**FACTORIAL**

int factorial(int n)

{

   if(n==0){

    return(1);

   }

else{

        int f=n\*factorial(n-1);

        return f;

}

}

int main()

{

    int fact, num;

    printf("enter the number");

    scanf("%d",&num);

     fact=factorial(num);

     printf("factorial is %d",fact);

**}**

**Dynamic pointer**

#include <stdio.h>

#include <stdlib.h>

int main()

{

      int \*ptr ,n ,i, j,temp;

      printf("Enter the total numbers : ");

      scanf("%d",&n);

      ptr=(int \*)malloc(n\*sizeof(int));

      printf("\nEnter %d Numbers: \n",n);

      for(i=0;i<n;i++)

      {

            scanf("%d", (ptr+i));

      }

      for(i=0;i<n;i++)

      {

            for(j=i+1;j<n;j++)

            {

                  if(\*(ptr+i) > \*(ptr+j))

                  {

                        temp=\*(ptr+i);

                        \*(ptr+i)=\*(ptr+j);

                        \*(ptr+j)=temp;

                  }

            }

      }

      printf("\nAfter Sorting in Ascending Order: \n");

      for(i=0;i<n;i++)

      printf("\n%d",\*(ptr+i));

      return 0;

}

**Without return**

#include <stdio.h>

int add(int \*i, int \*j)

{

int add;

add = \*i + \*j;

}

int main()

{

int a, b;

printf(" Input two numbers: ");

scanf("%d %d", &a,&b);

add(&a, &b);

printf("sum is %d", add);

}

**Structure**

Every member is assigned a unique memory location.

Change in the value of one data member does not affect other data members in the structure.

You can initialize multiple members at a time.

A structure can store multiple values of the different members

A structure’s total size is the sum of the size of every data member

Users can access or retrieve any member at a time.

**Union**

All the data members share a memory location.

Change in the value of one data member affects the value of other data members.

You can initialize only the first member at once.

A union stores one value at a time for all of its members

. A union’s total size is the size of the largest data member.

Users can access or retrieve only one member at a time.

#include <stdio.h>

#include <string.h>

int main()

{

    char a[] = "Hello ";

    char b[] = "World!";

    strcat(a,b);

    printf("Concatenated String: %s\n", a);

    return 0;

}

Pattern

#include <stdio.h>

int main()

{

int n,i=1;

printf("enter the number\n");

scanf("%d",&n);

for (int j=0;j<n;j++)

{

if(i==1)

{

for(i=0;i<n;i++)

{

printf("%d",i+1);

}

}

else

{

for(i=n+1;i>1;i--)

{

printf("%d",i-1);

}

}

printf("\n");

}

return 0;

}

ENUM

#include<stdio.h>

typedef enum{Mon, Tue, Wed, Thur, Fri, Sat, Sun}week;

int main()

{

week day;

day = Wed;

printf("%d",day);

return 0;

}

STRINGIZING

#include <stdio.h>

#define outstr(x,y) #x#y

int main()

{

printf("Out :"outstr(Hellow,world));

return 0;

}

**ARRAY OF STRUCTURE**

#include <stdio.h>

struct details

{

char name[20];

int age;

};

int main()

{

struct details d[3]

={ {"boby",15},

{"rahul",12},

{"anwar",16} };

for(int i=0;i<3;i++)

{

printf("%d name :%s \n",i+1,d[i].name);

printf("Age : %d",d[i].age);

}

return 0;

}