**15B17CI371 – Data Structures Lab**

**ODD 2024**

**Week 5-LAB A**

**Practice Lab**

**[CO: C270.2, C270.3]**

**Instructions:**

**1. All students must save all their programs with the nomenclature**

**(Enroll No\_W5\_LabA\_QuestionNo.cpp). Also store the Outputs as well.**

**2. Upload them as per the instructions given by your lab faculty.**

**Concepts: Sorting & Searching and their time complexity**

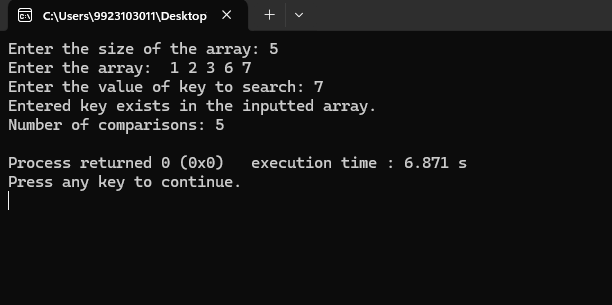
**Lab Questions:**

**Find the time and resource complexity of the questions. Also find the number of**

**swaps and comparisons required for each case.**

**1. Write a program using linear search to check whether the inputted element belong to**

**the it or not.**

****

#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int main()

{

int n;

cout<<"Enter the size of the array: ";

cin>>n;

int \*arr=new int[n];

cout<<"Enter the array: ";

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

{

cin>>arr[i];

}

int key,count=0;

cout<<"Enter the value of key to search: ";

cin>>key;

bool exist=false;

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

{

if(arr[i]==key)

{

exist=true;

cout<<"Entered key exists in the inputted array."<<endl;

count++;

break;

}

count++;

}

if(!exist)

{

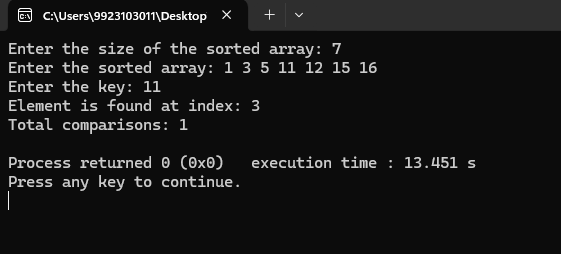
cout<<"Entered Element does not exist in the list"<<endl;

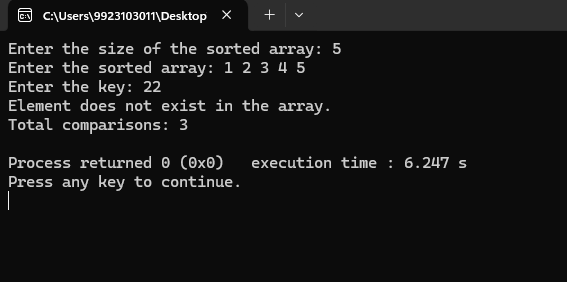
}

cout<<"Number of comparisons: "<<count<<endl;;

}

**2. Implement the binary search using iterative method.**

****

****

void bsearch(int arr[],int n)

{

int low=0;

int high=n-1;

int count=0,k;

cout<<"Enter the key: ";

cin>>k;

while(low<=high)

{

int mid=(low+high)/2;

if(k==arr[mid])

{

count++;

cout<<"Element is found at index: "<<mid<<endl;

cout<<"Total comparisons: "<<count<<endl;

exit(0);

}

else if(arr[mid]>k)

{

high=mid-1;

count++;

}

else

{

low=mid+1;

count++;

}

}

cout<<"Element does not exist in the array."<<endl;

cout<<"Total comparisons: "<<count<<endl;

}

int main()

{

int n;

cout<<"Enter the size of the sorted array: ";

cin>>n;

int \*arr=new int[n];

cout<<"Enter the sorted array: ";

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

{

cin>>arr[i];

}

bsearch(arr,n);

