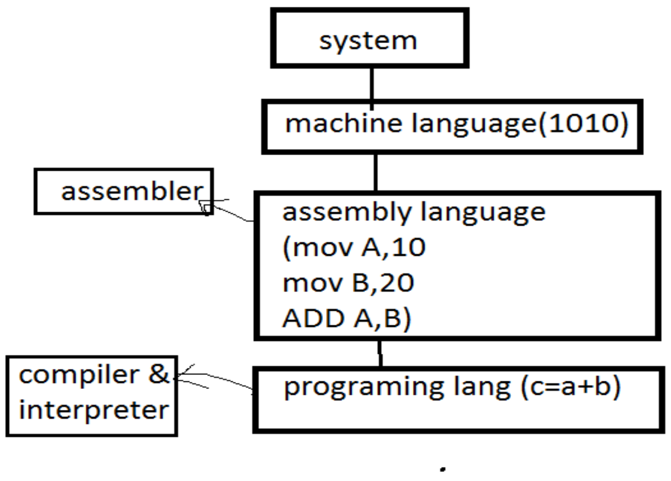
C programming language

HISTORY OF LANGUAGES



TRANSLATORS

COMPILER:-

1. It Translates total Program at a time in to machine understand language.
2. It debugs all the errors at a time.
3. Execution time is less compared to interpreter.
   EX: ‘C’, C++, JAVA Uses Compiler to translate the Program.

INTERPRETER:-

1. It Translates the Program Step by Step in to machine understand.
2. It debugs the errors one by one.
3. Execution time is more compared to compiler.
   EX: VB (Visual Basic) Uses Interpreter.

HISTORY OF ‘C’

s.no	Year	Language	Developed by
1	1960	ALGOL	International committee
2	1963	CPL	Cambridge university
3	1966	BCPL	Martin Richards
4	1967	B	Ken Thompson
5	1972	C	Dennis Ritchie

Program: A set of instruction is known as program.
Programming language: It is the interface or medium b/w user and system.
Rules of C:
1) Main () function is the start of program execution.
2) Each statement ends with a semicolon (;).
3) C language is lower case sensitive,i.e. all the predefined key words and functions must be written in lower case itself

Data Types

Classification of information is known as data types to tell what kind (format) of data it can store.

Type	Length	Range
char	1 byte	0 to 255
unsigned int	2 bytes	0 to 65,535
short int	2 bytes	-32,768 to 32,767
int	2 bytes	-32,768 to 32,767
unsigned long	4 bytes	0 to 4,294,967,295
long int	4 bytes	-2,147,483,648 to 2,147,483,647
float	4 bytes	3.4 * (10**-38) to

		3.4 * (10**+38)
double	8 bytes	1.7 * (10**-308) to 1.7 * (10**+308)
long double	10 bytes	3.4 * (10**-4932) to 3.4 * (10**+4932)

<u>Syntax for variable declaration</u>

datatype var1,var2,…;

int x,y,x;

float a,b;

<u>Format Specifiers</u>

They are used to specify the type of data displayed.

int → %d

long int → %ld

float → %f

double → %lf

long double → %Lf

char → %c

String → %s

Hexadecimal → %x

Unsigned Integer → unsigned int → %u

Operators in c examples:

/*ARITHMETICAL ASSIGNMENT OPERATORS*/

/*EXAMPLE FOR TERNARY OPERATORS*/

/*WRITE A PROGRAM TO FIND BIGGEST NUMBER IN GIVEN TWO NUMBERS*/

/*write a program to find biggest number in 3 numbers*/

INPUT & OUTPUT FUNCTIONS

OUTPUT FUNCTION

printf():

1)to display message.

syntax:

printf("message");

ex:

printf("welcome to c lang");

o/p: welcome to c lang

2) to display the value int a=10;

sytanx:

printf("c.s1.....csn", var1,.....,varn);

ex:

printf("%d",a);

o/p:10

3) to display the value with message.

syntax:

printf("msg c.s1,..msg,....c.sn",var1,....,varn);

ex:

printf("the a value is %d   ",a);

o/p:the a value is 10

INPUT FUNCITON:

SYNTAX:

scanf("c.s1.....c.sn",&var1,......,&varn);

int a;

float b,c;

char z;

ex: scanf("%d%f%f%c",&a,&b,&c,&z);

note: '&' is address or location of the variable

HEADER FILE:

A group of related functions.

#include<stdio.h> ex: printf(),scanf().,..etc

#include<conio.h> ex: clrscr(),getch().....etc

#include<math.h> ex: sin(),pow()....etc

#include<string.h> ex: strcpy(),strrev()....etc

Note:

1)If we are trying to assign float value into integer then after point the value is neglected i.e only integer value will be stored whereas vice versa for float.

int a=3.5;

Float b=8;

O/p

a=3

b=8.00000

2) Each and every fucntion ends with open and close paranthesis whereas keyword doesn't

ex: keywords: int, float

Function: printf() scanf()

Type casting examples:

/*DATA TYPE CASTING*/

/*data type casting*/

/*increment-decrement operation*/

SHORTCUT KEYS

1. ALT+F9	-	COMPILING THE PROGRAM
2. CTRL+F9	-	RUNNING THE PROGRAM.
3. F7	-	TRACING THE PROGRAM.
4. ALT+F5	-	USERS SCREEN.
5. CTRL+F1	-	HELP.

Examples:

/*write a programm to perform all arithmetical operations*/

/* write a program to find area of rectangile*/

/*write a program to find area of triangle*/

/*write a program to calculate simple interest*/

control statements:

----

1)simple if

2)if else

3)nested if else

4)if else ladder

5)switch()

simple if:

----

syntax:

```
        if(condtion)
        {
        statement-1;//if condition is true then statement-1 will be executed
        }
```

Note: In if block, only one statement is there then if block can have the flower brackets or if block can't have the flower brackets. For multiple statements compulsory flower brackets are required.

```
if(condtion)
        statement-1;//if condition is true then statement-1 will be executed
```

In simple if,first condition is checked if it is true then compiler will execute the if block or the statement-1.

Example :

Q) w.a.p to find the maximum of two numbers ?

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
clrscr();
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
if(a>b)
{
printf("a is maximum=%d",a);
}
getch();
}
```

Examples:

/* IF DECISION CONTROL STRUCTURE*/

/* write a program to find biggest number in given 3 numbers*/

draw back: In simple if, if the condition is false then compiler is not executing any statement.

note:

for if(condition) we are not specifying the semicolon because if we specify semicolon, irrespective of the condition the statement-1 will be executed i.e if the condition is false also then statement-1 is executed.

if else:

----

syntax:

```
if(condtion)
{
statement-1;//if condition is true then statement-1 will be executed
}
else
{
statement-2;//if condition is false then statement-2 will be executed
}
```

In if else, first condition is checked if it is true then compiler will execute the if block o0r the statement-1 but if the condition is false then else block or statement-2 is executed.

Example :

Q) w.a.p to find the maximum of two numbers ?

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
clrscr();
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
if(a>b)
{
printf("a is maximum=%d",a);
}
else
{
printf("b is maximum=%d",b);
}
getch();
}
```

draw back: In if else we are unable to write multiple conditions.

<u>nested if else:</u>

----

syntax:

```
        if(condtion-1)
        {
        if(condition-2)
```

```
statement-1;
else
statement-2;
}
else
{
statement-3;
}
```

In nested if else first condition-1 checked, if true then againa condition-2 is checked, if it is also true then statement-1 will be executed.but if condition-1 is true and condition-2 is false then statement-2 is executed.
if condition-1 is only false then else block or statement-3 is executed.

Q) w.a.p to find the given no is positive or negative and greater than 50 or not?

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a;
clrscr();
printf("enter a value\n");
scanf("%d",&a);
if(a>0)
{
if(a>50)
printf("a is positive and > 50");
else
printf(" a is positive and < 50");
}
else
{
printf("a is negative no");
}
getch();
}
```

draw back: if first condition is false then compiler is unable to execute remaining conditions.

if esle ladder :

----

syntax:

```
        if(condition-1)
        statement-1;
        else if(condition-2)
        statement-2;



        else if(condition-n)
        statement-n;
        else
        statement;
```

In if else first condition-1 checked,if true then statement-1 is executed but if false then condition-2 is checked,if it is true then statement-2 is executed. the same process is followed till the nth condition i.e if condition-n is true then statement-n is executed but if false then else block or statement is executed.

Q) w.a.p to accept 3 values and find maximum value?

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b,c;
clrscr();
printf("enter a,b,c values\n");
scanf("%d%d%d",&a,&b,&c);
if(a>b && a>c)
printf("a is maximum no=%d",a);
else if (b>a && b>c)
printf("b is maximum no %d",b);
else
printf("c is maximum no %d",c);
getch();
}
```

switch()

----

syntax:

```
switch(choice)
{
case label1: statement-1;
break;
case label2: statement-2;
break;
--------
--------
case labeln: statement-n;
break;
default: statement;
}
```

note: in the above syntax break,case and default are the keywords.

Q) w.a.p to accept the choice and print for 1.c lang, 2. C++ lang, 3. Java lang?

```
#include<stdio.h>
#include<conio.h>
void main()
{
int choice;
clrscr();
printf("enter the choice (1/2/3) \n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("1.C language");
break;

case 2:printf("2.C++ language");
break;
case 3: printf("3.java language");
break;
default:printf("wrong choice check again");
}
getch();
}
```

__**goto :**__ to send the compiler from one place to another place in a program.

__**forward goto:**__

syntax:

```
goto label;
st;
st;
st;
label:
```

__**backward goto:**__

syntax:

```
label:
st;
st;
st;
goto label;
```

Q) w.a.p to accept the choice and print for 1.c lang, 2. C++ lang, 3. Java lang?

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

```
{
int choice;
clrscr();
place:
printf("enter the choice (1/2/3) \n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("1.C language");
break;

case 2:printf("2.C++ language");
break;
case 3: printf("3.java language");
break;
default:printf("wrong choice check again");
goto place;
}
getch();
}
```

In the condition or control statements, only once the condition is executed but we can't execute the condition repeated number of times for that reason we are using loops

Examples:

/*FLUSHALL EXAMPLE*/

/*SPRINTF() EXAMPLE*/

/*STUDENTS MARKS MEMO*//* TO CALCULATE CURRENT BILL*/

/* WRITE A PROGRAM TO CHEK GIVEN CHARECTER IS VOWEL OR NOT*/

/*write a program to check given charecter is vowel or not*/

/*W A P TO CHECK GIVEN DATA IS VALID OR NOT*/

Loop :

The loops are classified in to 4 types

1) while loop
2) do while loop
3) for loop
4) nested for loop

increment(++)/decrement(--) operator:

++ operator will increment the variable value by 1 and -- operator will decrement the value by 1.

```
int i=1;                int i=3;
i++;            i--;
i=i+1;          i=i-1;
i=1+1;                  i=3-1;
i=2;            i=2;
```

while loop:

```
                initialisation; //optional
                while(condition)
                {
                st-1;
                st-2;
                st-n;
                ++/--; // optional
                }
```

In while loop, first compiler executes the condition, if the condition is true then while block is executed i.e the statements will be executed along with the ++/--. After that again condition is checked, if the condition is true the same process is repeated untill the condition is false.

Q) w.a.p accept a value and find the factorial?

```
#include<stdio.h>
#include<conio.h>
 void main()
 {
 int fact=1,n,i=1;
clrscr();
printf("enter the n value\n");
scanf("%d",&n);
while(i<=n)
{
fact=fact*i;
i++;
}
printf("fact=%d",fact);
getch();
}
```

when the condition is false compiler will not execute the while block.

Examples:

/* WHILE LOOP COTROL STRUCTURE*/

/*EXAMPLE (1)*/

/*example(2)*/

/*example(3)*/

/*example(4)*/

/*WRITE A PROGRAM TO FIND FACTORIAL VALUE*/

/*WRITE A PROGRAM TO PRINT ALL EVEN NUMBERS UP TO n*/

/*FOR ALL ODD NUMBERS*/

/* W.A.P TO FIND REVERSE OF GIVEN NUMBER*/

/* W A P TO FIND SUM OF DIGITS OF GIVEN NUMBER*/

/*W A P TO FIND SUM OF DIGITS OF GIVEN NUMBER(UP TO 1 DIGIT)*/

/* W A P TO CHEK GIVEN NUMBER IS PALINDROME NUMBER OR NOT.

/*WRITE A PROGRAM TO CHEK GIVEN NUMBER IS ARMSTRONG (OR)NOT*/

/*w a p to chek given number is perfect or not

/*w a p to chek given number is prime or not

/*w a p to chek given number is magic or not

/*w a p to printf FIBONACCI series

/*nested while loop*/

/*example*/

/*****ex*****/

/****example*****/

do while loop:

```
initialisation; //optional
do
{
st-1;
st-2;
st-n;
++/--; // optional
}
```

```
            while(condition);
```

In do while first the statements will be executed irrespective of the condition along with the ++/--. After that condition is checked, if the condition is true then the same process is continued untill the condition is false. when the condition is false the do while block is not executed.

Q) w.a.p accept a value and find the factorial?

```
#include<stdio.h>
#include<conio.h>
 void main()
 {
 int fact=1,n,i=1;
clrscr();
printf("enter the n value\n");
scanf("%d",&n);
do
{
fact=fact*i;
i++;
}
while(i<=n);
printf("fact=%d",fact);
getch();
}
```

Examples:

/*do---while control structure*/

/*example(1)*/

/*example(3)*factorial of number*/

<u>for loop :</u>

```
for(initialisation ;condition ;++/--)
{
st-1;
st-2;
st-n;
}
```

In for loop first compiler will execute initialisation after that condition is checked, if the condition is true then statements will be executed after that ++/-- operation is performed and then again the condition is checked, if the condition is true then the same process is continued till the condition is false. when the condition is false compiler will not execute the for block.

