
```

/* Wap to find the quotient & remainder

Nesting of if-else

Syntax

```
  If (condition)
  {
        If (condition)
           statements
          else
          statements
   }
  else
  {
          If (condition)
           statements
          else
          statements


   }
```

/* Wap to input three numbers & print greater number

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,n3;
printf("Enter  numbers:\n");
scanf("%d%d%d",&n1,&n2,&n3);
if (n1>n2 && n1>n3)
printf("%d is greater number",n1);
else
{
   if (n2>n3)
   printf("%d is greater number",n2);
   else
   printf("%d is greater number",n3);
}
}
```

Dangling Else problem

```
if
    if
        Statements
else
        statements
```

Itrative/looping

If we want to execute a group of statements multiple times then we should go for iterative statements.
C have three type of iterative statements

for loop
while loop
do while loop

<u>for loop</u>

Syntax

```
for(initialization;condition;updation)
{
        Statements
}
```

/* Wap to print Hello C world 10 times

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int i;
        for (i=0;i<=10;i++)
        {
        printf("I am Your c instructor\n");
        }
getch();
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int i;
        for (i=1;i<=100;i++)
        {
        printf("%d\n",i);
        }
getch();
}
```

/* Wap to print 100 to 1

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int i;
        for (i=100;i>1;i--)
        {
        printf("%d\t",i);
        }
getch();
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
        int i;
        for (i=100;i>1;i-=2)
        {
        printf("%d\t",i);
        }
getch();
}
```

/* W.A.P to input a number and print their table

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i;
printf("Enter  a numbers:\n");
scanf("%d",&n);
printf("Table of %d is\n",n);
for (i=1;i<=10;i++)
{
   printf("%d\n",n*i);
}

}
```

```
/*  W.A.P to input a number a print its factorial

#include<stdio.h>
#include<conio.h>
void main()
{
int n,fact=1,i;
printf("Enter  a numbers:\n");
scanf("%d",&n);
printf("Fatorial of %d is\n",n);
for (i=n;i>=1;i--)
{
   fact=i*fact;

}
printf("%d",fact);
}
```

```
/* Fibonacci Series

#include <stdio.h>
int main()
{
   int i, n, x = 0, y = 1, z;

   printf("Enter the number of terms: ");
   scanf("%d", &n);
```

```
    printf("Fibonacci Series: ");

    for (i = 1; i <= n; ++i)
    {
        printf("%d, ", x);
        z = x + y;
        x = y;
        y = z;
    }
    return 0;
}
```

while loop

if we want to execute a group of statements iteratively until some condition false,then we should use while loop

syntax

```
while (condition)
        statement
```

```
/* Wap to print Hello C world 10 times

#include<stdio.h>
#include<conio.h>
void main()
{
        int i=0;
        while (i<=10)
        {
        printf("Your name\n");
        i++ ;
        }
getch();
}
```

# W.A.P. to print 100 to 1 by using while loop.

/* W.A.P to input a number and print their table

```
#include<stdio.h>

#include<conio.h>

void main()

{

int n,i;

printf("Enter  a numbers:\n");

scanf("%d",&n);

printf("Table of %d is\n",n);

i=1;

while (i<=10)

{

   printf("%d\n",n*i);

   i++;

}
}
/*  W.A.P to input a number a print its factorial


#include<stdio.h>
#include<conio.h>
void main()
{
int n,fact=1;
printf("Enter  a numbers:\n");
scanf("%d",&n);
printf("Fatorial of %d is\n",n);
while (n>=1)
{
   fact=n*fact;
   n--;
}
printf("%d",fact);
}
/* Wap to input a number & print their digit sum
#include<stdio.h>
#include<conio.h>
void main()
```

```
{
int n,tot=0,rem;
printf("Enter number:");
scanf("%d",&n);
while (n>0)
{
rem=n%10;
tot=tot+rem;
n=n/10;
}
printf("The  sum=%d",tot);
}
```

do while loop

syntax

```
do
{
          statement
}
while (condition);
```

# Nested Loop

Pattern printing

