Q1. **Write a C++ program using pointers to compute the sum of all elements in an array.**

Ans:

```
#include<bits/stdc++.h>

using namespace std;


int main()

{

 int n;

 cout<<"\nEnter the number of elements";

 cin>>n;

 int *arr = new int[n];

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

    cin>>arr[i];

 int sum=0;

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

    sum+= *(arr+i);

 cout<<sum;

 return 0;

}
```

Output:

```
D:\Codes\arr_point.exe

Enter the number of elements 5
10 20 30 40 50
150
Process returned 0 (0x0)   execution time : 43.162 s
Press any key to continue.
```

Q2. **Write a C++ program using function that generates and prints first n Fibonacci numbers.**

Ans:

```
 #include<bits/stdc++.h>

 using namespace std;


 int Fib(int n)

 {

  if(n==1)

    return 1;

  else if(n==2)

    return 1;

  else

    return Fib(n-1) + Fib(n-2);

 }

int main()

{

   int n;

   cout<<"\nEnter the number of terms";

   cin>>n;

   for(int i=1; i<=n; i++)

     {cout<<Fib(i)<<" ";}

return 0;

}
```

Output:

Q3. **Write a C++ program that uses a function to sort an array.**

Ans:

```
  #include<bits/stdc++.h>

 using namespace std;



 int * Sort(int* arr, int n)

 {

   int temp;

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

     for(int j=0; j<n-1; j++)

        if(arr[j]>arr[j+1])

        {

           temp = arr[j];

           arr[j] = arr[j+1];

           arr[j+1] = temp;

        }

    return  arr;

}

int main()

{

    int n;

    cout<<"\nEnter the number of terms";

    cin>>n;

    int *a = new int[n];

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

      cin>>a[i];

     a= Sort(a,n);

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

        cout<<a[i]<<" ";

return 0;

}
```

Output:

```
D:\Codes\arr_point.exe

Enter the number of terms 6
20 40 10 50 35 66
10 20 35 40 50 66
Process returned 0 (0x0)   execution time : 15.362 s
Press any key to continue.
```

Q4. **Write a C++ program using recursive calls to evaluate f(x) = x- (x^3)/3! + (x^5)/5! – (x^7)/7! + …..**

Ans:

```
  #include<bits/stdc++.h>

 using namespace std;


 int Fact(int n)

{

  if(n==0)

   return 1;

  else if(n==1)

  return 1;

  else

   return n*Fact(n-1);

}


 float Eval(float x, int n)

{

  float p=x;

  if(n==1)

     return x;

  else

  {

     if(n%2==0)

        p = -(pow(x, 2*n-1)/ Fact(2*n-1));
```

```
      else
          p =  (pow(x, 2*n-1)/ Fact(2*n-1));
    }
    return p + Eval(x,n-1);


 }
 int main()
{
   int n;
   float x;
   cin>>x>>n;
   float sum = Eval(x,n);
   cout<<sum;
  return 0;
}
```

Output:

- WAP using pointer to read in an array of integer and print its element in an reverse order

```
#include<bits/stdc++.h>
using namespace std;
int main(){
 int n;
 cout<<"Enter the no. of integers: ";
 cin>>n;
 int a[n];
 for(int i=0;i<n;i++)
  cin>>*(a+i);

 cout<<"array in reversed order ";
 for(int i=0; i<n;i++)
  cout<<*(a+n-1-i)<<" ";

 return 0;
}
```


```
#include<bits/stdc++.h>
using namespace std;
int main(){
 int n;
 cout<<"Enter the no. of integers: ";
 cin>>n;
 int a[n];
 for(int i=0;i<n;i++)
  cin>>*(a+i);

 cout<<"array in reversed order ";
 for(int i=0; i<n;i++)
  cout<<*(a+n-1-i)<<" ";

 return 0;
}
```


output

```
Enter the no. of integers: 4
1 2 4 6
array in reversed order 6 4 2 1
```

- WAP using a function to calculate roots of a quadratic equation, function must use two pointers.

```
#include<bits/stdc++.h>
#include<math.h>
using namespace std;

int root(float *rptr,float *arr){
 int flag=0;
 float a=arr[0],b=arr[1],c=arr[2];
 float D=b*b-4*a*c;
 if(D>=0){
  rptr[0]=(-b+sqrt(D))/(2*a);
```

```
  rptr[1]=(-b-sqrt(D))/(2*a);
  flag=1;
 }
 return flag;
}

int main(){
 float r[2],coef[3];
 cout<<"enter the values of a,b,c: ";
 for(int i=0;i<3;i++)
   cin>>coef[i];

 int flag=root(r,coef);
 if (flag==0)
  cout<<"Real roots do not exist";

 else if(flag==1){
  if(r[0]==r[1])
    cout<<"Roots are equal.\nRoot: "<<r[0]<<endl;
  else
    cout<<"Roots :  "<<r[0]<<" and "<<r[1]<<endl;
 }
 return 0;
}
```

output

```
enter the values of a,b,c: 1 5 6
Roots :  -2 and -3
```

- WAP to search an element using pointer

```
#include<bits/stdc++.h>
#include<math.h>
using namespace std;

int main(){
 int n;
 cout<<"Enter the no. of elements: ";
 cin>>n;
 int a[n];
 cout<<"\nEnter the elements: ";
 for(int i=0;i<n;i++)
  cin>>a[i];

 int key;
 cout<<"\nEnter the key to be searched for: ";
 cin>>key;
 int flag=0;
 int i=0;
 for(i=0;i<n;i++){
  if(*(a+i)==key){
          flag=1;
```

```
        break;
  }
 }
 if (flag==1)
        cout<<"\nElement found at position:"<<i+1<<endl;
 else
        cout<<"Element not found"<<endl;
 return 0;
}
```

output:

```
Enter the no. of elements: 5

Enter the elements: 1 4 3 7 2

Enter the key to be searched for: 3

Element found at position:3
```