Q) w.a.p accept a value and find the factorial?

```
#include<stdio.h>
#include<conio.h>
 void main()
 {
 int fact=1,n,i;
clrscr();
printf("enter the n value\n");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
fact=fact*i;
}
printf("fact=%d",fact);
getch();
}
```

Examples:

```
/*FOR LOOP STRUCTURE*/
/*EXAMPLE (1)*/
/*EXAMPLE(2)*/
/*EXAMPLE(3)*/
/*W A P TO PRINT TABLE*/
/*FOR FACTORIAL*/
/*W A P TO CHEK GIVEN NUMBER IS STRONG NUMBER OR NOT
/*W A P TO PRINT THIS FORMATION
/*W A P TO PRINT THIS FORMATION          *
        * * *
       * * * * *
      * * * * * * *          */
/* w a p to print this format
  A B C D C B A      CHARACTER SET      ASCII
  A B C   C B A      A-----Z            65---90
  A B       B A      a-----z            97---122
  A           A      spase              32          */
/* w a p to print this format     1
      1 2 1
    1 2 3 2 1
   1 2 3 4 3 2 1            *
```

<u>nested for loop :</u>

```
          for(initialisation ;condition-1;++/--)  //outer loop
          {
```

```
        for(initialisation ;condition-2;++/--)  //inner loop
        {
        st-1;
        st-2;
        st-n;
        }
        }
```

In nested for loop first compiler will execute the outer loop initialisation and condition-1 is checked, if true then initialisation of inner loop is executed and condition-2 is checked,if true then statements are executed and ++/-- of inner loop is executed. After that again condition-2 is checked,if true the same process is continued untill the condition is false. if false the compiler will execute the ++/-- of outer loop and again check the condition-1, if true then inner loop will be execute from starting point onwards i.e initial point onwards when the condition is false then compiler will not execute the nested for block.

Example program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=2;i++)
{
for(j=1;j<=2;j++)
{
printf("HI\n");
}
}
getch();
}
```

Arrays:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
a=50;
a=30;
clrscr();
printf("a=%d\n",a);
printf("a=%d\n",a);
printf("a=%d\n",a);
getch();
}
```

o/p:

```
a=30
a=30
a=30
```

In the above output, 3 times the a value is printed as 30 because single variable can hold only one value at a time, it can't hold multiple values. To store the multiple values in a single variable we use arrays.

Arrays are classified in to three types :

1) single dimensional array

2) two dimensional array

3) multi dimension array

<u>single dimensional array :</u>

----

def:

collection of multiple values stored in a single variable of same data type or collection of homogenous elements is known as arrays.

syntax :

1) datatype variable_name[size]={value1,value2,...............,valuen};

ex: int a[3]={10,50,30};

float b[2]={3.2,2.5};

2) datatype variable_name[size];

ex: int a[4];

char name[20];note:

1)array index starts from zero onwards i.e a[0]=10, a[1]=50 and a[2]=30.

2)if we want single value we can represent that by index a[2]=30.

3)if we want group of values then we have to use loops while accepting, displaying or performing logic.

example program:

1) write a program to accept an array and display an array.

```
#include<stdio.h>

#include<conio.h>
```

```
void main()
{
int a[5],i;
clrscr();
printf("ACCEPTING AN ARRAY\n");
for(i=0;i<5;i++)
{
printf("enter the array elements\n");
scanf("%d",a[i]);
}
for("DISPLAYING AN ARRAY\n");
for(i=0;i<5;i++)
{
printf("a[%d]=%d\n",i,a[i]);
}
getch();
}
```

EXAMPLES:

```
/*READING DATA IN TO ARRAY AND PRINTING ELEMENTS FROM ARRAY*/
/*SUM OF ARRAY ELEMENTS*/
/*FIND BIGGEST AND SMALLEST IN ARRAY*/
/*TO INSERT ELEMENT AFTER KTH POSITION*/
/* FIND POSITION OF GIVEN ELEMENT*/
/*DELETING ELEMENT FROM ARRAY*/
/*sequential search*/
/*selection sort*/
/*reading elements*/
/*printing elements*/
/*printing elements*/
/*bubble sort*/
/*reading elements*/
```

```
/*printing elements*/
/*printing elements*/
/*INSERTION SORT*/
/*printing elements*/
/*BINARY SEARCH*/
/*printing elements*/
/*w a p to print ASCII values*/
```

2)Two dimensional array :

In order to store the values or represent the values in rows and columns we are using the two dimensional array.

syntax:

1) datatype variable_Name[row-size][column-size]={(value1,....,valuen),(valun1,...,valueNn)};

ex:

int a[2][2]={(1,2),(3,4)};

2) datatype variable_Name[row-size][column-size];

ex:

int a[2][2];

float b[2][4];

example program :

1) write a program for accepting an two dimensional array and displaying two dimensional array.

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

```
#include<conio.h>
void main()
{
int a[2][2],i,j;
clrscr();
printf("ACCEPTING AN ARRAY\n");
for(i=0;i<5;i++)
{
for(j=0;j<5;j++)
{
printf("enter the array elements\n");
scanf("%d",a[i][j]);
}
}
for("DISPLAYING AN ARRAY\n");
for(i=0;i<5;i++)
{
printf("\n");
for(j=0;j<5;j++)
{
printf("a[%d][%d]=%d\t",i,j,a[i][j]);
}
}
getch();
}
```

EXAMPLES:

/*DOUBLE DIMENSIONAL ARRAYS*/

/*EXAMPLE FOR DOUBLE DIMENSIONAL ARRAY READING ELEMENTS AND PRINTING ELEMENTS*/

/*MATRIX ADDITION*/

/*MATRIX ADDITION*/

/*MATRIX MULTIPLICATION*/

/*reading data in to matrix A*/

/*reading data in to matrix B*/

/*matrix multiplication*/

/*printing result*/

<u>arrays draw back :</u>

----

we can store only same data type elements we can't store different datatypes. To store different datatypes we can use structures concept.

In arrays we can store only similar datatype values, we can't store different data type values. To overcome that structures concept was introduced

<u>STRUCTURES:</u>

<u>DEF:</u>

Collection of different data types is known as structures.

<u>How to declare structure:</u>

```
struct structure_var
{
datatype1 var1,......,varn;
datatype2 var1,......,varn;

datatypen var1,......,varn;
};
```

ex:

```
struct student
```

```
{
int id;
float per;
char name[10];
};
```

Note :
The Data types declared inside the structure block doesn't occupy any memory until we create the object, object means allocating memory for the structure block.

How to create object:

1)syntax:

```
struct sturcture_var obj1,obj2,.........objn;
```

ex:

```
struct student s1,s2,............,sn;
```

```
struct structure_var
{
datatype1 var1,......,varn;
datatype2 var1,......,varn;

datatypen var1,......,varn;
}obj1,obj2,.........,objn;
```

ex:

```
struct student
{
int id;
float per;
char name[10];
}s1,s2,........,sn;
```

Example:

Q) w.a.p to accept student details and display the details?

```
#include<stdio.h>
#include<conio.h>
struct student
{
int id;
float per;
char name[10];
};
void main()
{
struct student raju;
clrscr();
printf("enter student details\n");
scanf("%d%f%s",&raju.id,&raju.per,&raju.name);
printf("id=%d \nper=%f \name=%s",raju.id,raju.per,raju.name);
getch();
}
```

note:

In the above program for one student, one object is created for 60 students, 60 objects are needed to be created.

ex: struct student ravi,sita;

ravi.name, ravi.per, ravi.id;
sita.name, sita.per,sita.id;

Advantage of structure:

the data type variable is declared only once and it can be accessed many number of times based on number of objects. whereas a single variable stores only one

value, for one student one variable and for 60 students 60 variables required to be decalred.
as in the example we can see "id" variable is declared only once and it is accessed twice with ravi and sita objects.

Disadvantages of structure:

As the no of objects increases the no of statements also increases.
To over come the draw back we can use array of structures.

Array of structures:
All the objects are collected in a single object with single dimension array as all are same type.
ex:
struct structure_var obj[size];
struct student s[60];

Q) w.a.p to accept three student details and display them?

```
include<conio.h>
struct student
{
int id;
char name[10];
};
void main()
{
struct student s[3];
int i;
clrscr();
for(i=0;i<3;i++)
{
printf("enter student details\n");
scanf("%d%f%s",&s[i].id,&s[i].per,&s[i].name);
}
for(i=0;i<3;i++)
{
printf("id=%d \nper=%f \name=%s",s[i].id,s[i].per,s[i].name);
}
getch();
}
```

EXAMPLES:

/*EXAMPLE (1)*/

STRUCTURE INITIALIZATION*/

STRUCTURE ARRAY*/

STRUCTURE ARRAY INITIALIZATION*/

STRUCTURE POINTER*/

/*EXAMPLE(1)

Strings :

A group of characters is known as strings.

The string in C programming language is actually a one-dimensional array of characters which is terminated by a null character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a null.

The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

```
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

The following is a list of functions found within the <string.h> header file:

1)**strcmp** String Compare

In the C Language, the required header for the strcmp function is:

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

STRCMP EXAMPLE

Let's look at an example to see how you would use the strcmp function in C program:

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

```
void main()
{
   int result;
  clrscr();
   char example1[50];
   char example2[50];

   strcpy(example1, "welcome to C programming ");
   strcpy(example2, "C programming is fun");

   result = strcmp(example1, example2);

   if (result == 0) printf("Strings are the same\n");

     (This is because the 'a' in the word 'at' is less than
      the 'i' in the word 'is' */
   if (result < 0) printf("Second string is less than the first\n");

getch();
}
```

Ex:/*w a p to perform string comparison*/

2) strcat String Concatenation

STRCAT EXAMPLE

Let's look at an example to see how you would use the strcat function in C program:

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
int main()
{
  char example[100];
  clrscr();
  strcpy(example, "TechOnTheNet ");
  strcat(example, "is ");
  strcat(example, "over ");
  strcat(example, "10 ");
  strcat(example, "years ");
  strcat(example, "old.");

  printf("%s\n", example);

getch();
}
```

Ex:/*w a p to append one string to another string*/

3)strcpy String Copy

STRCPY EXAMPLE

Let's look at an example to see how you would use the strcpy function in C program:

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

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

void  main()
{
char example[50];
   clrscr();
   strcpy (example, "TechOnTheNet.com knows strcpy!");
   printf("%s\n", example);

getch();
}
```

Ex:/* w a p to copy contents of one string to another*/

4) strlen String Length

int strlen(char array):This function accepts string as parameter and return integer i.e the length of String passed to it.

Example

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
void main()
{
   char string[]="spark";
   int len;
   len=strlen(string);
   printf("length of %s is %d\t",string,len);
}
```

Output::length of spark is 5.

Did you notice that strlen() does not include '\n' in string length or else length would be 6.

Ex:/* w a p to calculate string length*/

<u>5) strrev (string):</u>This function accepts single string as parameter and reverse that string.

Example

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

void main()
{
 char string[]="spark";
 clrscr();
 strrev(string);
printf("reverse string is %s",string);
getch();
}
```

Output: reverse string is kraps.

Ex:/*w a p to print reverse of given string*/

<u>6) strlwr (string)::</u>This function accepts single string that can be in any case(lower or upper).It converts the string in lower case.

Example

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
void main()
{
   char string1[]="SPArk";
   clrscr();
   strlwr(string1);
   printf("%s is in lower case",string1);
getch();
}
```

Output: spark is in lower case.

7) strupr (string):: This function accepts single string that can be in any case(lower or upper).It converts the string in upper case.

Example

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
void main()
{
   char string1[]="SPArk";
   clrscr();
   strupr(string1);
   printf("%s is in upper case",string1);
  getch();
}
```

8) char* strstr (main string,substring): This function accepts two strings i.e main string and substring.

It searches for the first occurrence substring in main string and returns the character pointer to the first char.

Example:/*strstr*/

/*w a p to find substring in the given string*/

Example

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
void main()
{
	char str1[]="programmingspark",str2[]="ming",*ptr;
	clrscr();
	ptr=strstr(str1, str2);
	printf("substring is: %s",ptr);
	getch();
}
```

Output : substring is mingspark

Hope you found this page useful.

Examples of strings:

/*w a to check given string palindrome or not*/

/*w a p to sort given strings*/

/*reading strings*/

/*printing strings*/

/*w a p to calculate length of string*/

/*w a p to copy contents from one string to another*/

/*w a pto reverse contents in strings*/

/*w a p to perform string comparision */

/* TO FIND SUBSTRING OF GIVEN STRINGS*/

/*W A P TO PRINT GREGORIAN CALENDER*/

Functions:

-> A set of instructions written to perform a particular task is known as function.

-> The functions can be classified into two types :

1)predefined functions

2)user defined functions

1)predefined function: A function defined by the software developer is known as predefined function.

ex: main(), printf()...etc

2)Userdefined function: A function defined by the programer is known as user defined functions.

ex: sum(),division()..etc

The user defined function has two properties:

->Function return type and parameters.

ex:

syntax:

```
return_type function_name(par1,......parn)
{
}
```

->The user defined functions can be classified into four types:

1)function without return type and without parameters.

ex: void sum();

2)function without return type and with parameters.

ex: void sum(int ,int);

3)function with return type and without parameters.

ex: int sum();

4)function with return type and with parameters.

ex: int sum(int,int);

Note:

-> void means return's nothing i.e when a function is not returning any value from sub

----

-> The user defined functions has three features:

1) function declaration or function prototype.

2)function call.

3)function body.

ex:

```
void main()
{
void show(); //function declaration and it is  function without return type and without parameters.
clrscr();
show(); // function call
getch();
}
```

```
void show() // function body
{
}
```

Note:

->A variable declared inside the function is limited to that particular function itself we can't use it outside the function.

```
void main()
{
int a=10;
show();
printf("a=%d",a); o/p: a=10.
}
void show()
{
int a=20;
}
```

->Here both the variables will occupy 2 bytes and both are different main block variable 'a' can't be used in show() and show() function variable 'a' can't be used in main() function.

-> Without function call, function body will not be executed.

-> A function body can't be written inside another function body.

-> By default the return type of user defined functions in c language is int.

1)Function without return type and wihout parameters.

w.a.p to accept two values and find sum.

program:

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

```
void main()
{
void sum();
clrscr();
sum();
getch();
}
void sum()
{
int a,b,c;
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
c=a+b;
printf("sum=%d",c);
}
```

Examples:

/*FUNCTIONS*/

/*NON RETURN TYPE FUNCTION WITH OUT ARGUMENT*/

2)function without return type and with parameters.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
void sum(int,int);
clrscr();
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
sum(a,b);
getch();
}
void sum(int x,int y)
{
int c;
c=x+y;
printf("sum=%d",c);
}
```

Examples:

/*NON RETURN TYPE FUNCTION WITH ARGUMENTS*/

3)function with return type and without parameters .