```
/*
#
# #
# # #
# # # #
# # # # #
*/
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,j;
printf("Enter number of rows:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
   for (j=1;j<=i;j++)
   {
      printf("#");
   }
```

```
printf("\n");
}
}

/*
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
*/
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,j;
printf("Enter number of rows:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
   for (j=1;j<=i;j++)
   {
        printf("%d ",j);
    }
printf("\n");
}
}

/*
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
*/

#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,j;
printf("Enter number of rows:");
scanf("%d",&n);
for(i=1;i<=n;i++)
```

```
{
    for (j=1;j<=i;j++)
    {
        printf("%d ",i);
    }
printf("\n");
}
}

/*
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
*/
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,j,a=1;
printf("Enter number of rows:");
```

```
scanf("%d",&n);
for(i=1;i<=n;i++)
{
   for (j=1;j<=i;j++)
   {
      printf("%d ",a);
      a++;
   }
printf("\n");
}
}
```

Transfer Statements

break

continue

goto

/* Wap to print number is even or odd

```
#include<stdio.h>

#include<conio.h>

void main()

{

int n;

printf("Enter a number:");

scanf("%d",&n);

if (n%2==0)

goto even;

else

goto odd;

even:

printf("%d is even number",n);

odd:

printf("%d is odd number",n);

}
```

Switch

Switch is a multi-way conditional control statements which helps us to make choice among a number of alternative.

The syntax of switch statements is

```
switch (expression)
```

```
{
case constant1:
        statements
case constant2:
        statements
case constant3:
        statements
default:
        statements
}


#include <stdio.h>

int main() {

   char operator;
   float firstNumber,secondNumber;

   printf("Enter an operator (+, -, *, /): ");
   scanf("%c", &operator);
```

```
printf("Enter two operands: ");
scanf("%f %f",&firstNumber, &secondNumber);

switch(operator)
{
    case '+':
        printf("%f ", firstNumber+secondNumber);
        break;

    case '-':
        printf("%f", firstNumber-secondNumber);
        break;

    case '*':
        printf("%f", firstNumber*secondNumber);
        break;

    case '/':
        printf("%f", firstNumber-secondNumber);
        break;
    default:
        printf("Error! operator is not correct");
    }
}
```

Functions

If a group of statements is repeatedly required then it is not recommended to write these statements everytime seperately.We have to define these statements as a single unit and we can call that unit any number of times based on our requirement without rewriting.

This unit is nothing but function.

The main advantage of functions is code Reusability.

function is a group of statements.

it's the smallest part of the program.

It is not automatically executed.

There are two types of function in C

1. predefine
2. user defined function

1. Predefine:

The functions which are coming along with C software automatically, are called pre-defined functions

Eg:

printf()

scanf()

sizeof()

pow()

etc..

2. User Defined Functions:

The functions which are developed by programmer explicitly according to business requirements ,are called user defined functions.

1. Function definition
2. Function declaration
3. Function call

How to create Function in C

Syntax

```
Return_type func_name(parameter)
{
        Statements;
        return statements
}
```

Parameter

Parameters are input to the functions. It is also called arguments.

Explain formal/dummy & Actual Parameter .

Return Statement

```
#include<stdio.h>
#include<conio.h>
int sum(int a,int b);
void main()
{
int n1,n2,n3;
printf("Enter numbers:");
scanf("%d,%d",&n1,&n2);
n3=sum(n1,n2);
printf("The sum=%d",n3);
}
int sum(int a,int b)
{
    int c;
    c=a+b;
return c;
}
```

```
#include<stdio.h>
#include<conio.h>
void even_odd(int a);   /* declaration
void main()
```

```
{
int n;
printf("Enter a number:");
scanf("%d",&n);
even_odd(n);
}
void even_odd(int a)
{
if (a%2==0){

printf("%d even\n",a);


}
else{
printf("%dis odd\n",a);
}
}

#include<stdio.h>
void main()
{
   int row;
```

```
    printf("Enter number of rows");
    scanf("%d",&row);
    pattern(row);
}
void pattern(int r)
{
int i,j;
for (i=1;i<=r;i++)
    {
        for(j=1;j<=i;j++)
            {
     printf("$ ");
                }
    printf("\n");
    }
}
```

Types of function basis on parameter