- WAP to take an input and print it using pointer

```
#include<bits/stdc++.h>
#include<math.h>
using namespace std;

int main(){
 int a;
 cout<<"enter the value of variable:";
 cin>>a;
 int *ptr=&a;
 cout<<"\nThe value of the variable is: "<<*ptr<<endl;
 return 0;
}
```

output:

```
enter the value of variable:5

The value of the variable is: 5
```

- WAP to find the largest element in an array

```
#include<bits/stdc++.h>
using namespace std;

int main(){
        cout<<"\nenter the no. of elements in the array:";
        int n;
        cin>>n;
        int a[n];
        cout<<"\nenter the elements in the array:";
        int m=INT_MIN;
        for(int i=0;i<n;i++){
                cin>>a[i];
```

```
            if(a[i]>m)
                m=a[i];
        }
        cout<<"\nlargest element in the array is:"<<m<<endl;
        return 0;
}
```

output:

```
enter the no. of elements in the array: 4

enter the elements in the array: 1 3 23 65

largest element in the array is: 65
```

- WAP to insert an element in an array

```
#include<iostream>
using namespace std;

int main(){
int n;
cout<<"enter no. of elements you want to enter";
cin>>n;
int ar[n+1];
cout<<"enter elements ";
for(int i=0;i<n;i++)
cin>>ar[i];

int index=-1,a;
cout<<"enter index where you want to insert and enter no.:";
cin>>index>>a;
for(int i=n+1;i>index;i--)
ar[i]=ar[i-1];

ar[index]=a;
for(int i=0;i<n+1;i++)
cout<<ar[i]<<" ";
return 0;
}
```

output:

```
enter no. of elements you want to enter5
enter elements 1 3 2 5 6
enter index where you want to insert and enter no.:4 23
1 3 2 5 23 6
```

- WAP to delete an element from an array

```
#include<iostream>
using namespace std;

int main(){
int n;
```

```
cout<<"enter no. of elements you want to enter ";
cin>>n;
int ar[n];
cout<<"enter the elements in the array: ";
for(int i=0;i<n;i++)
cin>>ar[i];

int index=-1;
cout<<"\nenter index where you want to delete the element: ";
cin>>index;

for(int i=index;i<n;i++)
ar[i]=ar[i+1];

for(int i=0;i<n-1;i++)
cout<<ar[i]<<" ";
return 0;
}
```

output:

```
enter no. of elements you want to enter 5
enter the elements in the array: 1 4 2 7 9

enter index where you want to delete the element: 3
1 4 2 9
```

- WAP of linear search

```
#include<iostream>
using namespace std;

int main(){
	int n,flag =0;
	cout<<"enter the number of elements in the array\n";
	cin>>n;
	int a[n];
	for(int i=0;i<n;i++)
	cin>>a[i];

	int temp;
	cout<<"enter the value to be searched\n";
	cin>>temp;
	for(int i=0;i<n;i++){
		if(a[i]==temp)
		flag =1;
	}
	if(flag)
	cout<<"present \n";
	else
	cout<<"not present";
}
```

output:

```
enter the number of elements in the array
5
1 3 2 5 6
enter the value to be searched
3
present
```

- WAP to sort an array using bubblesort

```
#include<iostream>
using namespace std;

int main(){
        int n;
        cout<<"enter the number of elements in the array\n";
        cin>>n;
        int a[n];
        for(int i=0;i<n;i++)
        cin>>a[i];
        int temp ;
   for(int i =0;i<n-1;i++)
   {
        for(int j=0;j<n-i-1;j++)
        {
                if(a[j]>a[j+1])
                {
                        temp = a[j+1];
                        a[j+1] = a[j];
                        a[j] = temp;
                        }
                }
        }
        for(int i=0;i<n;i++)
        cout<<a[i]<<" ";
}
```

output:

```
enter the number of elements in the array
4
1 4 2 9
sorted array is 1 2 4 9
```

- WAP to search an element using binary search

```
#include<bits/stdc++.h>
using namespace std;
int main(){
        int n,flag =-1;
        cout<<"enter the number of elements in the array\n";
        cin>>n;
        int a[n];
        for(int i=0;i<n;i++)
        cin>>a[i];
   sort(a,a+n);
        int first=0,mid,last=n-1;
```

```
	cout<<"enter the value to be searched\n";
	int temp;
	cin>>temp;
	while(first<=last)
	{
		mid = (first+last)/2;
		if(a[mid] == temp)
		{
			flag = mid;
			break;
		}
		else if(a[mid]>temp)
		{
			last = mid-1;
		}
		else
		{
			first=mid+1;
		}
	}
	if(flag!=-1)
	cout<<"In sorted array present at position "<<flag<<endl;
	else
	cout<<"not present";
return 0;
}
```

output:

```
enter the number of elements in the array
5
1 5 2 4 10
enter the value to be searched
1
In sorted array present at position 0
```

- WAP to create stack using linked list

```
#include<iostream> // insertion and deletion in stack using linked list.
using namespace std;

struct node{
	int data;
	node* next;
};
node* top = NULL;
void push(int x)
{
	node* temp = new node();
	temp->data = x;
	temp->next = top;
	top = temp;	// this line can directly include both the cases i.e., one when top  is null and
second when top is not null;
```

```
}
void pop()
{
	node* temp = top;
	if(top == NULL)
	cout<<"stack underflow";
	else
	top = temp->next;
	delete temp;
}
void print()
{
	node* temp = top;
	while(temp!=NULL)
	{
		cout<<temp->data<<" ";
		temp = temp->next;
	}
}
int main(){
	push(10);
	push(180);
	push(130);
	push(120);
	push(150);
	push(1330);
	push(1230);

	print();
	cout<<endl;
	pop();

	print();
}
```

output:

```
present list is 1230 1330 150 120 130 180 10
stack after one deletion 1330 150 120 130 180 10
```