```
#include<stdio.h>
#include<conio.h>
void main()
{
int res;
int sum();
clrscr();
res=sum();
printf("sum=%d",res); // printf("sum=%d",sum());
getch();
}
int sum()
{
int a,b,c;
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
c=a+b;
return c;
}
```

Example:/*RUTURN TYPE FUNCTION WITH OUT ARGUMENTS*/

4)function with return type and with parameters.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int sum(int,int);
int a,b,res;
clrscr();
printf("enter a,b values\n");
scanf("%d%d",&a,&b);
res=sum(a,b);
printf("sum=%d",res);
getch();
}
int sum(int a,int b)
{
int c;
c=a+b;
return c;
}
```

Example:/*RETURN TYPE FUNCTION WITH ARGUMENTS*/

Miscellaneous examples:

/*W A P TO PERFORM ALL ARITHMETICAL OPERATIONS*/

/* W R P TO FIND SUM OF n NATURAL NUMBERS*/

/*W A P PROGRAM TO PRINT ALL PALINDROM NUMBERS FROM 1 TO 1000*/

/*W A P TO PRINT ALL ARMSTRONG NUMBERS FROM 1 TO 1000*/

/*W A P TO PRINT ALL PERFECT NUMBERS FROM 1 TO 1000*/

/*PRIME NUMBERS FROM 1 TO 1000*/

/*MAGIC NUMBERS FROM 1 TO 1000*/

/*STRONG NUMBERS FROM 1 TO 1000*/

/*FIBONACCI SERIES FROM 1 TO 1000*/

FILES

A file is a collection of information stored on a storage device with a particular file name.

Uses of a file:-

1) Stores the data in the form of a file and we can retrieve it whenever we require.

2) Using the data from the file in different programs.

Opening a file:- we can open a file by using fopen()

Syntax: - open(file name, opening mode);

Ex: - FILE *fp;

fp=open("cmtes.txt","w");

File opening modes:-

1)w :- Writing the data into a file. If the file already exists its contents will be over written.

2)r:- Reading data from a file.

3)a:- Adds data into an existing file.(Appending)

4)w+ :- We can write data into a file and we can read data from the file. If the file already exists its contents will be over written, else creates a new file.

5)r+ :- Reading existing contents , writing new contents, modifying existing contents of a file.

6)a+ :- Reading existing contents, appending new contents at the end of a file.

Closing a file:- When the file processing is over, the file must be closed. We can close a file by using fclose( ).

syntax:- fclose(file pointer);

ex:- fclose(fp);

File functions:-

1) getc()

This function is used to read a character from a file.

syntax :- getc(file pointer);

ex:- getc(fp);

2) putc()

This function is used to write a character into a file.

syntax:- putc(character, file pointer);

ex:- putc(ch,fp);

3)f printf():- This function writes formatted data into a file.

Syntax:- fprintf(file pointer, “formatted string”, list of variables)

Ex:- fp=fopen(“student.dat”,”w”);

fprintf (fp , “%d %s %d”,sno ,name, marks);

4)fscanf():- This function reads formatted data from a file.

Syntax:- fscanf(file pointer, “formatted string”, list of variables)

Ex:- fp=fopen(“student.dat”,”r”);

fscanf(fp , “%d %s %d”,&sno ,name,&marks);

5) fwrite( ) This function is used to write the information in a binary file.

Syntax: - fwrite(address of struct variable, size of struct variable, no. of blocks, file pointer);

ex: - fwrite(&b, sizeof(b),1,fp);

6) fread() This function is used to read the data from a binary file.

Syntax:- fread(address of struct variable, size of struct variable , no. of blocks, file pointer);

ex:- fread(&b,sizeof(b),1,fp);

7) rewind():-This function moves the record pointer to the starting position of the file.

Syntax:- rewind(file pointer);

ex:- rewind(fp);

Examples of files:

/*FILES*/

/*READING DATA FROM DISK FILE CHARECTER BY CHARECTER*/

/*WRITING DATA TO DISK FILE CHARECTER BY CHARECTER*/

/*READING DATA FROM DISK FILE LINE BY LINE*/

/* WRITING TEXT TO DISK FILE LINE BY LINE*/

/*FILE MENU PROGRAM*/

/*DYNAMIC MEMORY ALLOCATION USING "malloc"*/

/*DYNAMIC MEMORY ALLOCATION USING"calloc"*/

/*DYNAMIC MEMORY ALLOCATION USING"realloc"*/

/*printing elements*/

/*SINGLE LINKED LIST*/

/*CIRCULAR LINKED LIST*/

/*DOUBLE LINKED LIST*/

/*DOUBLE CIRCULAR*/

/*STACK implimemtation using ARRAY*/

/*QUEUE USING LINKED LIST*/

/*BINARY TREE*/

“BASIC PROGRAMS” IN C

/*write a programm to perform all arithmetical operations*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int a,b,c;
	clrscr();
	printf("enter any two numbers");
	scanf("%d %d", &a,&b);
	c=a+b;
	printf("\n addition is %d",c);
	c=a-b;
	printf("\n substraction is %d",c);
	c=a*b;
	printf("\n multiplication is %d",c);
	c=a/b;
	printf("\n division is %d",c);
	c=a%b;
	printf("\n modulo division is %d",c);
	getch();
}
```

/* write a program to find area of rectangile*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int len,bre,area;
	clrscr();
	printf("enter length and breadth");
	scanf("%d %d",&len,&bre);
	area=len*bre;
```

```
        printf("area is %d",area);
        getch();
}
```

/*write a program to find area of triangle*/

```
# include<stdio.h>
# include<conio.h>
voidmain()
{
        float bre,hei;
        clrscr();
        printf("enter breadth and height");
        scanf("%d %d",&bre,&hei);
        area=0.5*bre*hei;
        printf("area is %f",area);
        getch();
}
```

/*write a program to calculate simple interest*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        unsigned long p;
        int t;
        float r,si;
        clrscr();
        printf("enter principleamount");
        scanf("%|U",&p);
        printf("enter time in monthes");
        scanf("%f",&t);
        printf("enter rate of interest");
        scanf("%f",&r);
        si=(p*t*r)/100;
        printf("simple interest is %f", si);
        getch();
}
```

/*DATA TYPE CASTING*/

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

```
{
        int a,b;
        float c;
        clrscr();
        a= 10;
        b=3;
        c=a/b;
        printf("value of c=%f",c);
        getch();
}
```

/*data type casting*/

```
# include<stdio.h>
# include<conio.h>
void main()
{
        int a,b;
        float c;
        clrscr();
        a=10;
        b=3;
        c=(float) a/b;
        printf("value of c=%f",c);
        getch();
}
```

/*increment-decrement operation*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a;
        clrscr();
        printf("\n enter value of a");
        scanf("%d",&a);
        printf("\n post increiment a=%d",a++);
        printf("\n value of a=%d",a);
        printf("\n post decrement a=%d",a--);
        printf("\n value of a=%d",a);
        printf("\n preincrement a=%d",++a);
        printf("\n predecrement a=%d",--a);
        getch();
}
```

/*ARITHMETICAL ASSIGNMENT OPERATORS*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a=10,b=3;
        clrscr();
        printf("a=%d and b=%d",a,b);
        a+=b;
        printf("\n then a=%d",a);
        getch();
}
```

/*EXAMPLE FOR TERNARY OPERATORS*/

```
# include<stdio.h>
# include<conio.h>
void main()
{
        int age;
        clrscr();
        printf("enter your age:");
        scanf("%d",&age);
        (age>=18)?
        printf("you are eligible for voting")
        :
        printf("you are not eligible for voting");
        getch();
}
```

/*WRITE A PROGRAM TO FIND BIGGEST NUMBER IN GIVEN TWO NUMBERS*/

```
# include<stdio.h>
# include<conio.h>
void main()
{
        int a,b;
        clrscr();
        printf("enter any two numbers");
        scanf("%d %d",&a,&b);
```

```
	(a>b)?printf("%d is greater",a):printf("%d is greater",b);
	getch();
}
```

/*write a program to find biggest number in 3 numbers*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int a,b,c;
	clrscr();
	printf("enter three numbers");
	scanf("%d %d %d", &a,&b,&c);
	(a>b && a>c)?
	printf("%d is greater",a)
	:
	(b>c)?
	printf("%d is greater ",b)
	:
	printf("%d is greater",c);
	getch();
}
```

/* IF DECISION CONTROL STRUCTURE*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int age;
	clrscr();
	printf("enter your age");
	scanf("%d", &age);
	if(age>=18)
	printf("your eligible for voting");
	else
	printf("your are not eligible for voting");
	getch();
}
```

/* write a program to find biggest number in given 3 numbers*/

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

```
void main()
{
        int a,b,c;
        clrscr();
        printf("enter any three numbers");
        scanf("%d %d %d", &a,&b,&c);
        if(a>b && a>c)
        printf("%d is greater ",a);
        else
        if(b>c)
        printf("%d is greater",b);
        else
        printf("%d is greater",c);
        getch();
}
```

/*FLUSHALL EXAMPLE*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int n;
        char ch;
        clrscr();
        printf("enter any number:");
        scanf("%d", &n);
        printf("enter any charecter:");
        flushall();
        scanf("%c",&ch);
        printf("\n n=%d and ch=%c",n,ch);
        getch();
}
```

/*SPRINTF() EXAMPLE*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char str[2];
        clrscr();
        printf("hello");
        sprintf(str,"AJAY");
        printf("\n str=%s",str);
        getch();
}
```

/*STUDENTS MARKS MEMO*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int sno,m1,m2,m3,tot;
        char name[80],res[10],grade;
        float avg;
        clrscr();
        printf("enter student number:");
        scanf("%d",&sno);
        printf("enter student name:");
        flushall();
        gets(name);
        printf("enter marks........\n");
        printf("c language:");
        scanf("%d",&m1);
        printf("c++ language:");
        scanf("%d",&m2);
        printf("java language:");
        scanf("%d",&m3);
        tot=m1+m2+m3;
        if(m1<35||m2<35||m3<35)
{
        sprintf(res,"fail");
        grade='0';
        avg='0';
}
        else
{
        sprintf(res,"pass");
        avg=(float) tot/3;
        if(avg>=60)
        grade='A';
        else
        if(avg>=50)
        grade='B';
        else
        grade='C';
}
        clrscr();
        printf("\n-------------------");
        printf("\n STUDENT MARKS MEMO");
        printf("\n-------------------");
        printf("\n STUDENT NUMBER :%d",sno);
        printf("\n STUDENT NAME :%s",name);
        printf("\n C LANGUAGE :%d",m1);
        printf("\n C++ LANGUAGE :%d",m2);
        printf("\n JAVA LANGUAGE :%d",m3);
        printf("\n------------------");
        printf("\n TOTAL:%d",tot);
        printf("\n AVG:%0.2f,\n GRADE:%c,\n RESULTS:%s", avg,grade,res);
        printf("\n------------------");
        getch();
}
```

/* TO CALCULATE CURRENT BILL*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
        int cmr,pmr,units;
        float rate,tot;
        char cc;
        clrscr();
        printf("enter current month reading:");
        scanf("%d",&cmr);
        printf("enter previous month reading:");
        scanf("%d",&pmr);
        if(cmr<pmr)
{
        printf("\n invalid reading");
        exit(0);
}
        printf("\n enter customer code [dorbor:]:");
        flushall();
        scanf("%c",&cc);
        if(cc=='d' || cc=='D')
        rate=3.5;
        else if(cc=='b'||cc=='B')
        rate=7.00;
        else if (cc=='i'||cc=='I')
        rate=11.00;
        else
{
        printf("\n invalid customer code");
        exit(0);
}
        units=cmr-pmr;
        tot=units*rate;
        printf("\n number of units changed :%d",units);
        printf("\n rate of unit :%0.2f",rate);
        printf("\n amount to pay :%0.2f",tot);
        getch();
}
```

/* WRITE A PROGRAM TO CHEK GIVEN CHARECTER IS VOWEL OR NOT*/

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{
        int ch;
        clrscr();
        printf("enter any charecter...");
        scanf("%c",&ch);
        ch=tolower(ch);
        if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
        printf(" \n its vowel");
        else
        printf("\n its not vowel");
        getch();
}

main()
{
        int opt;
        clrscr();
        printf("choose your background colour...");
        printf("\n 1.blue");
        printf("\n 2.green");
        printf("\n 3.cyan");
```

```
            printf("\n 4.red");
            printf("\n enter your choice!");
            scanf("%d",&opt);
            switch(opt)
{
            case 1: textbackground(1);
                        break;
            case 2: textbackground(2);
                        break;
            case 3:  textbackground(3);
                         break;
            case 4: textbackground(4);
                        break;
            default:
                         printf("wrong choice....");
                         getch();
 }
            clrscr();
            getch();
 }
```

/*write a program to check given charecter is vowel or not*/

```
main()
{
            char ch;
            clrscr ();
            printf("enter any charecter");
            scanf("%c",&ch);
            ch=tolower(ch);
            switch(ch)
            {
                        case 'a':
                        case 'e':
                        case 'i':
                        case 'o':
                        case 'u':
                        printf("its vowel");
                        break;
                        default:
                        printf("its not vowel");
            }
                         getch();
            }
```

/*W A P TO CHECK GIVEN DATA IS VALID OR NOT*/

```
main()
{
        int dd, mm,yy,flag;
        clrscr();
        printf("enter date[dd mm yy]:");
        scanf("%d%d%d",&dd,&mm,&yy);
        if(yy<1)
        flag=0;
        else if(mm<1||mm>12)
        flag=0;
        else
{
        switch(mm)
        {
          case 1:
          case 3:
          case 5:
          case 7:
          case 8:
          case 10:
          case 12:
              if (dd<1||dd>31)
              flag=0;
              break;
          case 4:
          case 6:
          case 9:
          case 11:
              if(dd<1||dd>30)
              flag=0;
              break;
          case 2:if(yy%400==0||yy%4==0&&yy%100!=0)
          {
                  if(dd<1||dd>29)
                  flag=0;
          }
          else
          {
                  if (dd<1||dd>28)
                  flag=0;
          }
        }
```

```
   }
            if(flag==1)
            printf("date is valid");
            else
            printf("date is invalid");
            getch();
}
```

