Practical No. 02

1. Searching Techniques:-
   1. Linear Search
   2. Binary Search

Code:-

#include<iostream>

#define max 99

using namespace std;

int n;

int A[max];

class LinearSearch

{

int temp,count;

public:

LinearSearch()

{

count = 0;

temp = 0;

}

void getdata()

{

cout<<"Enter the size of the Array"<<endl;

cin>>n;

cout<<"Enter the values of the Array"<<endl;

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

{

cin>>A[i];

}

}

void display()

{

cout<<endl;

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

{

cout<<A[i]<<"\t";

}

cout<<endl;

}

void search(int x)

{

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

{

if(x == A[i])

{

count++;

cout<<"\nElement has been found at position = "<<i+1<<endl;

display();

break;

}

}

if(count == 0)

{

cout<<"\nElement Not Found"<<endl;

A[n]=x;

cout<<endl;

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

{

cout<<A[i]<<"\t";

}

cout<<endl;

n++;

}

}

void sort()

{

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

{

count = 0;

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

{

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

{

temp = A[j];

A[j] = A[j+1];

A[j+1] = temp;

count++;

}

}

if(count == 0)

{

break;

}

}

display();

}

int bsearch(int x)

{

int low = 0;

int high = n;

while(low<=high)

{

int mid=(low+high)/2;

if(A[mid]<x)

{

low=mid+1;

}

else if(A[mid]>x)

{

high=mid-1;

}

else

{

return mid;

}

}

return -1;

}

};

int main()

{

LinearSearch ls;

int s,c=0,k=-1;

ls.getdata();

ls.display();

for( ; ; ){

cout<<"Options :\n";

cout<<"\nEnter 1 to Sort.\nEnter 2 for a Linear Search.\nEnter 3 for a Binary Search.\nEnter 4 to Display Array.\nEnter 9 to Exit.\n";

cin>>c;

if(c==9)

{

cout<<"Bye!!!"<<endl;

break;

}

switch(c)

{

case 2:

cout<<"Enter Any Integer Number To Search If It's There You'll Get Results Otherwise It'll be Added To The Array\n";

cin>>s;

ls.search(s);

break;

case 1:

ls.sort();

break;

case 3:

cout<<"Enter Any Integer Number To Search\n"<<endl;

cin>>s;

ls.sort();

k = ls.bsearch(s);

if(k!=-1)

{

cout<<"Element found at position "<<k+1<<endl;

}

else

{

cout<<"Element not found"<<endl;

}

break;

case 4:

ls.display();

break;

default:

cout<<"Invalid Number"<<endl;

break;

}

}

return 0;

}

Snapshot:-