- WAP to create stack using array

```
#include<iostream>
using namespace std;
int size=10,top=-1,ar[10];
void insert(int item){
 if(top==size-1) cout<<"stack is full";
 else{
  top++;
  ar[top]=item;
  }
 }
```

```
void Delete(){
 if(top==-1) cout<<"stack is empty";
 else
  top--;
 }

void print(){
 cout<<"present stack is";
 for(int i=top;i>=0;i--)
  cout<<ar[i]<<" ";
  cout<<endl;
 }

int main(){
 int n,item;

 while(1){
 cout<<" 1 for insert, 2 for delete,any other for exit \n";
 cin>>n;
  if(n==1){
   cout<<"enter element";
   cin>>item;
   insert(item);
   print();
   }
  else if(n==2){
   Delete();
   print();
  }
  else return 0;

 }
 return 0;
}
```

output:

```
 1 for insert, 2 for delete,any other for exit
1
enter element12
present stack is12
 1 for insert, 2 for delete,any other for exit
1
enter element16
present stack is16 12
 1 for insert, 2 for delete,any other for exit
2
present stack is12
```

Q. Program to transpose a sparse matrix

```
#include<bits/stdc++.h>

using namespace std;

int main(){
        cout << "enter limit\n";
        int n, temp;
        cin>>n;
        int ar[n][3];
        for(int i=0;i<n;i++){
                cin>>ar[i][0]>>ar[i][1]>>ar[i][2];
        }

      for(int i=0; i<n; i++){
        for(int j=0;j< n-1-i; j++){
          if(ar[j][1]>ar[j+1][1]){
            temp=ar[j][0];
                ar[j][0]=ar[j+1][0];
                ar[j+1][0]=temp;
            temp=ar[j][1];
                ar[j][1]=ar[j+1][1];
                ar[j+1][1]=temp;
            temp=ar[j][2];
                ar[j][2]=ar[j+1][2];
                ar[j+1][2]=temp;
               }
               }
               temp=ar[n-1-i][1];
               ar[n-1-i][1]=ar[n-1-i][0];
               ar[n-1-i][0]=temp;

        }

      cout<<endl;
        for(int i=0;i<n;i++){
                cout<<ar[i][0]<<" "<<ar[i][1]<<" "<<ar[i][2];
           cout<<endl;
        }
}
```

Output:

```
enter limit
9
1 1 2
1 4 1
2 2 2
2 5 3
3 3 1
4 2 2
4 4 1
5 1 3
5 3 1

1 1 2
1 5 3
2 2 2
2 4 2
3 3 1
3 5 1
4 1 1
4 4 1
5 2 3

Process returned 0 (0x0)   execution time : 65.395 s
Press ENTER to continue.
```

Q. Program to add two sparse matrices

```
#include <stdio.h>
#include<stdlib.h>
struct Element
{
 int i;
 int j;
 int x;
};
struct Sparse
{
 int m;
 int n;
 int num;
 struct Element *ele;
};
void create(struct Sparse *s)
{
 int i;

 printf("Eneter Dimensions");
 scanf("%d%d",&s->m,&s->n);
 printf("Number of non-zero");
 scanf("%d",&s->num);

 s->ele=(struct Element *)malloc(s->num*sizeof(struct
Element));
 printf("Eneter non-zero Elements");
 for(i=0;i<s->num;i++)
 scanf("%d%d%d",&s->ele[i].i,&s->ele[i].j,&s->ele[i].x);

}
void display(struct Sparse s)
{
 int i,j,k=0;

 for(i=0;i<s.m;i++)
 {
 for(j=0;j<s.n;j++)
 {
 if(i==s.ele[k].i && j==s.ele[k].j)
 printf("%d ",s.ele[k++].x);
 else
 printf("0 ");
 }
 printf("\n");
 }
}
struct Sparse * add(struct Sparse *s1,struct Sparse *s2)
{
 struct Sparse *sum;
```

```
 int i,j,k;
 i=j=k=0;

 if(s1->n != s2->n && s1->m != s2->m)
 return NULL;
 sum=(struct Sparse *)malloc(sizeof(struct Sparse));
sum->ele=(struct Element *)malloc((s1->num+s2->num)*sizeof(struct Element));
 while(i<s1->num && j<s2->num)
 {
 if(s1->ele[i].i<s2->ele[j].i)
 sum->ele[k++]=s1->ele[i++];
 else if(s1->ele[i].i>s2->ele[j].i)
 sum->ele[k++]=s2->ele[j++];
 else
 {
 if(s1->ele[i].j<s2->ele[j].j)
 sum->ele[k++]=s1->ele[i++];
 else if(s1->ele[i].j>s2->ele[j].j)
 sum->ele[k++]=s2->ele[j++];
 else
 {
 sum->ele[k]=s1->ele[i];
 sum->ele[k++].x=s1->ele[i++].x+s2->ele[j++].x;
 }

 }
 }
 for(;i<s1->num;i++)sum->ele[k++]=s1->ele[i];
 for(;j<s2->num;j++)sum->ele[k++]=s2->ele[j];
 sum->m=s1->m;
 sum->n=s1->n;
 sum->num=k;


 return sum;
}
int main()
{
 struct Sparse s1,s2,*s3;

 create(&s1);
 create(&s2);

 s3=add(&s1,&s2);

 printf("First Matrix\n");
 display(s1);
 printf("Second Matrix\n");
 display(s2);
 printf("Sum Matrix\n");
 display(*s3);
```

```
 return 0;
}
```

Output:

```
Eneter Dimensions5 5
Number of non-zero5
Eneter non-zero Elements0 0 1
1 1 1
2 2 1
3 3 1
4 4 1
Eneter Dimensions5 5
Number of non-zero5
Eneter non-zero Elements0 0 2
1 0 2
2 0 2
3 0 2
4 0 2
Sum Matrix
3 0 0 0 0
2 1 0 0 0
2 0 1 0 0
2 0 0 1 0
2 0 0 0 1

Process returned 0 (0x0)   execution time : 58.836 s
Press ENTER to continue.
```