/* WHILE LOOP COTROL STRUCTURE*/

/*EXAMPLE (1)*/

```
main()
{
           int n=1;
           clrscr();
           while(n<5)
{
           printf("\n value of n=%d",n);
           n++;
}
           getch();
}
```

/*example(2)*/

```
main()
{
           int n=5;
           clrscr();
           while(n>0)
{
           printf("\n value of n=%d",n);
           n--;
}
           getch();
}
```

/*example(3)*/

```
main()
{
           int n=1;
           clrscr();
```

```
        while(n>0)
{
        printf("\n value of n=%d",n);
        n++;
}
        getch();
}
```

/*example(4)*/

```
main()
{
        clrscr();
        while(1)
        printf("\t HELLO");
        getch();
}main()
{
        int n;
        unsigned long sum=0;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        while(n>0)
{
        sum=sum+n;
        n--;
}
        printf("sum is %lu",sum);
        getch();
}
```

/*WRITE A PROGRAM TO FIND FACTORIAL VALUE*/

```
main()
{
        int n;
        unsigned long fact=1;
        clrscr();
        printf("enter any number:");
        scanf("%d",&n);
        while(n>0)
{
        fact=fact*n;
        n--;
}
        printf("\n factorial value is %lu",fact);
        getch();
}
```

/*WRITE A PROGRAM TO PRINT ALL EVEN NUMBERS UP TO n*/

```
main()
{
        int i=2,n;
        clrscr();
        printf("\n enter any number");
        scanf("%d",&n);
        printf("\n even numbers are.......");
        while(i<=n)
{
        printf("%d \t",i);
        i+=2;
}
        getch();
}
```

/*FOR ALL ODD NUMBERS*/

```
main()
{
        int i=1,n;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        printf("\n odd numbers......");
        while(i<=n)
{
        if(i%2!=0)
        printf("%d \t",i);
        i++;


}
        getch();
}
```

/* W.A.P TO FIND REVERSE OF GIVEN NUMBER*/

```
main()
{
            unsigned long n,rev=0,d;
            clrscr();
            printf("enter any number");
            scanf("%lu",&n);
            while(n>0)
      {
             d=n%10;
             rev=rev*10+d;
             n=n/10;
      }
             printf("\n reverse number is %lu",rev);
             getch();
}
```

/* W A P TO FIND SUM OF DIGITS OF GIVEN NUMBER*/

```
main()
{
            unsigned long n;
            int sum=0;
            clrscr();
            printf("enter any number");
            scanf("%lu",&n);
            while(n>0)
     {
            sum=sum+(n%10);
            n=n/10;
     }
            printf("\n sum is %d",sum);
            getch();
}
```

/*W A P TO FIND SUM OF DIGITS OF GIVEN NUMBER(UP TO 1 DIGIT)*/

```
main()
{
            unsigned long n,d;
            int sum=0;
            clrscr();
            printf("enter any number");
            scanf("%lu",&n);
```

```
	start:
	while(n>0)
    {
	d=n%10;
	sum=sum+d;
	n=n/10;
    }
	if (sum>9)
    {
	n=sum;
	sum=0;
	goto start;
    }
	printf("\n sum is %d",sum);
	getch();
}
```

/* W A P TO CHEK GIVEN NUMBER IS PALINDROME NUMBER OR NOT.

PALINDROME NUMBER:GIVEN NUMBER AND REVERSE NUMBER MUST BE EQUAL*/

```
main()
{
	unsigned long n,temp,rev=0;
	clrscr();
	printf("enter any number");
	scanf("%lu",&n);
	temp=n;
	while(n>0)
    {
	rev=(rev*10)+(n%10);
	n=n/10;
    }
	if (temp==rev)
	printf("\n its palindrome number");
	else
	printf("\n its not palindrome number");
	getch();
}
```

/*WRITE A PROGRAM TO CHEK GIVEN NUMBER IS ARMSTRONG (OR)NOT*/

```
main()
{
        int n,temp,sum=0;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        temp=n;
        while(n>0)
  {
     sum=sum+(n%10)*(n%10)*(n%10);
     n=n/10;
  }
     if (temp==sum)
     printf("\n its armstrong number");
     else
     printf("\n its not armstrong number");
     getch();
}
```

/*w a p to chek given number is perfect or not

number=6,2,8 are perfect numbers

sum of factors

here 6 factors are 1,2,3,6 but we are not considering 6 so sum=6

28 factors are 1,2,4,7,14

sum=28 */

```
main()
{
        int n,i,sum=0;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
```

```
            i=1;
            while(i<=n/2)
    {
            if(n%i==0)
            sum=sum+i;
            i++;
    }
            if (sum==n)
            printf("\n its perfect number");
            else
            printf("\n its not perfect number");
            getch();
}
```

/*w a p to chek given number is prime or not

prime numbers=2,3,5,7........

divisible by 1 and itself only*/

```
main()
{
            int i,n;
            clrscr();
            printf("enter any number");
            scanf("%d",&n);
            i=2;
            while(i<n)
      {
            if (n%i==0)
            break;
            i++;
      }
            if (i==n)
            printf("\n its prime number");
            else
            printf("\n its not prime number");
            getch();
}
```

/*w a p to chek given number is magic or not

numbers=5,25,625 are magic numbers*/

```
main()
{
        int n,n2;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        n2=n*n;
        while(n>0)
    {
        if (n%10!=n2%10)
        break;
        n=n/10;
        n2=n2/10;
    }
        if (n==0)
        printf("\n its magic number");
        else
        printf("its not magic number");
        getch();
}
```

/*w a p to printf FIBONACCI series

0,1,1,2,3,5,8,13...........*/

```
main()
{
        int prev=0,next=1,cur,n;
        clrscr();
        printf("\n enter range to print fibonacci series:");
        scanf("%d",&n);
        printf("\n %d %d",prev,next);
        cur=prev+next;
        while(cur<n)
    {
        printf("%d",cur);
        prev=next;
        next=cur;
        cur=prev+next;
    }
        getch();
}
```

/*printing till 5*/

```
main()
{
        int     n=1;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        while(n<5)
    {
        printf("%d\t",n);
        n++;
    }
        getch();
}
```

/*nested while loop*/

```
main()
{
        int i,j;
        clrscr();
        i=1;
        while(i<=5)
        {
        j=1;
        while (j<=5)
        {
        printf("%d\t",j);
        j++;
        }
        printf("\n\n");
        i++;
    }

        getch();
}
```

/*example*/

```
main()
{
	int i,j;
	i=1;
	clrscr();
	while(i<=5)
	{
		j=1;
		while(j<=i)
		{
			printf("%d\t",j);
			j++;
		}
		printf("\n");
		i++;
	}
	getch();
}
```

/*****ex*****/

```
main()
{
	int i,j;
	i=5;
	clrscr();
	while(i>=1)
	{
		j=1;
		while(j<=i)
		{
			printf("%d\t",j);
			j++;
		}
		printf("\n\n");
		i--;
	}
	getch();
}
```

/****example*****/

```
main()
```

```
{
        int i,j;
        i=5;
        clrscr();
        while(i>=1)
        {
                j=i;
                while(j>=1)

                {
                        printf("%d\t",j);
                        j--;
                }
                printf("\n\n");
                i--;
        }
        getch();
}
```

/*FOR LOOP STRUCTURE*/

/*EXAMPLE (1)*/

```
main()
{
        int i;
        clrscr();
        for (i=1;i<5;i++)
        printf("%d\t",i);
        getch();
}
```

/*EXAMPLE(2)*/

```
main()
{
        int i;
        clrscr();
        for(i=5;i>=1;i--)
        printf("%d\t",i);
        getch();
}
```

/*EXAMPLE(3)*/

```
main()
{
        clrscr();
        for(; ;)
        printf("HELLO\t\t");
}
```

/*W A P TO PRINT TABLE*/

```
main()
{
        int n,i;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        for(i=1;i<=20;i++)
        printf("\n %d*%d=%d",i,n,i*n);
        getch();
}
```

/*FOR FACTORIAL*/

```
main()
{
        int n,fact;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        for(fact=1;n>0;n--)
        fact=fact*n;
        printf("%d",fact);
        getch();
}
```

```
/*W A P TO CHEK GIVEN NUMBER IS STRONG NUMBER OR NOT

MEANING:SUM OF FACTORIALS OF DIGITS

FOR EX:1!+4!+5!

    1+24+120=145*/
main()
{
        int n,n2,temp,fact,sum=0;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        n2=n;
        while(n>0)
        {
                fact=1;
                for(temp=n%10;temp>0;temp--)
                fact=fact*temp;
                n=n/10;
                sum=sum+fact;
        }
        if(n2==sum)
        printf("\n its strong number");
        else
        printf("\n its not strong number");
        getch();
}
```

```
/*W A P TO PRINT THIS FORMATION

              *

             *  *

           *  *   *

      *   *  *   *      */
main()
{
        int i,j,n;
        clrscr();
        printf("enter no. of rows[max12]:");
        scanf("%d",&n);
        for(i=1;i<=n;i++)
        {
                for(j=1;j<=n-i;j++)
                printf("%3c",32);
                for(j=1;j<=i;j++)
                printf("%6c",'*');
                printf("\n");
        }
        getch();
}
```

```
/*W A P TO PRINT THIS FORMATION

                    *

               * * *

             * * * * *

           * * * * * * *            */
main()
{
            int i,j,n;
            clrscr();
            printf("enter no of rows[max12]:");
            scanf("%d",&n);
            for(i=1;i<=n;i++)
            {
                    for(j=1;j<=n-i;j++)
                    printf("%3c",32);
                    for(j=1;j<=(i*2)-1;j++)
                    printf("%3c",'*');
                    printf("\n");
            }
            getch();
}
```

/* w a p to print this format

```
  A B C D C B A      CHARACTER SET     ASCII

  A B C   C B A      A-----Z           65---90

  A B       B A      a-----z           97---122

  A           A      spase             32            */
```

```
  main()
  {
        int i,j,k,n;
        clrscr();
        printf("enter number of rows[max12]:");
        scanf("%d",&n);
        for(i=0;i<n;i++)
        printf("%3c",65+i);
        for(i=i-2;i>=0;i--)
        printf("%3c",65+i);
        printf("\n");
        for(i=1;i<n;i++)
        {
                for(j=0;j<n-i;j++)
                printf("%3c",65+j);
                for(k=1;k<=(i*2)-1;k++)
                printf("%3c",32);
                for(j=j-1;j>=0;j--)
                printf("%3c",65+j);
                printf("\n");
        }
        getch();
  }
```

/* w a p to print this format

pyramid of disity

```
          1

         1 2 1

      1 2 3 2 1

    1 2 3 4 3 2 1              */
```

```
main()
{
        int i,j,n;
        clrscr();
        printf("enter no. of rows[max12]:");
        scanf("%d",&n);
        for(i=1;i<n;i++)
        {
                for(j=1;j<=n-i;j++)
                printf("%3c",32);
                for(j=1;j<=i;j++)
                printf("%3d",j);
                for(j=j-2;j>=1;j--)
                printf("%3d",j);
                printf("\n");
        }
        getch();
}
   /*do---while control structure*/
```

/*example(1)*/

```
main()
{
        int n=1;
        clrscr();
        do
        {
                printf("\n %d",n);
                n++;
        }
        while(n>5);
        getch();
}
```

/*example(3)*factorial of number*/

```
main()
{
        int i,n,fact;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        i=1;
        fact=1;
        do
        {
                fact=fact*i;
                i++;
        }
        while(i<=n);
        printf("\n factorial value is%d",fact);
        getch();
}
```

/*FUNCTIONS*/

/*NON RETURN TYPE FUNCTION WITH OUT ARGUMENT*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        void add();
        clrscr();
        add();
        getch();
}
void add()
{
        int a,b,c;
        printf("enter any two numbers");
        scanf("%d %d",&a,&b);
        c=a+b;
        printf("addition is %d",c);
}
```

/*NON RETURN TYPE FUNCTION WITH ARGUMENTS*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a,b;
        void add(int,int);
        clrscr();
        printf("enter any two numbers");
        scanf("%d %d",&a,&b);
        add(a,b);
        getch();
}
void add(int x,int y)
{
        printf("addition is %d",(x+y));
}
```

/*RUTURN TYPE FUNCTION WITH OUT ARGUMENTS*/

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

```
#include<conio.h>
void main()
{
	int c;
	int add();
	clrscr();
	c=add();
	printf("\n addition is %d",c);
	getch();
}
int add()
{
	int a,b;
	printf("enter any two numbers");
	scanf("%d%d",&a,&b);
	return(a+b);
}
```

/*RETURN TYPE FUNCTION WITH ARGUMENTS*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int a,b,c;
	int add(int,int);
	clrscr();
	printf("enter any two numbers");
	scanf("%d%d",&a,&b);
	c=add(a,b);
	printf("\n addition is %d",c);
	getch();
}
int add(int x,int y)
{
	return(x+y);
}
```

/*W A P TO PERFORM ALL ARITHMETICAL OPERATIONS*/

```
#include<stdio.h>
#include<conio.h>
main()
{
        int a,b,c;
        int add(int,int);
        int sub(int,int);
        int mul(int,int);
        int div(int,int);
        clrscr();
        printf("enter any two numbers");
        scanf("%d%d",&a,&b);
        c=add(a,b);
        printf("\n addition is %d",c);
        c=sub(a,b);
        printf("\n substraction is %d",c);
        c=mul(a,b);
        printf("\n multiplidation is %d",c);
        c=div(a,b);
        printf("\n division is %d",c);
        getch();
}
        int add(int x,int y)
        {
                return(x+y);
        }
        int sub(int x,int y)
        {
                return(x-y);
        }
        int mul(int x,int y)
        {
                return(x*y);
        }
        int div(int x,int y)
        {
                return(x/y);
        }
```

/* W R P TO FIND SUM OF n NATURAL NUMBERS*/

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


  void main()
{
        int n,l;
```

```
            int sum(int);
            clrscr();
            printf("enter any number");
            scanf("%d",&n);
            l=sum(n);
            printf("sum is %d",l);
            getch();
}
int sum(int n)
{
            int l=0;
            while(n>0)
            {
                        l=l+n;
                        n--;
            }
            return(l);
}
```