```
main()
```

Arrays

1 d array   declaration & initialization

```
data_type array_name[size];

ie. int age[100];
    float sal[15];
    char nm[30];

int lakshya[5]={1,2,3,4,5 }
```

# WAP input values into array & print them

```
#include <stdio.h>
void main()
{
   int i =0,arr[5];

   printf("\nEnter elements : \n");

   for(i=0; i<5; i++)
```

```
    {
        printf("Enter arr[%d] : ",i);
        scanf("%d",&arr[i]);
    }
    printf("\nElements are : ");

    for(i=0; i<5; i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
}
```

# WAP to find even & odd number from given array

```
void main()
{
    int i =0,arr[5],even=0,odd=0;

    printf("\nEnter elements : \n");
```

```
   for(i=0; i<5; i++)
   {
      printf("Enter arr[%d] : ",i);
      scanf("%d",&arr[i]);
   }
   for(i=0; i<5; i++)
   {
      if (arr[i]%2==0)
      even++;
      else
      odd++;
   }
   printf("\neven number=%d and odd number=%d: ",even,odd);

   printf("\n");
}
```

# Wap to input two 1-d arrays & add them

```
#include <stdio.h>
```

```
void main()
{
   int i =0,a1[5],a2[5],c[5];

   printf("\nEnter first array elements : \n");

   for(i=0; i<5; i++)
   {
      printf("Enter a1[%d] : ",i);
      scanf("%d",&a1[i]);
   }
      printf("\nEnter second array elements : \n");

   for(i=0; i<5; i++)
   {
      printf("Enter a2[%d] : ",i);
      scanf("%d",&a2[i]);
   }



   for(i=0; i<5; i++)
```

```
    {
        c[i]=a1[i]+a2[i];
    }
    printf("\nThe sum of Elements are : ");
    for(i=0; i<5; i++)
    {
     printf("%d ",c[i]);
    }

    printf("\n");
}
```

# traverse of array in backward direction

```
void main()
{
```

```
int i ,j,arr[5],temp;

printf("\nEnter elements : \n");

for(i=0; i<5; i++)
{
    printf("Enter arr[%d] : ",i);
    scanf("%d",&arr[i]);
}
for(i=0,j=4; i<j; i++,j--)
{
   temp=arr[i];
   arr[i]=arr[j];
   arr[j]=temp;
}
printf("\nAfter Traversing of an array ");
for(i=0; i<5; i++)
{
    printf("%d ",arr[i]);
}
```

```
    printf("\n");
}
```

Passing 1-d array into a function

# WAP to find even and odd number from array

```
#include <stdio.h>
void readArray(int arr[], int size)
{
    int i =0;

    printf("\nEnter elements : \n");

    for(i=0; i<size; i++)
    {
        printf("Enter arr[%d] : ",i);
        scanf("%d",&arr[i]);
    }
}
void printArray(int arr[],int size)
{
    int i =0;

    printf("\nElements are : ");

    for(i=0; i<size; i++)
    {
        printf("\n\tarr[%d] : %d",i,arr[i]);
    }
    printf("\n");
}

int main()
{
    int arr[10];

    readArray(arr,10);
    printArray(arr,10);
    return 0;
}
```

2 d array   declaration & initialization

```
data_type array_name[row_size][col_size];

ie. int age[3][3]={{1,2,3},
                   {4,5,6},
                   {7,8,9}}
```

# Wap to input 2-d array & print all elements

```
#include <stdio.h>
```

```
void main()
{
   int i ,j,arr[3][3];

   printf("\nEnter elements : \n");

   for(i=0; i<3; i++)
   {
      for(j=0; j<3; j++)
      {
         scanf("%d",&arr[i][j]);
      }
   }
   printf("\nElements are : \n");

   for(i=0; i<3; i++)
   {
      for(j=0; j<3; j++)
      {
      printf("%d ",arr[i][j]);
      }
      printf("\n");
```