Q. Program for insertion and deletion in Stack using array

```
#include<iostream>
using namespace std;

#define MAXSIZE 10
int top =-1;

int a[MAXSIZE];

void push(int data)
{
	if(top==MAXSIZE)
	cout<<"stack is full"<<endl;
	else
	{
		top = top+1;
		a[top] = data;
	}
}
int pop()
{
	int temp;
	if(top==-1)
	cout<<"stack is empty"<<endl;
	else
	{
		temp = a[top];
		top--;
		return temp;
	}
}
void print()
{
	if(top==-1)
	cout<<"stack is empty "<<endl;
	else
	{
		for(int i=0;i<=top;i++)
		cout<<a[i]<<" ";
		cout<<endl;
	}
}
int main()
{
	push(200);
	push(87);
	push(35);
	push(89);
	push(99);
	print();
	int poped = pop();
```

```
        cout<<"poped is "<<poped<<endl;
        print();
}
```

Output:

```
200 87 35 89 99
poped is 99
200 87 35 89

Process returned 0 (0x0)   execution time : 0.001 s
Press ENTER to continue.
```

Q. Program to convert infix to postfix expression.

```
#include<bits/stdc++.h>
using namespace std;
int prec(char c)
{
    if(c == '^')
    return 3;
    else if(c == '*' || c == '/')
    return 2;
    else if(c == '+' || c == '-')
    return 1;
    else
    return -1;
}
void conversion(string s){
    stack<char> st;
    st.push('#');
    int l=s.length();
    string ns;
    for(int i=0;i<l;i++){
        if((s[i]>='a' && s[i]<='z') ||(s[i]>='A' && s[i]<='Z')){
            ns+=s[i];
        }
        else if(s[i]=='('){
            st.push(s[i]);
        }
        else if(s[i] == ')')
        {
            while(st.top() != '#' && st.top() != '(')
            {
                char c = st.top();
                st.pop();
               ns += c;
            }
            if(st.top() == '(')
            {
                char c = st.top();
                st.pop();
            }
        }
        else{
            while(st.top() != '#' && prec(s[i]) <= prec(st.top()))
            {
                char c = st.top();
                st.pop();
                ns += c;
            }
            st.push(s[i]);
        }

    }
```

```
    while(st.top() != '#')
    {
        char c = st.top();
        st.pop();
        ns += c;
    }

    cout << ns << endl;
    }
int main()
{
    string exp = "a+b*(c^d-e)^(f+g*h)-i";
    conversion(exp);
    return 0;
}
```

Output:

```
abcd^e-fgh*+^*+i-

Process returned 0 (0x0)   execution time : 0.007 s
Press ENTER to continue.
```

Q. Program for Insertion and Deletion in Queue using Array

```
#include<bits/stdc++.h>


using namespace std;

int n=0;

void print(int *a,int front,int rear)
{
 cout<<"Queue:\n";
 for(int i=front+1;i<rear;i++)
 {
   cout<<a[i]<<" ";
 }
 cout<<endl;
}

int insert(int *a,int item,int rear)
{
 if(rear==n)
 {cout<<"queue full\n";exit(0);}
 a[rear]=item;
 rear=rear+1;
 return rear;
}

int deletion(int a[],int front,int rear)
{
 if (front+1==rear)
 {
  cout<<"queue is empty\n";
  exit(0);
 }
 front=front+1;
 return front;
}

int main()
{

 char ch='Y';
 cout<<"enter the limit of array:";
 cin>>n;
 int a[n];
 int front=-1, rear=0;
 while(ch=='Y'||'y')
 {
 int choice=-1;
 cout<<"1.Insert\n2.Delete\n";
 cin>>choice;
```

```
if(choice==1)
{
  cout<<"enter item to be inserted:";
  int item;
  cin>>item;
  rear=insert(a,item,rear);

}

else if(choice==2)
{
  front=deletion(a,front,rear);

}


else
{
 cout<<"Wrong choice\n";

}
print(a,front,rear);
cout<<"\nContinue??(Y or N):";
cin>>ch;
}



return 0;
}
```

Output:

```
enter the limit of array:20
1.Insert
2.Delete
1
enter item to be inserted:2
Queue:
2

Continue??(Y or N):Y
1.Insert
2.Delete
1
enter item to be inserted:5
Queue:
2 5

Continue??(Y or N):Y
1.Insert
2.Delete
2
Queue:
5

Continue??(Y or N):N
```

Q. Program for Tower of Hanoi problem

```
#include<bits/stdc++.h>
#include<math.h>

using namespace std;

int main()
{

 int n;
 cout<<"\nEnter the no. of discs:";
 cin>>n;
 cout<<"no. of shifts required:";
 cout<<pow(2,n)-1<<endl;
 return 0;
}
```

Output:

```
Enter the no. of discs:5
no. of shifts required:31

Process returned 0 (0x0)   execution time : 2.562 s
Press ENTER to continue.
```

Q. Program for postfix evaluation.

```
#include<bits/stdc++.h>

using namespace std;

int a[100]; int top=-1, N;

void push(int item)
{
 if(top==N-1)
   cout<<"\nThe stack is full";
 else
   {
    top=top+1;
    a[top]=item;
    cout<<"\nElement added successfully";
   }
}

int pop()
{
 int item;
 if(top==-1)
   cout<<"\nStack is empty!";
 else
   {
    item=a[top];
    top=top-1;
    return item;
   }
}

int main()
{
 string E;
 cout<<"\nEnter the postfix expression:";
 getline(cin, E);
 N=E.size();
 cout<<N;
 int i=0, result;
 cout<<E;
 char x; int a, b;
 x=E[i];
 while(i!=N)
 {cout<<x;
  if(x>='0'&&x<='9')
     push(x-48);
  else
    {
     switch(x)
     {case '+':a=pop();
```

```
            b=pop();
            result=a+b;
            cout<<"\nElement added!";
            cout<<result;
            push(result);
            break;
    case '-':a=pop();
            b=pop();
            result=(int)b-(int)a;
            push(result);
            break;
    case '*':a=pop();
            b=pop();
            result=(int)a*(int)b;
            push(result);
            break;
    case '/':a=pop();
            b=pop();
            result=(int)b/(int)a;
            push(result);
            break;
    }
   }
 i=i+1;
 x=E[i];
 }
 cout<<"\nAns:"<<pop;
 return 0;
}
```