/*W A P PROGRAM TO PRINT ALL PALINDROM NUMBERS FROM 1 TO 1000*/

```
int palindrome(int n)
{
            int rev=0,temp=n;
            while(n>0)
            {
                        rev=(rev*10)+(n%10);
                        n=n/10;
            }
            return(temp==rev);
}
void main()
{
            int n=1;
            clrscr();
            printf("\n palindrome numbers from 1 to 1000:\n");
            for(n=1;n<=1000;n++)
            {
                        if(palindrome(n))
                        printf("%5d",n);
            }
            getch();
}
```

/*W A P TO PRINT ALL ARMSTRONG NUMBERS FROM 1 TO 1000*/

```
int armstrong(int n)
{
	int arm=0,temp=n;
	while(n>0)
	{
		arm=arm+(n%10)*(n%10)*(n%10);
		n=n/10;
	}
	return(temp==arm);
}
void main()
{
	int n;
	clrscr();
	printf("\n armstrong numbers from 1 to 1000:\n");
	for(n=1;n<=1000;n++)
	{
		if(armstrong(n))
		printf("%5d",n);
	}
	getch();
}
```

/*W A P TO PRINT ALL PERFECT NUMBERS FROM 1 TO 1000*/

```
int perfect(int n)
{
	int i,per=0;
	for(i=1;i<=n/2;i++)
	{
		if(n%i==0)
		per=per+i;
	}
	return(per==n);
}
void main()
{
	int n;
	clrscr();
	printf("\n perfect numbers from 1 to 1000:\n");
	for(n=1;n<=1000;n++)
	{
			if(perfect(n))
			printf("%5d",n);
		}
		getch();
}
```

/*PRIME NUMBERS FROM 1 TO 1000*/

```
int prime(int n)
{
		int i;
		if(n==1)
		return 0;
		for(i=2;i<n;i++)
		{
			if(n%i==0)
			return 0;
		}
		return 1;
}
void main()
{
		int n;
		clrscr();
		printf("\n prime numbers from 1 to 1000:\n");
		for(n=1;n<=1000;n++)
		{
			if(prime(n))
			printf("%5d",n);
		}
		getch();
}
```

/*MAGIC NUMBERS FROM 1 TO 1000*/

```
int magic(int n)
{
		int n2=n*n;
		while(n>0)
		{
			if(n%10!=n2%10)
			return 0;
			n=n/10;
```

```
                n2=n2/10;
        }
        return 1;
}
void main()
{
        int n;
        clrscr();
        printf("\n magic numbersfrom 1 to 1000:\n");
        for(n=1;n<1000;n++)
        {
                if(magic(n))
                printf("%5d",n);
        }
        getch();
}
```

/*STRONG NUMBERS FROM 1 TO 1000*/

```
unsigned long factorial(int n)
{
        unsigned long fact=1;
        while(n>0)
        {
                fact=fact*n;
                n--;
        }
        return fact;
}
int strong(int n)
{
        int temp=n,str=0;
        while(n>0)
        {
                str=str+factorial(n%10);
                n=n/10;
        }
        return(temp==str);
}
void main()
{
        int n;
        clrscr();
        printf("\n strong numbers from 1 to 1000:\n");
```

```
	for(n=1;n<=1000;n++)
	{
		if(strong(n))
		printf("%5d",n);
	}
	getch();
}
```

/*FIBONACCI SERIES FROM 1 TO 1000*/

```
void fibonacci(int n)
{
	int prev=0,next=1,cur;
	printf("%d%d",prev,next);
	cur=prev+next;
	while(cur<n)
	{
		printf("%d\t",cur);
		prev=next;
		next=cur;
		cur=prev+next;
	}
}
void main()
{
	clrscr();
	printf("\n FIBONACCI SERIES from1 to 1000:\n");
	fibonacci(1000);
	getch();
}
```

/*STORAGE CLASSES*/

/*(1)AUTOMATIC STORAGE CLASS*/

/*EXAMPLE(1)*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int n;
	clrscr();
```

```
        printf("\n initial value is%d",n);
        getch();
}
```

/*EXAMPLE(2)*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int n=100;
        clrscr();
        {
                int n=500;
                {
                        int n=700;
                        printf("\n in 3rd block n=%d",n);
                }
                printf("\n in 2nd block n=%d",n);
        }
        printf("\n in main block n=%d",n);
        getch();
}
```

/*(2)REGISTER STORAGE CLASS*/

/*EXAMPLE(1)*/

```
main()
{
        register int n,a;
        clrscr();
        for(a=1;a<=100;a++)
        printf("%5d",a);
        getch();
}
```

/*(3)GLOBAL STORAGE CLASS*/

/*EXAMPLE(1)*/

```
int n;
void fun1()
```

```
{
        n=500;
        printf("\n in fun1() n=%d",n);
}
void fun2()
{
        n=900;
        printf("\n in fun 2() n=%d",n);
        n+=10;
}
void main()
{
        clrscr();
        printf("\n initial value is n=%d",n);
        n=20;
        printf("\n in main() n=%d",n);
        fun1();
        fun2();
        printf("\n in main n=%d",n);
        getch();
}
```

/*RECURSIVE FUNCTIONS*/

```
/*example(1)*/
#include<stdio.h>
#include<conio.h>
unsigned long factorial(int n)
{
        unsigned long fact;
        if(n==1)
        return 1;
        else
        fact=n*factorial(n-1);
        return fact;
}
void main()
{
        int n;
        clrscr();
        printf("\n enter any number:");
        scanf("%d",&n);
        printf("\n factorial value is %lu",factorial(n));
        getch();
}
```

/*EXAMPLE(2)*/

/*SUM OF n NATURAL NUMBERS*/

```
unsigned long sum(int n)
{
        if(n==0)
        return 0;
        else
        return n+sum(n-1);
}
main()
{
        int n;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        printf("\n sum is %lu",sum(n));
        getch();
}
```

/*EXAMPLE(3)*/

/*SUM OF DIGITS*/

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

int sum(int n)
{
        if(n==0)
        return 0;
        else
        return (n%10)+sum(n/10);
}
main()
{
        int n;
        clrscr();
        printf("enter any number");
        scanf("%d",&n);
        printf("\n sum is %d",sum(n));
        getch();
}void fibonacci(int prev,int next,int n)
{
        if(prev+next<n)
        {
                printf("%d",prev+next);
                fibonacci(next,prev+next,n);
        }
}
main()
{
        int n;
        clrscr();
        printf("enter range to print FIBONACCI SERIES:\n");
        scanf("%d",&n);
        fibonacci(0,l,n);
        getch();
}
```

/*SWAPPING USING THIRD VARIABLE*/

```
main()
{
        int a,b,c,temp;
        clrscr();
        a=100;
        b=500;
        printf("\n before swapping a=%d and b=%d",a,b);
        temp=a;
        a=b;
        b=temp;
        printf("\n after swapping a=%d and b=%d",a,b);
        getch();
}
```

/*SWAPPING WITH OUT USING THIRD VARIABLE*/

```
main()
{
        int a,b;
        clrscr();
        a=100;
        b=500;
```

```
	printf("before swapping a=%d and b=%d",a,b);
	a=a+b;
	b=a-b;
	a=a-b;
	printf("\n after swapping a=%d and b=%d",a,b);
	getch();
}
```

/*SWAPPING*/

```
main()
{
	int a,b;
	clrscr();
	a=100;
	b=500;
	printf("\n before swapping a=%d and b=%d",a,b);
	b=(a+b)-(a=b);
	printf("\n after swapping a=%d and b=%d",a,b);
	getch();
}
```

/*CALL BU VALUE: CALLING ANY FUNCTION BY PASSING VALUES AS ARGUMENTS*/

```
void swap(int x,int y)
{
	int temp;
	temp=x;
	x=y;
	y=temp;
}
main()
{
	int a,b;
	clrscr();
	a=100;
	b=500;
	printf("\n before swapping a=%d and b=%d",a,b);
	swap(a,b);
	printf("\n after swapping a=%d,b=%d",a,b);
	getch();
}
```

/*POINTER*/

/*EXAMPLE(1)*/

```
main()
{
	int *np,n=100;
	np=&n;
	clrscr();
	printf("\n value of n=%d",n);
	printf("\n value of n=%d",*np);
	printf("\n adress of n=%u",&n);
	printf("\n adress of stored in np=%u",np);
	printf("\n value of n=%d",*(&n));
	printf("\n value of n=%d",**(&np));
	getch();
}
```

/*SIZEOF*/

/*EXAMPLE(2)*/

```
main()
{
	int n,*np;
	float f,*fp;
	char ch,*cp;
	clrscr();
	printf("\n variable n allocates %d bytes and np=%d bytes",sizeof(n),sizeof(np));
	printf("\n cariable f allocates %d bytes and fp=%d bytes",sizeof(f),sizeof(fp));
	printf("\n variable ch allocates %d bytes and cp=%d bytes",sizeof(ch),sizeof(cp));
	getch();
}
```

/*EXAMPLE(3)*/

/*INCREMENTING OF POINTER MEANS IT LOCATES NEXT MEMORY LOCATION*/

```
main()
{
	int n=10,*np;
	float f=3.14,*fp;
```

```
        char ch='a', *cp;
        np=&n;
        fp=&f;
        cp=&ch;
        clrscr();
        printf("\n before incrementing______________");
        printf("\n n=%d f=%f ch=%c",n,f,ch);
        printf("\n np=%u fp=%u cp=%u",np,fp,cp);
        printf("\n after incrementing_________________");
        n++;
        f++;
        ch++;
        np++;
        fp++;
        cp++;
        printf("\n n=%d f=%f ch=%c",n,f,ch);
        printf("\n np=%u fp=%u cp=%u",np,fp,cp);
        getch();
}
```

/*VOID POINTERS*/

/*EXAMPLE(4)*/

```
main()
{
        int n=10;
        float f=3.14;
        char ch='a';
        void *gp;
        clrscr();
        gp=&n;
        printf("\n value of n=%d",*(int*)gp);
        gp=&f;
        printf("\n value of f=%f",*(float*)gp);
        gp=&ch;
        printf("\n value of ch=%c",*(char*)gp);
        getch();
}
```

/*CALL BY REFERENCE:CALLING ANY FUNCTION BY PASSING REFERENCES AS ARGUMENTS*/

```
void swap(int *x,int *y)
{
	int temp;
	temp=*x;
	*x=*y;
	*y=temp;
}
void main()
{
	int a=100,b=500;
	clrscr();
	printf("\n before swapping a=%d and b=%d",a,b);
	swap(&a,&b);
	printf("\n after swapping a=%d and b=%d",a,b);
	getch();
}
```

/*READING DATA IN TO ARRAY AND PRINTING ELEMENTS FROM ARRAY*/

```
#include<stdio.h>
#include<conio.h>
#define SIZE 5
main()
{
	int a[SIZE];
	int i;
	clrscr();
	/*reading elements*/
	printf("enter elements in array A[5]:\n\n");
	for(i=0;i<SIZE;i++)
	{
		printf("enter element at a[%d]:",i);
		scanf("%d",&a[i]);
	}
	/*printing elements*/
	printf("\n elements are__________\n\n");
	for(i=0;i<SIZE;i++)
	printf("%5d",a[i]);
	getch();
}
```

/*SUM OF ARRAY ELEMENTS*/

```
#define SIZE 5
main()
{
        int a[SIZE],i,sum=0;
        clrscr();
        printf("\n enter elements in array:\n\n");
        for(i=0;i<SIZE;i++)
        {
                printf("enter element at a[%d]:",i);
                scanf("%d",&a[i]);
                sum=sum+a[i];
        }
        printf("\n sum is %d",sum);
        getch();
}
```

/*FIND BIGGEST AND SMALLEST IN ARRAY*/

```
#define SIZE 5
main()
{
        int a[SIZE],i,big,small;
        clrscr();
        printf("\n enter elements in array:\n");
        for(i=0;i<SIZE;i++)
        {
                printf("enter element at a[%d]",i);
                scanf("%d",&a[i]);
        }
        big=small=a[0];
        for(i=1;i<SIZE;i++)
        {
        if(big<a[i])
        big=a[i];
        if(small>a[i])
        small=a[i];
}
printf("\n big=%d and small=%d",big,small);
getch();
}
```

/*TO INSERT ELEMENT AFTER KTH POSITION*/

```
main()
{
        int a[10]={10,20,45,56,78};
        int i,pos,ele,n=5;
        clrscr();
        printf("\n elements in array \"A\":");
        for(i=0;i<5;i++)
        printf("%5d",a[i]);
        printf("\n enter position where to insert:");
        scanf("%d",&pos);
        if(pos<1||pos>n)
        {
                printf("\n position not found");
                exit(0);
        }
        printf("\n enter element to insert:");
        scanf("%d",&ele);
        /*moving elements*/
        for(i=n;i>pos;i--);
        a[i]=a[i-1];
        a[pos]=ele;
        n++;
        printf("\n elements are_____________:");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        getch();
}
```

/* FIND POSITION OF GIVEN ELEMENT*/

```
main()
{
        int a[5]={10,20,45,56,78};
        int i,pos,ele,flag=0;
        clrscr();
        printf("\n elements in array :\n");
        for(i=0;i<5;i++)
        printf("%5d",a[i]);
        printf("\n enter element to find:");
        scanf("%d",&ele);
        for(i=0;i<5;i++);
        {
                if (ele==a[i])
                {
                        pos=i+1;
                        flag=1;
                        break;
                }
        }
        if(flag==1)
        printf("\n element position is %d",pos);
        else
        printf("\n element not found");
        getch();
}
```

/*DELETING ELEMENT FROM ARRAY*/