```
    }

}
```

# WAP to input two 2-d array nd Add them

```
#include <stdio.h>
void main()
{
    int i ,j,a[3][3],b[3][3],c[3][3];

    printf("\nEnter First Array elements : \n");

    for(i=0; i<3; i++)
    {
        for(j=0; j<3; j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    printf("\nEnter Second Array elements : \n");
```

```
for(i=0; i<3; i++)
{
   for(j=0; j<3; j++)
   {
      scanf("%d",&b[i][j]);
   }
}
for(i=0; i<3; i++)
{
   for(j=0; j<3; j++)
   {
      c[i][j]=a[i][j]+b[i][j];
   }
}
printf("\nArray Elements sum : \n");

for(i=0; i<3; i++)
{
   for(j=0; j<3; j++)
   {
   printf("%d ",c[i][j]);
   }
```

```
        printf("\n");
    }

}
```

Pointer

Pointers in C language is a variable that stores/points the address of another variable. A **Pointer in C** is used to allocate memory dynamically i.e. at run time

Find the address of variable

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1=10;
float n2=10.5;
printf("Address of n1=%p\n",&n1);
printf("Address of n1=%p",&n2);
}
```

Declaration of pointer

```
        data_type  *pname;
```

Dereference of a pointer variable

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1=10;
float n2=10.5;
int *p1=&n1;
float *p2=&n2;
printf("Value of p1= address of n1=%p\n",p1);
printf("Value of p2= address of n2=%p\n",p2);
printf("address of p1=%p\n",p1);
printf("address of p2=%p\n",p2);
printf("Value of n1=%d\n",*(&n1));
printf("Value of n2=%f\n",*(&n2));
}
```

```
#include <stdio.h>
void swap(int *x, int *y);

int main () {
   int a = 100;
   int b = 200;

   printf("Before swap, value of a : %d\n", a );
   printf("Before swap, value of b : %d\n", b );
   swap(&a, &b);

   printf("After swap, value of a : %d\n", a );
   printf("After swap, value of b : %d\n", b );

   return 0;
}
void swap(int *x, int *y) {

   int temp;
   temp = *x;
   *x = *y;
   *y = temp;

}
```

Strings

```
#include <stdio.h>

int main()
{
   char array[100];

   printf("Enter a string\n");
   scanf("%s", array);

   printf("Your string: %s\n", array);
   return 0;
}

#include <stdio.h>
```

```
int main()
{
  char z[100];

  printf("Enter a string\n");
  gets(z);

  printf("The string: %s\n", z);
  return 0;
}
```

Function	Work of Function
strlen()	Calculates the length of string
strcpy()	Copies a string to another string
strcat()	Concatenates(joins) two strings
strcmp()	Compares two string
strlwr()	Converts string to lowercase
strupr()	Converts string to uppercase

WAP to find the length of a string

```
#include <stdio.h>
#include <string.h>

int main()
{
  char a[100];
  int length;

  printf("Enter a string to calculate it's length\n");
  gets(a);

  length = strlen(a);

  printf("Length of the string = %d\n", length);

  return 0;
}
```

strcmp() return 0

```
#include <stdio.h>
```

```
#include <string.h>

int main()
{
  char a[100], b[100];

  printf("Enter a string\n");
  gets(a);

  printf("Enter a string\n");
  gets(b);

  if (strcmp(a,b) == 0)
    printf("The strings are equal.\n");
  else
    printf("The strings are not equal.\n");

  return 0;
}
```

strcpy()

```
#include <stdio.h>
#include <string.h>

int main()
{
   char str1[10]= "awesome";
   char str2[10];
   char str3[10];

   strcpy(str2, str1);
   strcpy(str3, "well");
   puts(str2);
   puts(str3);

   return 0;
}
```

strcat() function

```
#include <stdio.h>
#include <string.h>
int main()
```

```
{
    char str1[100] = "This is ", str2[100] = "C Programming";

    strcat(str1,str2);

    puts(str1);
    puts(str2);

    return 0;
}
```