Output:

```
Enter the postfix expression:53+2*697-/-
53+2*697-/-5
Element added successfully3
Element added successfully+
Element added!8
Element added successfully2
Element added successfully*
Element added successfully6
Element added successfully9
Element added successfully7
Element added successfully-
Element added successfully/
Element added successfully-
Element added successfully
Ans:13
Process returned 0 (0x0)   execution time : 16.295 s
Press ENTER to continue.
```

Q. Program to print the contents of a linked list

```
#include<iostream>
using namespace std;

struct node{
	int data;
	node* next;
};
node* head = NULL;
void insert(int x)
{
	node* temp = new node();
	temp->data = x;
	temp->next = NULL;

	// insert at beginning

	temp->next = head;
	head = temp;
}
void print()
{
	node* temp = head;
	while(temp!=NULL)
	{
	 cout<<temp->data<< " ";
	 temp = temp->next;
    }
	cout<<endl;
}
int main()
{
	insert(5);
	insert(4);
	insert(3);
	insert(2);
	insert(1);

	print();
}
```

Output:

```
1 2 3 4 5

Process returned 0 (0x0)   execution time : 0.001 s
Press ENTER to continue.
```

Q. Program to insert a node between two nodes in a linked list

```
#include<iostream>
using namespace std;

struct node{
	int data;
	node* next;
};
node* head = NULL;
insert(int x)
{
	node* temp = new node();
	temp->data = x;
	temp->next = NULL;

	// insert at beginning

	temp->next = head;
	head = temp;
}
print()
{
	node* temp = head;
	while(temp!=NULL)
	{
	 cout<<temp->data<< " ";
	 temp = temp->next;
    }
	cout<<endl;
}
int main()
{

	insert(5);
	insert(4);
	insert(3);
	insert(2);
	insert(1);
	insert(56);
	insert(12);
	insert(166);
	insert(163);
	insert(156);
	insert(99);

	print();

	int x,pos;
	cout<<"enter the position where you want to insert a new node\n";
	cin>>pos;
	cout<<"enter the value which you want to insert\n";
```

```
        cin>>x;

        int count =1;

        if(pos==1)
        insert(x);
        else
        {
                node* temp1 = head;
                while(count<pos-1)
                {
                        temp1 = temp1->next;
                        count++;
                }
                node* newval = new node();
                newval->data = x;
                newval -> next = NULL;
                newval ->next = temp1->next;
                temp1->next = newval;
        }


        print();
}
```

Output:

```
99 156 163 166 12 56 1 2 3 4 5
enter the position where you want to insert a new node
5
enter the value which you want to insert
45
99 156 163 166 45 12 56 1 2 3 4 5

Process returned 0 (0x0)   execution time : 17.579 s
Press ENTER to continue.
```

Q. Program to delete a node in linked list

```
#include<iostream>
using namespace std;

struct node{
	int data;
	node* next;
};
node* head = NULL;
void insert(int x)
{
	node* temp = new node();
	temp->data = x;
	temp->next = NULL;

	// insert at beginning

	temp->next = head;
	head = temp;
}
void print()
{
	node* temp = head;
	while(temp!=NULL)
	{
	 cout<<temp->data<< " ";
	 temp = temp->next;
    }
	cout<<endl;
}
void del(int pos)
{
	node* temp1 = head;
	int count=1;
	if(pos == 1)
	{
		head= head->next;
		delete temp1;
	}
	else
	{
		node* temp2;
		while(count<pos-1)
		{
			temp1 = temp1->next;
			count++;
		}
		temp2 = temp1->next;
		temp1->next = temp2->next;
		int x = temp2->data;
		delete temp2;
```

```
            cout <<"the value deleted is "<<x<<endl;


        }


}
int main()
{

        insert(5);
        insert(4);
        insert(3);
        insert(2);
        insert(1);
        insert(56);
        insert(12);
        insert(166);
        insert(163);
        insert(156);
        insert(99);

        print();

        int x,pos;
        cout<<"enter the position where you want to delete a new node\n";
        cin>>pos;

        del(pos);
        print();


}
```

Output:

```
99 156 163 166 12 56 1 2 3 4 5
enter the position where you want to delete a new node
4
the value deleted is 166
99 156 163 12 56 1 2 3 4 5

Process returned 0 (0x0)   execution time : 5.666 s
Press ENTER to continue.
```

Q. Polynomial addition using linked list

```
#include<bits/stdc++.h>

using namespace std;


struct Node
{
  int coeff;
  int exp;
  Node* next;
};


Node* push(Node* start,int c, int e)
{
	Node* ptr=start;
	if (start==NULL)
	{
		start=new Node;
		start->exp=e;
		start->coeff=c;
		//cout<<"start";
	}
	else
	{
		while(ptr->next!=NULL)
		{
			ptr=ptr->next;
		}
		Node* p=new Node;
		p->coeff=c;
		p->exp=e;
		ptr->next=p;
		//cout<<"extra";
	}
return start;
}


void print(Node* start)
{
	Node* ptr=start;
	while(ptr!=NULL)
	{
		cout<< ptr->coeff <<"*x^"<< ptr->exp <<" + ";
		ptr=ptr->next;
	}

	cout<<"0"<<endl;
}
```

```
int main()
{
	Node* start1=NULL;
	Node* start2=NULL;
	int c;
	int count=0;
	cout<<"enter the degree of polyomial 1: ";
	cin>>count;

	cout<<endl;
	cout<<"enter coefficients of polynomial 1:(starting from x^n)";
	for(int e=count;e>=0;e--)
	{
		cin>>c;
		if(c!=0)
			start1=push(start1,c,e);
		//cout<<c<<" "<<e<<endl;
	}
	print(start1);
	count=0;
	cout<<"enter the degree of polyomial 2: ";
	cin>>count;

	cout<<endl;
	cout<<"enter coefficients of polynomial 2:(starting from x^n)";
	for(int e=count;e>=0;e--)
	{
		cin>>c;
		if(c!=0)
			start2=push(start2,c,e);
		//cout<<c<<" "<<e<<endl;
	}
	print(start2);


Node* first=start1;
Node* second=start2;

Node* start3=NULL;

while(first!=NULL && second!=NULL)
{
	if (first->exp==second->exp)
	{
		start3=push(start3,first->coeff + second->coeff, first->exp);
		first=first->next;
		second=second->next;
	}
	else if (first->exp > second->exp)
	{
```

```
                start3=push(start3,first->coeff, first->exp);
                first=first->next;
        }
        else
        {
                start3=push(start3,second->coeff, second->exp);
                second=second->next;
        }

}

if(first==NULL)
{
        while(second!=NULL)
        {
                start3=push(start3,second->coeff, second->exp);
                second=second->next;
        }
}

else if (second==NULL)
{
        while(first!=NULL)
        {
                start3=push(start3,first->coeff, first->exp);
                first=first->next;
        }

}

print(start3);
}
```