```
main()
{
        int a[]={10,23,56,45,67,90};
        int n=6,i,pos;
        clrscr();
        /*printing elements*/
        printf("\n elements in array \"A\":");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        printf("\n enter position to delete:");
        scanf("%d",&pos);
        if(pos<1||pos>n)
        {
                printf("\n position not found");
                exit(0);
        }
```

/*deleting element*/

```
        for(i=pos;i<n;i++)
        a[i-1]=a[i];
        n--;
        printf("\n after deleting elements are");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        getch();
}
```

/*sequential search*/

```
int seqsearch(int a[],int n,int ser)
{
        int i;
        for(i=0;i<n;i++);
        {
                if(ser==a[i])
                return i;
        }
        return -1;
}
main()
{
        int a[]={10,34,23,56,67,89,90,77};
        int n=8,index,ser,i;
        clrscr();
        /*printing elements*/
        printf("\n elements in array\"a\":");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        printf("\n enter element to search:");
        scanf("%d",&ser);
        index=seqsearch(a,n,ser);
        if (index==-1)
        printf("\n element not found");
        else
        printf("\n element found at index:%d",index);
        getch();
}
```

/*selection sort*/

```
main()
{
        int a[10],n,i,j,temp;
        clrscr();
        printf("enter no. of elements to store [max 10]:");
        scanf("%d",&n);
        if(n<1||n>02)
        {
                printf("array index out of range");
                exit(0);
        }
```

```
/*reading elements*/
        printf("\n enter elements________:\n");
        for(i=0;i<n;i++)
        {
                printf("enter element at a[%d]:",i);
                scanf("%d",&a[i]);
        }

/*printing elements*/
        clrscr();
        printf("\n elements before sorting:");
        for(i=1;i<n;i++)
        printf("%5d",a[i]);

        /*sorting process*/
        for(i=0;i<n-1;i++)
        {
                for(j=i+1;j<n;j++)
                {
                        if(a[i]>a[j])
                        {
                                temp=a[i];
                                a[i]=a[j];
                                a[j]=temp;
                        }
                }
        }

        /*printing elements*/
        printf("\n elements after sorting:");
        for(i=0;i<n;i++);
        printf("%5d",a[i]);
        getch();
}

/*bubble sort*/
main()
{
        int a[10],n,i,j,temp;
        clrscr();
        printf("enter no.of elements to store[max10]:");
        scanf("%d",&n);
        if(n>10)
        {
                printf("\n memory not available");
                exit(0);
        }
```

/*reading elements*/

```
        for(i=0;i<n;i++)
        {
                printf("enter element at a[%d]:",i);
                scanf("%d",&a[i]);
        }
```

/*printing elements*/

```
        printf("\n before sorting elements are__________");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        /*sorting process*/
        for(j=0;j<n-i;j++)
        {
                for(j=0;j<n-i;j++)
                {
                        if(a[j]>a[j+1])
                        {
                                temp=a[j];
                                a[j]=a[j+1];
                                a[j+1]=temp;
                        }
                }
        }
```

/*printing elements*/

```
        printf("\n after sorting elements are_______________");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        getch();
}
```

/*INSERTION SORT*/

```
main()
```

```
{
        int a[10],n,i,j,temp;
        clrscr();
        printf("enter no.of elements to store[max 10]:");
        scanf("%d",&n);
        if(n>10)
        {
                printf("\n memory not available");
                exit(0);
        }
        /*reading elements*/
        for(i=0;i<n;i++)
        {
                printf("\n enter element at a[%d]:",i);
                scanf("%d",&a[i]);
                for(j=0;j<i;j++)
                {
                        if(a[i]<a[j]);
                        {
                                temp=a[i];
                                a[i]=a[j];
                                a[j]=temp;
                        }
                }
        }
```

/*printing elements*/

```
        printf("\n after inserting elements are_______________");
        for(i=0;i<n;i++);
        printf("%5d",a[i]);
        getch();
}
```

/*BINARY SEARCH*/

```
int binsearch(int a[],int n,int ser)
{
        int first,last,mid;
        first=0;
        last=n-1;
        while(first<=last)
        {
                mid=(first+last)/2;
                if(ser==a[mid])
                return mid;
                else if(ser<a[mid])
                last=mid-1;
                else
                first=mid+1;
        }
        return -1;
}
main()
{
        int a[10],n,i,j,temp,ser;
        clrscr();
        printf("enter no.of elements to store[max10]:");
        scanf("%d",&n);
        if(n>0)
        {
                printf("\n memory not available");
                exit(0);
        }
        /*reading elements*/
        for(i=0;i<n;i++)
        {
                printf("\n enter element at a[%d]:",i);
                scanf("%d",&a[i]);
                for(j=0;j<i;j++)
                {
                        if(a[i]<a[j])
                        {
                                temp=a[i];
                                a[i]=a[j];
                                a[j]=temp;
                        }
                }
        }
```

/*printing elements*/

```
        printf("\n elements in array_____________");
        for(i=0;i<n;i++)
        printf("%5d",a[i]);
        printf("\n enter element to search:");
        scanf("%d",&ser);
        i=binsearch(a,n,ser);
        if(i==-1)
        printf("\n elements not found");
        else
        printf("\n elements found at index:%d",i);
        getch();
}
```

/*w a p to print ASCII values*/

```
main()
{
        int n;
        clrscr();
        for(n=1;n<256;n++);
        {
                printf("%d= %c",n,n);
                if(n%10==0)
                {

                        getch();
                        clrscr();
                }
        }
        getch();
}
```

/*DOUBLE DIMENSIONAL ARRAYS*/

/*EXAMPLE FOR DOUBLE DIMENSIONAL ARRAY READING ELEMENTS AND PRINTING ELEMENTS*/

```
main()
{
        int a[2][2];
        int i,j;
        clrscr();
        printf("\n enter elements in array a[2][2]:\n");
        for(i=0;i<2;i++)
        {
                for(j=0;j<2;j++)
                {
                        printf("\n enter element at a[%d][%d]:",i,j);
```

```
				scanf("%d",&a[i][j]);
			}
		}
		/*printing elements*/
		printf("\nelements are:\n");
		for(i=0;i<2;i++)
		{
			for(j=0;j<2;j++)
			printf("%5d",a[i][j]);
			printf("\n");
		}
		getch();
}
```

/*MATRIX ADDITION*/

```
#define row 2
#define col 2
main()
{
		int a[row][col],b[row][col],c[row][col];
		int i,j;
		clrscr();
		printf("\n enter data for matrix\"a\":\n\n");
		for(i=0;i<row;i++)
		{
			for(j=0;j<col;j++)
			{
				printf("enter element at a[%d][%d]:",i,j);
				scanf("%d",&a[i][j]);
			}
		}
		printf("\n enter data for matrix\"b\":\n\n");
		for(i=0;i<row;i++)
		{
			for(j=0;j<col;j++)
			{
				printf("enter element at b[%d][%d]:",i,j);
				scanf("%d",&b[i][j]);
			}
		}
```

/*MATRIX ADDITION*/

```
	for(i=0;i<row;i++)
	{
		for(j=0;j<col;j++)
		c[i][j]=a[i][j]+b[i][j];
	}
	/*printing result matrix*/
	printf("\n matrix addition is :\n\n");
	for(i=0;i<row;i++)
	{
		for(j=0;j<col;j++)
		printf("%5d",c[i][j]);
		printf("\n");
	}
	getch();
}
```

/*MATRIX MULTIPLICATION*/

```
main()
{
	int a[10][10],b[10][10],c[10][10];
	int nr1,nc1,nr2,nc2;
	int i,j,k;
	clrscr();
	printf("\n enter first matrix size rows cols maxsize[10 10]:");
	scanf("%d%d",&nr1,&nc1);
	if(nr1>10||nc1>10)
	{
		printf("array index out of range");
		exit(0);
	}
	printf("\n enter first matrix size rows cols amx size[10 10]:");
	scanf("%d%d",&nr2,&nc2);
	if(nr2>10||nc2>10)
	{
		printf("array index out of range");
		exit(0);
	}
	if(nc1!=nr2)
	{
		printf("\n matrix multiplication is not possible");
		exit(0);
	}
```

/*reading data in to matrix A*/

```
clrscr();
printf("\n enter data for matrix A[%d][%d]:\n",nr1,nc1);
for(i=0;i<nr1;i++)
{
        for(j=0;j<nc1;j++)
        {
                printf("enter element at a[%d][%d]:",i,j);
                scanf("%d",&a[i][j]);
        }
}
```

/*reading data in to matrix B*/

```
clrscr();
printf("\n enter data for mayrix B[%d][%d]:\n",nr2,nc2);
for(i=0;i<nr2;i++)
{
        for(j=0;j<nc2;j++)
        {
                printf("enter element at B[%d][%d]:",i,j);
                scanf("%d",&b[i][j]);
        }
}
```

/*matrix multiplication*/

```
for(i=0;i<nr1;i++)
{
        for(j=0;j<nc2;j++)
        {
                c[i][j]=0;
                for(k=0;k<nc1;k++)
                c[i][j]=c[i][j]+(a[i][k]*b[k][j]);
        }
}
```

/*printing result*/

```
printf("\n matrix multiplication c[%d][%d]:\n",nr1,nc2);
for(i=0;i<nr1;i++)
{
        for(j=0;j<nc2;j++)
        printf("%5d",c[i][j]);
```

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

/*STRINGS*/

/*Strlen*/

/* w a p to calculate string length*/

```
#include<string.h>
main()
{
          char str[80];
          int len;
          clrscr();
          printf("\n enter any string:");
          gets(str);
          len=strlen(str);
          printf("\n stringstr=\"%s\"",str);
          printf("\n length is=%d",len);
          getch();
}
```

/*Strcpy*/

/* w a p to copy contents of one string to another*/

```
#include<string.h>
main()
{
          char src[80],dest[80];
          clrscr();
          printf("\n enter source string :");
          gets(src);
          strcpy(dest,src);
          printf("\n after copying______________");
          printf("\n source string src=%s",src);
          printf("\n destination string dest=%s",dest);
          getch();
}
```

/*Strrev*/

/*w a p to print reverse of given string*/

```
#include<stdio.h>
main()
{
	char str[80];
	clrscr();
	printf("\n enter any string :");
	gets(str);
	printf("\n before reversing contents str:%s",str);
	strrev(str);
	printf("\n after reversing cotents str:%s",str);
	getch();
}
```

/*strcat*/

/*w a p to append one string to another string*/

```
#include<string.h>
main()
{
	char dest[80],src[80];
	clrscr();
	printf("\n enter source string :");
	gets(src);
	printf("\n enter destination string:");
	gets(dest);
	printf("\n src=%s and dest=%s",src,dest);
	printf("\n after appending______________");
	strcat(dest,src);
	printf("\n src=%s and dest=%s",src,dest);
	getch();
}
```

/*strstr*/

/*w a p to find substring in the given string*/

```
#include<string.h>
main()
{
        char str1[80],str2[80];
        char *s;
        clrscr();
        printf("\n enter string str 1:");
        gets(str1);
        printf("\n enter any word in the previous string:");
        gets(str2);
        s=strstr(str1,str2);
        printf("\n substring is:%s",s);
        getch();
}
```

/*Strcmp*/

/*w a p to perform string comparison*/

```
#include<string.h>
main()
{
        char str1[80],str2[80];
        int i;
        clrscr();
        printf("\n enter string s1:");
        gets(str1);
        printf("\n enter string str2:");
        gets(str2);
        i=strcmp(str1,str2);
        if(i>0)
        printf("\n %s is greater than %s",str1,str2);
        else if(i<0)
        printf("\n %s is greater thanm %s",str2,str1);
        else
        printf("\n both are equal");
        getch();
}
```

/*w a to check given string palindrome or not*/

```
#include<string.h>
main()
{
        char st[80],temp[80];
        clrscr();
        printf("enter any string:");
        gets(st);
        strcpy(temp,st);
        printf("\n given string st:%s",st);
        strrev(st);
        printf("\n reverse string st:%s",st);
        if(strcmpi(st,temp)==0)
        printf("\n its palindrome string");
        else
        printf("\n its not palindrome string");
        getch();
}
```

/*w a p to sort given strings*/

```
#include<string.h>
main()
{
        char str[10][30];
        char temp[30];
        int n,i,j;
        clrscr();
        printf("enter no. of strings to store:");
        scanf("%d",&n);
        if(n>10)
        {
                printf("\n memory not available");
                exit(0);
        }
```

/*reading strings*/

```
        printf("\n enter strings ...........\n");
        for(i=0;i<n-1;i++)
        {
                printf("enter string at str[%d]:",i);
                flushall();
                gets(str[i]);
```

```
	}
```

/*printing strings*/

```
	clrscr();
	printf("\n before sorting strings are.........\n");
	for(i=0;i<n;i++)
	printf("\n \t\t\t%s",str[i]);
	/*sorting process*/
	for(i=0;i<n-1;i++)
	{
		for(j=i+1;j<n;j++)
		{
			if(strcmpi(str[i],str[j]>0))
			{
				strcpy(temp,str[i]);
				strcpy(str[i],str[j]);
				strcpy(str[j],temp);
			}
		}
	}
	printf("\n after sorting strings are ...........\n");
	for(i=0;i<n;i++)
	printf("\n\t\t\t %s",str[i]);
	getch();
}
```

/*w a p to calculate length of string*/

```
#include<string.h>
int strlength(const char *s)
{
	int i;
	i=0;
	while(*(s+i))
	i++;
	return i;
}
main()
{
	char st[80];
	clrscr();
	printf("\n enter any string:");
	gets(st);
	printf("\n length is:%d",strlength(st));
	getch();
}
```

/*w a p to copy contents from one string to another*/

```
#include<string.h>
void strcopy(char *s1,char *s2)
{
	int i;
	for(i=0;*(s2+i);i++)
	*(s1+i)=*(s2+i);
	*(s1+i)='\0';
}
main()
{
	char st1[80],st2[80];
	clrscr();
	printf(" enter any string:");
	gets(st1);
	strcopy(st2,st1);
	printf("\n string st1:%s",st1);
	printf("\n after copying___________");
	printf("\n string s2:%s",st2);
	getch();
}
```

/*w a pto reverse contents in strings*/