Output:

```
enter the degree of polyomial 1: 2

enter coefficients of polynomial 1:(starting from x^n)2
5
1
2*x^2 + 5*x^1 + 1*x^0 + 0
enter the degree of polyomial 2: 1

enter coefficients of polynomial 2:(starting from x^n)6
2
5*x^1 + 2*x^0 + 0
2*x^2 + 11*x^1 + 3*x^0 + 0

Process returned 0 (0x0)   execution time : 61.110 s
Press ENTER to continue.
```

Q. Program to search element in linked list

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

struct Node
{
   int key;
   struct Node* next;
};

void push(struct Node** head_ref, int new_key)
{

   struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));

   new_node->key  = new_key;

   new_node->next = (*head_ref);

   (*head_ref)    = new_node;
}


bool search(struct Node* head, int x)
{
   struct Node* current = head;
   while (current != NULL)
   {
      if (current->key == x)
         return true;
      current = current->next;
   }
   return false;
}


int main()
{

   struct Node* head = NULL;
   int x = 21;


   push(&head, 10);
   push(&head, 30);
   push(&head, 11);
   push(&head, 21);
   push(&head, 14);

   int element;
```

```
    printf("\nEnter element to be searched:");
    scanf("%d", &element);
    search(head, element)? printf("Yes") : printf("No");
    return 0;
}
```

Output:

```
Enter element to be searched:11
Yes
Process returned 0 (0x0)   execution time : 2.530 s
Press ENTER to continue.
```

Q. Write a C++ program to for inorder traversal in trees.

Ans:

```
 #include<iostream>

#include<stack>

using namespace std;


struct node{

	int data;

	node* left;

	node* right;

};


node* newnode(int data)

{

	node* temp = new node();

	temp->data = data;

	temp->left = NULL;

	temp->right = NULL;

	return temp;

}


void inorder(node* root)

{

	stack <node*> s;

	node* current = root;


	while(!s.empty() || current!=NULL)

	{

		while(current!=NULL)
```

```
                {
                        s.push(current);
                        current = current->left;
                }

                current = s.top();
                s.pop();

                cout<<current->data<<" ";

                current = current->right;
        }

}
int main()
{
        node* root = newnode(1);
        root->left = newnode(2);
        root->right = newnode(3);
        root->left->left = newnode(4);
        root->left->right = newnode(5);
        root->left->right->left = newnode(6);
        root->left->right->right = newnode(7);

        inorder(root);

}
```

Output:

```
4 2 6 5 7 1 3
Process returned 0 (0x0)   execution time : 0.156 s
Press any key to continue.
```

Q. Write a C++ Program for preorder traversal.

Ans:

```
#include<bits/stdc++.h>

#include<stack>


using namespace std;


struct Node

{

        int info;

        Node* left;

        Node* right;


};




Node* push_node(Node* root,int item)

{

        if(root==NULL)

                {
```

```
                root=new Node;
                root->left=NULL;
                root->right=NULL;
                root->info=item;
                cout<<"Pushed"<<item<<endl;
            }

        else if(item <= root->info)
            {root->left=push_node(root->left,item);}
        else
            {root->right=push_node(root->right,item);}
        return root;
}

void preorder(Node* root)
{
        stack< Node* > Stack;
        Stack.push(root);

        while(!Stack.empty())
        {
            Node* temp=Stack.top();
            Stack.pop();

            cout<<temp->info<<" ";

            if(temp->right!=NULL)
                Stack.push(temp->right);
            if(temp->left!=NULL)
```

```
                    Stack.push(temp->left);


        }
}


void traverse(Node* ptr)
{
        cout<<ptr->info<<" ";
        if (ptr->left!=NULL)
                traverse(ptr->left);
        if (ptr->right!=NULL)
                traverse(ptr->right);
}



int main()
{
        Node* root=NULL;
        cout<<"enter the no. of tree elements:";
        int n;
        cin>>n;
        int temp;
        for(int i=0;i<n;i++)
        {
                cin>>temp;
                root=push_node(root,temp);
        }
        cout<<endl;
        preorder(root);
```

```
        return 0;
}
```

Output:

```
enter the no. of tree elements: 6
1 2 3 4 5 6
Pushed1
Pushed2
Pushed3
Pushed4
Pushed5
Pushed6

1 2 3 4 5 6
Process returned 0 (0x0)   execution time : 7.214 s
Press any key to continue.
```

Q. Write a C++ program for Postorder traversal.

Ans:

```
 #include<iostream>
#include<stack>
using namespace std;

struct node{
        int data;
        node* left;
        node* right;
};

node* newnode(int data)
{
        node* temp = new node();
```

```
        temp->data = data;
        temp->left = NULL;
        temp->right = NULL;
        return temp;
}
void postorder(node* root)
{
        stack <node*> s;
        node* p = root;

        while(!s.empty() || p!=NULL)
        {
                while(p!=NULL)
                {
                  if(p->right!=NULL)
                        s.push(p->right);
                        s.push(p);
                        p = p->left;
                }



                p = s.top();
                s.pop();

                if(s.empty())
                {
                 cout<<p->data<<" ";
                 break;
```