```
#include<string.h>
int strlength(char *s)
{
	int i;
	for(i=0;*(s+i);i++);
	return i;
}
void reverse(char *s)
{
	int i,j,len;
	char ch;
	len=strlength(s);
	j=len-1;
```

```
        for(i=0;i<len/2;i++,j--)
        {
                ch=*(s+i);
                *(s+i)=*(s+j);
                *(s+j)=ch;
        }
}
main()
{
        char str[80];
        clrscr();
        printf("enter any string:");
        gets(str);
        printf("\n given string str:%s",str);
        reverse(str);
        printf("\n reverse string str:%s",str);
        getch();
}
```

/*w a p to perform string comparision */

```
int compare(char *s1,char *s2)
{
        int i;
        for(i=0;*(s1+i);i++)
        {
                if(*(s1+i)!=*(s2+i))
                return *(s1+i)-*(s2+i);
        }
        return (*(s1+i)-*(s2+i));
}
void main()
{
        int i;
        char st1[80],st2[80];
        clrscr();
        printf("enter any string:");
        gets(st1);
        printf("enter another string:");
        gets(st2);
        i=compare(st1,st2);
        if(i>0)
        printf("%s is greater than %s",st1,st2);
        else if(i<0)
```

```
		printf("%s is greater than %s",st2,st1);
		else
		printf("both are equal");
		getch();
}
```

/* TO FIND SUBSTRING OF GIVEN STRINGS*/

```
char *substring(char *src,char *dest)
#include<string.h>
{
		int i,j,k;
		for(i=0;*(src+i);i++)
		{
				j=0;
				if(*(dest+j)==*(src+i))
				{
						k=i;
						for(j=0;*(dest+i);j++,k++)
						{
								if(*(dest+j)!=*(src+k)
								break;
						}
						if(*(dest+j)==NULL)
						return(src+i);
				}

		}return NULL;
}
main()
{
		char str1[80],str2[80],*s;
		clrscr();
		printf("enter source string:");
		gets(str1);
		printf("enter any word to find in previous string:");
		gets(str2);
		printf("\n sub string is %s",s);
		getch();
}
```

/*W A P TO PRINT GREGORIAN CALENDER*/

```
int getday(int mm,int yy)
```

```
{
        int month_days[]={31,28,31,30,31,30,31,31,30,31,30,31};
        unsigned long totdays;
        int curdays,nleap,i,day;
        if(yy%400==0||yy%4==0&&yy%100!=0)
        month_days[1]=29;
        totdays=(unsigned long)(yy-1)*365;
        nleap=((yy-1)/400)+((yy-1)/4)-((yy-1)/100);
        totdays+=nleap;
        curdays=0;
        for(i=0;i<mm-1;i++)
        curdays+=month_days[i];
        totdays+=curdays;
        return(totdays)%7;
}
void printcalender(int mm,int yy)
{
        int month_days[]={31,28,31,30,31,30,31,31,30,31,30,31};
        char month_name[][20]={"january","february","march","april","may",
                              "june","july","august","september","october",
                              "november","december"};
        char week_names[][20]={"monday","tuesday","wednesday","thursday",
                                  "friday","saturday","sunday"};
        int c,r,i,day;
        if(yy%400==0||yy%4==0&&yy%100!=0)
        month_days[1]=29;
        gotoxy(30,3);
        printf("%s-%d",month_name[mm-1],yy);
        c=3;
        r=5;
        for(i=0;i<7;i++)
        {
                gotoxy(c,r);
                printf("%s",week_names[i]);
                c+=10;
        }
        day=getday(mm,yy);
        c=3+(day*10);
        r=7;
        for(i=1;i<=month_days[mm-1];i++)
        {
                gotoxy(c,r);
                printf("%d",i);
                c+=10;
                if(c==73)
                {
                        c=3;
                        r+=2;
                }
        }
}
void title()
{
        gotoxy(1,1);
        printf("use arrow keys");
        gotoxy(60,1);
        printf("alt+x to exit");
        gotoxy(30,1);
        printf("gregorian calender");
        gotoxy(70,25);
        printf("AJAY");
}
main()
{
        int mm,yy;
        char ch;
        clrscr();
        printf("enter month and year:");
        scanf("%d%d",&mm,&yy);
        if(mm<1||mm>12||yy<1)
        {
                printf("\n invalid values:");
                exit(0);
        }
        do
        {

        clrscr();
        title();
        printcalender(mm,yy);
        ch=getch();
        switch(ch)
        {
                case 80:
                        mm++;
                        if(mm==13)
                        {
                                mm=1;
```

```
                    yy++;
                }
                break;
        case 72:
                mm--;
                if(mm<1)
                {
                    mm=12;
                    yy--;
                }
                break;
        case 75:
                yy--;
                if(yy<1)
                yy=1;
                break;
        case 77:
                yy++;
                break;
        case 45:
                clrscr();
                gotoxy(30,12);
                printf("good bye sir________");
                getch();
                exit(0);
        }
    }while(1);
}
```

/*STRUCTURES*/

/*EXAMPLE (1)*/

```
struct student
{
    int sno;
    char sname[80];
    float fee;
};
main()
{
    struct student s;
    clrscr();
    printf("\n enter student information _____________\n");
```

```
        printf("\n student number:");
        scanf("%d",&s.sno);
        printf("\n student name:");
        flushall();
        gets(s.sname);
        printf("course fee:");
        scanf("%f",&s.fee);
        printf("\n student details____________\n");
        printf("\n student number :%d",s.sno);
        printf("\n student name:%s",s.sname);
        printf("\n course fee:%0.2f",s.fee);
        printf("\n memory allocations_____________");
        printf("\n memory occupied by variables=%d bytes",sizeof(s));
        printf("\n address of s.sno:%u",&s.sno);
        printf("\n address of s.sname:%u",&s.sname);
        printf("\n address of s.fee :%u",&s.fee);
        getch();
}
```

/*EXAMPLE(2)

STRUCTURE INITIALIZATION*/

```
struct student
{
        int sno;
        char sname[80];
        float fee;
};
main()
{
        struct student s={101,"abc",2500.00};
        clrscr();
        printf("\n student details..........\n");
        printf("\n student number:%d",s.sno);
        printf("\n student name:%s",s.sname);
        printf("\n course fee:%0.2f",s.fee);
        getch();
}
```

/*EXAMPLE(3)

STRUCTURE ARRAY*/

```
struct student
{
	int sno;
	char sname[80];
	float fee;
};
main()
{
	struct student s[3];
	int i;
	float fee;
	clrscr();
	printf("\n enter student details.........\n");
	for(i=0;i<3;i++)
	{
		printf("\n record [%d]:",i+1);
		printf("\n enter student number:");
		scanf("%d",&s[i].sno);
		printf("\n enter student name");
		flushall();
		gets(s[i].sname);
		printf("\n enter fee          :");
		scanf("%f",&fee);
		s[i].fee=fee;
	}
	printf("\n student details______________\n");
	for(i=0;i<3;i++)
	printf("\n %10d  %30s   %12.2f",s[i].sno,s[i].sname,s[i].fee);
	getch();
}
```

/*EXAMPLE (4)

STRUCTURE ARRAY INITIALIZATION*/

```
struct student
{
	int sno;
	char sname[80];
```

```
        float fee;
}s[]={
        1001,"xxx",5000.00,
        1002,"yyy",9000.00,
        1003,"zzz",12000.00
   };
main()
{
        int i;
        clrscr();
        printf("\n student details......\n");
        for(i=0;i<3;i++)
        printf("\n %10d  %30s    %12.2f",s[i].sno,s[i].sname,s[i].fee);
        getch();

}
```

/*EXAMPLE(5)

STRUCTURE POINTER*/

```
struct student
{
        int sno;
        char sname[80];
        float fee;

};
main()
{
        struct student *ptr,s={101,"xyz",5000.00};
        clrscr();
        ptr=&s;
        printf("\n student number:%d",ptr->sno);
        printf("\n student name:%s",ptr->sname);
        printf("\n coyrse fee:%0.2f",ptr->fee);
        printf("\n student number :%d",(*ptr).sno);
        printf("\n student name:%s",(*ptr).sname);
        printf("\n course fee:%0.2f",(*ptr).fee);
        getch();
}
```

/*EXAMPLE(1)

UNION*/

```
union student
{
	int sno;
	char sname[80];
	float fee;
};
main()
{
	union student s;
	clrscr();
	printf("\n enter student number :");
	scanf("%d",&s.sno);
	printf("\n enter student name:");
	flushall();
	gets(s.sname);
	printf("\n enter fee:");
	scanf("%f",&s.fee);
	printf("\n student number:%d",s.sno);
	printf("\n student name:%s",s.sname);
	printf("\n course name:%0.2f",s.fee);
	getch();
}
```

/*FILES*/

/*READING DATA FROM DISK FILE CHARECTER BY CHARECTER*/

```
#include<stdio.h>
main()
{
	FILE *fp;
	char fname[20];
	char ch;
	clrscr();
	printf("enter file name to reads:");
	scanf("%s",fname);
	fp=fopen(fname,"r");
	if(fp==NULL)
	{
		printf(" file not found");
		exit(0);
	}
	ch=fgetc(fp);
	while(ch!=EOF)
	{
		printf("%c",ch);
		ch=fgetc(fp);
		delay(10000);
	}
	fclose(fp);
	getch();
}
```

/*WRITING DATA TO DISK FILE CHARECTER BY CHARECTER*/

```
#include<stdio.h>
main()
{
	FILE *fp;
	char fname[20];
	char ch;
	clrscr();
	printf(" enter file name to read:");
	scanf("%s",fname);
	fp=fopen(fname,"w");
	if(fp==NULL)
	{
		printf("file not found");
		exit(0);
	}
	printf("\n enter some charecters press(cntrl+z)to save:\n");
	ch=getchar();
	while(ch!=EOF)
	{
		fputc(ch,fp);
		ch=getchar();
	}
	fclose(fp);
	printf("\n1.file copyied");
	getch();
}
```

/*READING DATA FROM DISK FILE LINE BY LINE*/

```
#include<stdio.h>
#include<dos.h>
main()
{
	FILE *fp;
	char fname[20];
	char ch,st[80];
	clrscr();
	printf("enter file name to read:");
	scanf("%s",fname);
	fp=fopen(fname,"r");
	if(fp==NULL)
	{
		printf("file not found");
		exit(0);
	}
	fgets(st,80,fp);
	while(!feof(fp))
	{
		printf("%s",st);
		fgets(st,80,fp);
		delay(10000);
	}
	fclose(fp);
	getch();
}
```

/* WRITING TEXT TO DISK FILE LINE BY LINE*/

```
#include<stdio.h>
#include<string.h>
main()
{
	FILE *fp;
	char fname[20];
	char ch,st[80];
	clrscr();
	printf("enter file name to read:");
	scanf("%s",fname);
	fp=fopen(fname,"w");
	if(fp==NULL)
	{
		printf("file not found");
```

```
                exit(0);
        }
        printf("\n enter some line of text[quit] to stop:\n");
        gets(st);
        while(strcmpi(st,"quit")!=0)
        {
                fputs(st,fp);
                fputs("\n",fp);
                gets(st);
        }
        fclose(fp);
        printf("\n1.file copyied");
        getch();
}


#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct emp
{
        int eno;
        char ename[80];
        float esal;
}e;
main()
{
        char ch;
        FILE *fp;
        fp=fopen("employ.txt","a+");
        do
        {
                clrscr();
                printf("\n enter employ number:");
                scanf("%d",&e.eno);
                printf("enter employ name :");
                flushall();
                gets(e.ename);
                printf("\n enter employ salary :");
                scanf("%f",&e.esal);
                fprintf(fp,"\n %10d %40s %12.2f",e.eno,e.ename,e.esal);
                printf("\n do u wish to continue[y/n]:");
                flushall();
                ch=getchar();
        }while(ch=='y'||ch=='y');
        fseek(fp,ol,SEEK_SET);
        while(fscanf(fp,"%d %s %f",&e.eno,&e.ename,&e.esal)>0)
        printf("\n %10d %40s %12.2f",e.eno,e.ename,e.esal);
        fclose(fp);
        getch();
}
```

/*FILE MENU PROGRAM*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
struct emp
{
        int eno;
        char ename[80];
        float esal;
}e;
FILE *fp1,*fp2;
void openfile()
{
        fp1=fopen("dummy.txt","ab");
        if(fp1==NULL)
        {
                printf("\n unable to open file:");
                exit(0);
        }
}
        void addrecord()
        {
                openfile();
                printf("\n enter emp number:");
                scanf("%d",&e.eno);
                printf("\n enter emp name:");
                flushall();
                gets(e.ename);
                printf("enter emp salary:");
                scanf("%f",&e.esal);
                fwrite(&e,sizeof(e),1,fp1);
                printf("\n insert done........");
                fclose(fp1);
        }
```

```
void label()
{
        int i;
        printf("\n %10s %40s %12s","emp_number","emp_name",
        "emp_salary");
        for(i=1;i<=80;i++);
        printf("%c",196);
}
void display()
{
        openfile();
        label();
        while(fread(&e,sizeof(e),1,fp1))
        printf("\n %10d %40s %12.2f",e.eno,e.ename,e.esal);
        fclose(fp1);
}
void search()
{
        int num,flag=0;
        openfile();
        printf("\n enter employ number to search:");
        scanf("%d",&num);
        while(fread(&e,sizeof(e),1,fp1))
        {
                if(e.eno==num)
                {
                        flag=1;
                        printf("\n %10d %40s %12.2f",e.eno,e.ename,e.esal);
                        break;
                }
        }
        if(flag==0)
        printf("\n record not found");
        fclose(fp1);
        }
        void modify()
        {
                int num,flag=0;
                fp2=fopen("dummy.txt","w");
                openfile();
                printf("enter employ number to modify:");
                scanf("%d",&num);
                while(fread(&e,sizeof(e),1,fp1))
                {
```

```
                if(e.eno==num)
                {
                        flag=1;
                        printf("enter employ salary:");
                        scanf("%f",&e.esal);
                }
                fwrite(&e,sizeof(e),1,fp2);
        }
        fcloseall();
        remove("emp7.txt");
        rename("dummy.txt","emp7.txt");
        if(flag==1)
        printf("\n update done.......");
        else
        printf("\n record not found");
}
void delrecord()
{
        int num,flag=0;
        fp2=fopen("dummy.txt","w");
        openfile();
        printf("\n enter employ number to delete:");
        scanf("%d",&num);
        while(fread(&e,sizeof(e),1,fp1))
        {
        if(e.eno==num)
        {
                flag=1;
        }
        else
        fwrite(&e,sizeof(e),1,fp2);
        }
        fcloseall();
        remove("emp7.txt");
        rename("dummy.txt","emp7.txt");
        if(flag==1)
        printf("\n delete done...........");
        else
        printf("\n record not found");
}
main()
{
        int opt;
        clrscr();
        do
        {
                clrscr();
                printf("\n employ details.......");
                printf("\n 1.enter new employ record:");
                printf("\n 2.search employ record:");
                printf("\n 3.modify employ record:");
                printf("\n 4.delete employ record:");
                printf("\n 5.list of all employ records:");
                printf("\n 6.exit:");
                printf("\n\n enter your choice");
                scanf("%d",&opt);
                switch(opt)
                {
                        case 1:
                                addrecord();
                                break;
                        case 2:
                                search();
                                break;
                        case 3:
                                modify();
                                break;
                        case 4:
                                delrecord();
                                break;
                        case 5:
                                display();
                                break;
                        case 6:
                                exit(0);
        default:
                printf("\n wrong choice");
                getch();
        }
        getch();
        }while(1);
}
```