```
                }
                 node* temp = s.top();
                 s.pop();

                if(temp == p->right)
                {
                        s.push(p);
                        p = p->right;
                }
        else
                {
                        s.push(temp);
                        cout<<p->data<<" ";
                        p = NULL;
                }



        }
}

int main()
{
        node* root = newnode(1);
        root->left = newnode(2);
        root->right = newnode(3);
        root->left->left = newnode(4);
        root->left->right = newnode(5);
        root->left->right->left = newnode(6);
        root->left->right->right = newnode(7);
```

```
        postorder(root);

return 0;

}
```

Output:

```
4 6 7 5 2 3 1
Process returned 0 (0x0)   execution time : 0.172 s
Press any key to continue.
```

Q. Write a C++ program to implement Merge Sort.

Ans.

```
 #include<bits/stdc++.h>

 using namespace std;

void Merge(int arr[],int , int, int);

void MergeSort(int a[],int left, int right)

{

   int mid;

   if(left<right)

   {

      mid = left + (right-left)/2;

      MergeSort(a,left, mid);

      MergeSort(a,mid+1, right);

      Merge(a, left, mid, right);

   }

}

void Merge(int arr[], int left, int mid, int right)

{

   int sz = right - left + 1;

   int *temp = new int[sz];

   int temp_pos = 0;

   int left_end = mid;

   int start=left;

   int right_start=mid+1;


   while(left<=left_end && right_start <=right)

   {

      if(arr[left] < arr[right_start])

      {       temp[temp_pos] = arr[left];

         temp_pos++;
```

```
        left++;
      }
      else
      {
        temp[temp_pos]= arr[right_start];
        temp_pos++;
        right_start++;
      }

    }
    while(left <= left_end)
    {
      temp[temp_pos] = arr[left];
      left++;
      temp_pos++;
    }
    while(right_start <= right)
    {
      temp[temp_pos] = arr[right_start];
      temp_pos++;
      right_start++;
    }

    for(int i=0; i<sz; i++)
     {arr[start] = temp[i];
      start++;}
}

int main()
{
```

```
    int n;

    cin>>n;

    int *brr =new int[n];

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

        cin>>brr[i];

    MergeSort(brr, 0, n-1);


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

        cout<<brr[i]<<" ";


   return 0;


}
```

Output:

```
D:\Codes\Sorting\Merge_Sort.exe
1 6 2 9 4
1 2 4 6 9
Process returned 0 (0x0)   execution time : 20.473 s
Press any key to continue.
```

Q. Write a C++ program to implement Selection Sort.

Ans.

```
#include<bits/stdc++.h>

using namespace std;

void SelectionSort(int arr[], int n)

{

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

   {

      int mini = i;

      for(int j= i+1; j<n; j++)

      {

         if(arr[j]< arr[mini])

            mini=j;

       }

      int temp;

      temp = arr[i];

      arr[i] = arr[mini];

      arr[mini] = temp;

   }

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

   cout<<arr[i]<<" ";

}

int main()

{

   int n;

   cin>>n;

   int *arr= new int[n];

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

      cin>>arr[i];

   SelectionSort(arr, n);
```

```
    return 0;

}
```

Output:

```
D:\Codes\Sorting\Selection_Sort.exe
9
9 7 5 3 1 2 4 6 8
1 2 3 4 5 6 7 8 9
Process returned 0 (0x0)   execution time : 12.025 s
Press any key to continue.
```

Q. Write a C++ program to implement Quick Sort.

Ans.

```
#include<bits/stdc++.h>

using namespace std;

int Partition(int a[], int , int);

void QuickSort(int a[], int left, int right)

{

    int pivot;

    if(left<right)

    {

        pivot = Partition(a, left, right);

        QuickSort(a, left, pivot-1);

        QuickSort(a, pivot+1, right);

    }

}

int Partition(int arr[], int left, int right)

{  int pivot = arr[left];

    int start = left;

    int fin = right;

    int temp;
```

```
    while(start<fin)
    {
        while(arr[fin] > pivot)
            fin--;
        temp = arr[fin];
        arr[fin] = arr[start];
        arr[start] = temp;
        while(arr[start] <= pivot)
            start++;
        temp = arr[start];
        arr[start] = arr[fin];
        arr[left] = temp;
    }
    return start;
}



int main()
{
    int n;
    cin>>n;
    int *brr =new int[n];
    for(int i=0; i<n; i++)
        cin>>brr[i];
    QuickSort(brr, 0, n-1);


    for(int i=0; i<n; i++)
        cout<<brr[i]<<" ";
   return 0;
}
```

Output:

```
7
7 5 3 1 2 4 6
1 2 3 4 5 6 7
Process returned 0 (0x0)   execution time : 5.407 s
Press any key to continue.
```

Q. Write a C++ program to implement Heap Sort.

Ans:

```
#include<bits/stdc++.h>

using namespace std;


void heap(int arr[], int n, int i)

{

   int largest = i;

   int left= 2*i+1;

   int right = 2*i+2;


   if(left<n && arr[left]>arr[largest])

      largest=left;


   if(right<n && arr[right]>arr[largest])

      largest=right;


   if(largest!=i)

   {

      int temp = arr[i];

      arr[i]=arr[largest];
```

```
        arr[largest]=temp;


        heap(arr, n, largest);
    }
}


void heapsort(int arr[], int n)
{
    for(int k=n/2-1; k>=0; k--)
    {
        heap(arr, n, k);
    }


    for(int k=n-1; k>=0; k--)
    {
      int temp = arr[k];
        arr[k]=arr[0];
        arr[0]=temp;
      heap(arr, k, 0);
    }
}


void display(int arr[], int n)
{
    for(int i=0; i<n; i++)
        cout<<arr[i]<<" ";
}



int main()
```

```
{
   int n;
   cin>>n;
   int *a = new int[n];
   for(int i=0; i<n; i++)
      cin>>a[i];

  heapsort(a, n);

  display(a,n);
  return 0;
}
```

Output:

```
6
1 3 5 6 4 2
1 2 3 4 5 6
Process returned 0 (0x0)   execution time : 6.632 s
Press any key to continue.
```

Program for Breadth First Search in a graph

```
#include<bits/stdc++.h>
#include<queue>
using namespace std;
int adj[5][5]={0};
int visited[5]={0};
queue<int> Q;
int main()
{
	int start=0;
	adj[0][1]=1;
	adj[1][0]=1;
	adj[0][2]=1;
	adj[2][0]=1;
	adj[1][3]=1;
	adj[3][1]=1;
	adj[2][3]=1;
	adj[3][2]=1;
	adj[3][4]=1;
	adj[4][3]=1;
	Q.push(start);
	visited[start]=1;
	int i;
	while(!Q.empty())
	{
		i=Q.front();
		Q.pop();
		cout<<i<<endl;
		for(int j=0;j<5;j++)
		{
			if(adj[i][j]==1 and visited[j]==0)
			{
				Q.push(j);
				visited[j]=1;
			}
		}
	}

	return 0;
}
```

```
0
1
2
3
4

Process returned 0 (0x0)   execution time : 0.002 s
Press ENTER to continue.
```

Program for Depth First Search in a graph

```
#include<bits/stdc++.h>
#include<stack>
using namespace std;
int adj[5][5]={0};
int visited[5]={0};
stack<int> S;
int main()
{
	int start=0;
	adj[0][1]=1;
	adj[1][0]=1;
	adj[0][2]=1;
	adj[2][0]=1;
	adj[1][3]=1;
	adj[3][1]=1;
	adj[2][3]=1;
	adj[3][2]=1;
	adj[3][4]=1;
	adj[4][3]=1;
	S.push(start);
	visited[start]=1;
	int i;
	cout<<start<<endl;
	while(!S.empty())
	{
		i=S.top();
		int flag=0;
		for(int j=0;j<5;j++)
		{
			if (adj[i][j]==1 and visited[j]==0 )
			{
				S.push(j);
				visited[j]=1;
				cout<<j<<endl;
				flag=1;
				break;
			}
		}
		if(flag==0)
		{
			S.pop();
		}
	}
}
```

```
0
1
3
2
4

Process returned 0 (0x0)   execution time : 0.001 s
Press ENTER to continue.
```

Program to implement Dijkstra's algorithm to find shortest path in a graph

```
#include<iostream>
#include<limits.h>
using namespace std;
int v = 0;
int mindist(int d[], bool dset[])
{
	int minimum=INT_MAX,index;
	for(int i=0;i<v;i++)
	{
		if(dset[i]==false && d[i]<=minimum)
		{
			minimum=d[i];
			index=i;
		}
	}
	return index;
}
int main()
{
	cout<<"Enter total no. of vertex"<<endl;
	cin>>v;
	int g[v][v];
	int d[v];
	bool dset[v];
	for(int i=0;i<v;i++)
	  for(int j=0;j<v;j++)
	    cin>>g[i][j];
	for(int i=0;i<v;i++)
	{
		d[i]=INT_MAX;
		dset[i]=false;
	}
	d[0]=0;
	for(int i=0;i<v;i++)
	{
		int u=mindist(d,dset);
		dset[u]=true;
		for(int j=0;j<v;j++)

		{
			if(dset[j]==false && g[u][j]!=0 && d[u]!=INT_MAX && d[u]+g[u][j]<d[j])
			d[j]=d[u]+g[u][j];
		}
	}
	cout<<"Vertex\t\tDistance from source"<<endl;
	for(int i=0;i<v;i++)
	 cout<<i+1<<"\t\t"<<d[i]<<endl;

	return 0;
```

}

```
Enter total no. of vertex
7
0 5 3 0 0 0 0
0 0 2 0 3 0 1
0 0 0 7 7 0 0
2 0 0 0 0 6 0
0 0 0 2 0 1 0
0 0 0 0 0 0 0
0 0 0 0 1 0 0
Vertex          Distance from source
1               0
2               5
3               3
4               9
5               7
6               8
7               6

Process returned 0 (0x0)   execution time : 112.823 s
Press ENTER to continue.
```

Program to find all pair shortest path in a graph using Warshall's algorithm.

```
#include<iostream>
using namespace std;
int  main()
{
    int v;
    cin>>v;
    int g[v][v],n,i,j,k;
    cout<<"Enter adjacency matrix \n";
    for(i=0;i<v;i++)
        for(j=0;j<v;j++)
            cin>>g[i][j];
    for(k=0;k<v;k++)
    {
        for(i=0;i<v;i++)
        {
            if (g[i][k]>0)
            {
                for(j=0;j<n;j++)
                    {
                        if(g[k][j]>0 && g[i][j]>(g[i][k]+g[k][j]))
                        {
                            g[i][j]=g[i][k]+g[k][j];
                        }
                    }
            }
        }
    }
    cout<<"\nOutput : \n";
    for(i=0;i<v;i++)
    {
        for(j=0;j<v;j++)
        {
            cout<<g[i][j]<<" ";
        }
        cout<<"\n";
    }
}
```

```
7
Enter adjacency matrix
0 5 3 0 0 0 0
0 0 2 0 3 0 1
0 0 0 7 7 0 0
2 0 0 0 0 6 0
0 0 0 2 0 1 0
0 0 0 0 0 0 0
0 0 0 0 1 0 0

Output :
0 5 3 0 0 0 0
0 0 2 0 3 0 1
0 0 0 7 7 0 0
2 0 0 0 0 6 0
0 0 0 2 0 1 0
0 0 0 0 0 0 0
0 0 0 0 1 0 0

Process returned 0 (0x0)   execution time : 54.909 s
Press ENTER to continue.
```