/*DYNAMIC MEMORY ALLOCATION USING "malloc"*/

```
#include<stdio.h>
main()
{
	int *a,n,i;
	clrscr();
	printf("enter number of elements to store:");
	scanf("%d",&n);
	a=(int*)malloc(n *sizeof(int));
	if(a==NULL)
	{
		printf("memory not available");
		exit(0);
	}
	printf("\n enter elements____________\n");
	for(i=0;i<n;i++)
	{
		printf("enter element at %u:",a+i);
		scanf("%d",a+i);
	}
	/*printing elements*/
	printf("\n elements are______________\n");
	for(i=0;i<n;i++)
	printf("\n %5d",*(a+i));
	getch();
}
```

/*DYNAMIC MEMORY ALLOCATION USING"calloc"*/

```
#include<stdio.h>
main()
{
	int *a,n,i;
	clrscr();
	printf("enter no.of elements to store:");
	scanf("%d",&n);
	a=(int*)calloc(n, sizeof(int));
	if(a==NULL)
	{
		printf("memory not available");
		exit(0);
	}
	printf("\n enter elements___________\n");
	for(i=0;i<n;i++)
```

```
        {
                printf("enter element at %u:",a+i);
                scanf("%d",a+i);
        }
        /*printing elements*/
        printf("\n elements are___________\n");
        for(i=0;i<n;i++)
        printf("\n %5d",*(a+i));
        getch();
}
```

/*DYNAMIC MEMORY ALLOCATION USING"realloc"*/

```
#include<stdio.h>
main()
{
        int *a,*ptr,n1,n2,i;
        clrscr();
        printf("enter number of elements to store:");
        scanf("%d",&n1);
        a=(int*)calloc(n1,sizeof(int));
        if(a==NULL)
        {
                printf("memory not available");
                exit(0);
        }
        printf("\n enter elements______________\n");
        for(i=0;i<n1;i++)
        {
                printf("enter element at %u:",a+i);
                scanf("%d",a+i);
        }
        printf("\n enter number of elements to store:");
        scanf("%d",&n2);
        ptr=(int*)realloc(a,n1+n2);
        for(i=n1;i<n1+n2;i++)
        {
                printf("enter element at %u :",ptr+i);
                scanf("%d",ptr+i);
        }
        free(a);
        /*printing elements*/
        n2=n1+n2;
        printf("\n elements are________________\n");
        for(i=0;i<n2;i++)
        printf("\n %u %5d",ptr+i,*(ptr+i));
        getch();
}
```

/*SINGLE LINKED LIST*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct emp
{
        int eno;
        char ename[80];
        float esal;
        struct emp *next;
};
void main()
{
        struct emp *first=NULL,*cur,*last,*temp;
        char ch;
        float *f,f1;
        f=&f1;
        printf("%f",*f);
        clrscr();
        do
        {
                cur=(struct emp*)malloc(sizeof(struct emp));
                if(cur==NULL)
                {
                        printf("\n memory not available");
                        exit(0);
                }
                printf("\n enter emp number :");
                scanf("%d",&cur->eno);
                printf("enter emp name :");
                flushall();
                gets(cur->ename);
                printf("enter emp salary :");
                scanf("%f",&cur->esal);
                cur->next=NULL;
```

```
		if(first==NULL)
		first=last=cur;
		else
		{
			last->next=cur;
			last=cur;
		}
		printf("\n Do u wish to continue[y/n]:");
		flushall();
		scanf("%c",&ch);
		}while(ch=='y'||ch=='Y');
		temp=first;
		printf("\n employ details____________\n");
		while(temp)
		{
		printf("\n %d %s %f",temp->eno,temp->ename,temp->esal);
		temp=temp->next;
		}
		getch();
	}
```

/*CIRCULAR LINKED LIST*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct emp
{
	int eno;
	char ename[80];
	float esal;
	struct emp *next;
};
void main()
{
	struct emp *first=NULL,*cur,*last,*temp;
	char ch;
	float *f,f1;
	f=&f1;
	printf("%f",*f);
	clrscr();
	do
{
        cur=(struct emp*)malloc(sizeof(struct emp));
        if(cur==NULL)
        {
                printf("\n memory not available:");
                exit(0);
        }


        printf("\n enter emp number:");
        scanf("%d",&cur->eno);
        printf("\n enter emp name:");
        flushall();
        gets(cur->ename);
        printf("\n enter emp salary:");
        scanf("%f",&cur->esal);
        cur->next=NULL;
        if(first==NULL)
        {
                first=last=cur;
                last->next=first;
        }
        else
        {
                last->next=cur;
                last=cur;
                last->next=first;
        }
        printf("\n Do u wish to continue [y/n]:");
        flushall();
        scanf("%c",&ch);
        }while(ch=='y'||ch=='Y');
        temp=first;
        printf("\n employ details____________\n");
        do
        {
        printf("\n %d %s %f",temp->eno,temp->ename,temp->esal);
        temp=temp->next;
        }while(temp!=first);
        getch();
}
```

/*DOUBLE LINKED LIST*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct number
{
        struct number *prev;
        int data;
        struct number *next;
};
void main()
{
        struct number *first=NULL,*cur,*last,*temp;
        char ch;
        clrscr();
        do
        {
                cur=(struct number*)malloc(sizeof(struct number));
                if(cur==NULL)
                {
                        printf("\n memory not available");
                        exit(0);
                }
                printf("enter any number :");
                scanf("%d",&cur->data);
                if(first==NULL)
                {
                        first=last=cur;
                        last->next=NULL;
                        first->prev=NULL;
                }
                else

                {
                        last->next=cur;
                        cur->prev=last;
                        last=cur;
                        last->next=NULL;
                }
                printf("\n Do u wish to continue[y/n]:");
                flushall();
                scanf("%c",&ch);
                }while(ch=='y'||ch=='Y');
```

```
		printf("\n data in FORWARD:");
		temp=first;
		while(temp)
		{
			printf("%5d",temp->data);
			temp=temp->next;
		}
		printf("\n data is BACKWARD:");
		temp=last;
		while(temp)
		{
			printf("%5d",temp->data);
			temp=temp->prev;
		}
		getch();
	}
```

/*DOUBLE CIRCULAR*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct number
{
	struct number *prev;
	int data;
	struct number *next;
};
void main()
{
	struct number *first=NULL,*last,*cur,*temp;
	int n;
	clrscr();
	do
	{
		printf("enter any number[-999]to stop:");
		scanf("%d",&n);
		if(n==-999)
		break;
		cur=(struct number *)malloc(sizeof(struct number));
		if(cur==NULL)
		{
			printf("memory not available");
```

```
                        exit(0);
                }
                cur->data=n;
                if(first==NULL)
                {
                        first=last=cur;
                        last->next=first;
                        first->prev=last;
                }
                else
                {
                        last->next=cur;
                        cur->prev=last;
                        last=cur;
                        last->next=first;
                        first->prev=last;
                }
        }while(1);
        printf("\n list in forward:");
        temp=first;
        do
        {
                printf("%5d",temp->data);
                temp=temp->next;
        }while(temp!=first);
        printf("\n list in backward:");
        temp=last;
        do
        {
                printf("%5d",temp->data);
                temp=temp->prev;
        }while(temp!=last);
        getch();
}                       /*STACK*/
```

/*STACK implimemtation using ARRAY*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define SIZE 5
int stk[SIZE];
int tos=-1;
void push(int ele)
```

```
{
        if(tos==SIZE-1)
        printf("\n stack over flow");
        else
        stk[++tos]=ele;
}
int pop(int *ele)
{
        if (tos==-1)
        return 0;
        else
        *ele=stk[tos--];
        return 1;
}
        void display()
        {
                int i;
                for(i=tos;i>=0;i--)
                printf("\n \t\t\t %d",stk[i]);
        }
        void main()
        {
                int num,opt;
                do
                {
                        clrscr();
                        display();
                        printf("\n stack operations.........\n");
                        printf("\n1.push");
                        printf("\n2.pop");
                        printf("\n3.exit");
                        printf("enter your option:");
                        scanf("%d",&opt);
                        switch(opt)
                        {
                                case 1:
                                        printf("\n enter any number:");
                                        scanf("%d",&num);
                                        push(num);
                                        break;
                                case 2:
                                        if(pop(&num))
                                        printf("\n popped element is %d",num);
                                        else
                                        printf("\n stak underflow");
                                        break;
                              case 3:
                                        exit(0);
                    default:
                              printf("\n wrong choice");
                              getch();
                    }
                    getch();
          }while(1);
}
```

/*STACK IMPLIMENTATION USING LINKED LIST*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct stack
{
          int data;
          struct stack *next;
};
struct stack *last=NULL;
void push(int ele)
{
          struct stack *cur=(struct stack *)malloc(sizeof(struct stack));
          if(cur==NULL)
          printf("\n stak overflow");
          else
          {
                    cur->data=ele;
                    cur->next=last;
                    last=cur;
          }
}
int pop(int *ele)
{
          struct stack *temp;
          if(last==NULL)
          return 0;
          else
          {
                    *ele=last->data;
                    temp=last;
```

```
		last=last->next;
		free(temp);
	}
	return 1;
	}
	void display()
	{
		struct stack *temp=last;
		while(temp)
		{
			printf("\n\t\t\t %d",temp->data);
			temp=temp->next;
		}
	}
	void main()
	{
		int num,opt;
		do
		{
			clrscr();
			display();
			printf("\n stack operations..........\n");
			printf("\n1.push");
			printf("\n2.pop");
			printf("\n3.exit");
			printf("enter your option:");
			scanf("%d",&opt);
			switch(opt)
			{
				case 1:
					printf("\n enter any number:");
					scanf("%d",&num);
					push(num);
					break;
				case 2:
					if(pop(&num))
					printf("\n popped element is %d",num);
					else
					printf("stack underflow");
					break;
				case 3:exit(0);
		default:
			printf("\n wrong choice.........");
			getch();
```

```
                }
                getch();
        }while(1);
}
```

/*QUEUE USING LINKED LIST*/

```
#include<stdio.h>
#include<conio.h>
struct queue
{
        int data;
        struct queue *next;
};
struct queue *first=NULL,*cur,*last;
void insert(int num)
{
        cur=(struct queue *)malloc(sizeof(struct queue));
        if(cur==NULL)
        {
                printf("queue is full");
        }
        else
        {
                cur->data=num;
                cur->next=NULL;
                if(first==NULL)
                first=last=cur;
                else
                {
                        last->next=cur;
                        last=cur;
                }
        }
}
int delnode(int *num)
{
        struct queue *temp;
        if(first==NULL)
        return 0;
        else
        {
                *num=first->data;
                temp=first;
                first=first->next;
                free(temp);
        }
        return 1;
}
void display()

{
        struct queue *temp=first;
        while(temp)
        {
                printf("%5d",temp->data);
                temp=temp->next;
        }
}
void main()
{
        int num,opt;
        do
        {
                clrscr();
                printf("\n1.insert");
                printf("\n2.delete");
                printf("\n3.exit");
                printf("\n enter your option:");
                scanf("%d",&opt);
                switch(opt)
                {
                        case 1:
                                printf("\n enter number in to queue:");
                                scanf("%d",&num);
                                insert(num);
                                break;
                        case 2:
                                if(delnode(&num))
                                printf("\n deleted from queue%d",num);
                                else
                                printf("\n queue is empty");
                                break;
                        case 3:
                                exit(0);
                default:
                        printf("\n wrong choice...........");
                        getch();
                }
                getch();
        }while(1);
}
```

/*BINARY TREE*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct tree
{
        struct tree *left;
        int data;
        struct tree *right;
};
struct tree *insert(struct tree *root,int num)
{
        if(root==NULL)
        {
                root=(struct tree *)malloc(sizeof(struct tree));
                root->data=num;
                root->left=NULL;
                root->right=NULL;
        }
        else if(num>root->data)
        root->right=insert(root->right,num);
        else if(num<root->data)
        root->left=insert(root->left,num);
        return root;
}
void inorder(struct tree *root)
{
        if(root)
        {
                inorder(root->left);
                printf("%5d",root->data);
                inorder(root->right);
        }
}
void preorder (struct tree *root)
{
        if(root)
        {
```

```
		printf("%5d",root->data);
		preorder(root->left);
		preorder(root->right);
	}
}
void postorder(struct tree *root)
{
	if(root)

	{
		postorder(root->left);
		postorder(root->right);
		printf("%5d",root->data);
	}
}
void main()
{
	int num;
	struct tree *root=NULL;
	do
	{
	clrscr();
	printf("\n inorder:");
	inorder(root);
	printf("\n preorder:");
	preorder(root);
	printf("\n postorder:");
	postorder(root);
	printf("\n enter element[-999 to stop]:");
	scanf("%d",&num);
	if(num==-999)
	break;
	root=insert(root,num);
	}while(1);
}
```