}

**3. Write a function to find kth smallest /largest element in unsorted array.**

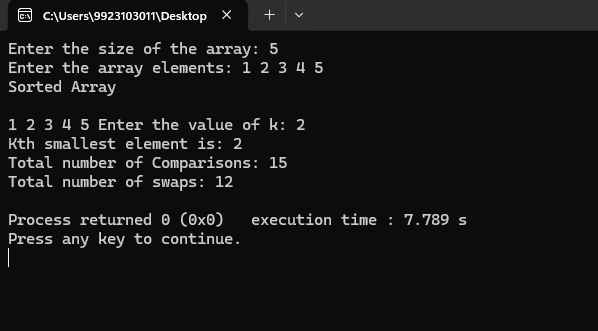
**Examples:**

**Input: arr[ ] = {7, 10, 4, 3, 20, 15}, k = 3**

**Output: 7**

**Input: arr[ ] = {7, 10, 4, 3, 20, 15}, k = 4**

**Output: 10**

****

class q3

{

public:

int swaps,count=0;

int partition(int arr[], int low, int high) {

int pivot = arr[high];

int i = low - 1;

for (int j = low; j <= high - 1; j++) {

if (arr[j] < pivot) {

i++;

swap(arr[i], arr[j]);

swaps++;count++;

}

}

swap(arr[i + 1], arr[high]);

swaps++;

return i + 1;

}

void quickSort(int arr[], int low, int high) {

if (low < high) {

count++;

int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

void kthsmallest(int arr[],int n)

{

map<int,int> m;

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

{

m[arr[i]]++;

}

int k;

cout<<"Enter the value of k: ";

cin>>k;

int s=1;

map<int,int>::iterator it=m.begin();

while(it!=m.end())

{

if((k==s) &&(it->second==1))

{

count++;

cout<<"Kth smallest element is: "<<(it->first)<<endl;

}

s++;

it++;

}

}

};

int main() {

int n;

cout<<"Enter the size of the array: ";

cin>>n;

int \*arr=new int[n];

cout<<"Enter the array elements: ";

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

cin>>arr[i];

}

q3 obj;

obj.quickSort(arr, 0, n - 1);

cout << "Sorted Array"<<endl<<endl;

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

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

}

obj.kthsmallest(arr,n);

cout<<"Total number of Comparisons: "<<obj.count<<endl;

cout<<"Total number of swaps: "<<obj.swaps<<endl;

}

4. Given a sorted array of size N and an integer K, find the position at which K is

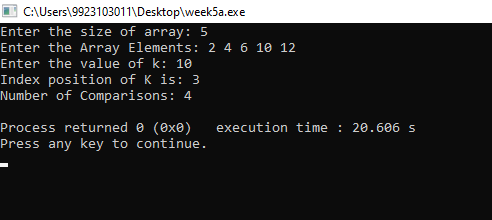
present in the array using interpolation search.

Example:

Input: N = 5, arr[ ] = {1 2 3 4 5}, K = 4

Output: 3

Explanation: 4 appears at index 3.



#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int count=0;

int interpolation(int arr[],int n)

{

int k;

cout<<"Enter the value of k: ";

cin>>k;

int low=0;

int high=n-1;

while(low<=high)

{

int mid=low+(k-arr[low])\*((high-low)/(arr[high]-arr[low]));

if(arr[mid]==k)

{

count++;

return mid;

}

else if(arr[mid]>k)

{

count++;

high=mid-1;

}

else

{

count++;

low=mid+1;

}

}

return -1;

}

int main()

{

int n;

cout<<"Enter the size of array: ";

cin>>n;

int \* arr=new int[n];

cout<<"Enter the Array Elements: ";

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

{

cin>>arr[i];

}

int res=interpolation(arr,n);

if(res==-1)

{

cout<<"Element K does not exist in the list."<<endl;

}

else

{

cout<<"Index position of K is: "<<res<<endl;

}

cout<<"Number of Comparisons: "<<count<<endl;

}

5. Given a sorted array of Strings and a String x, find an index of x if it is present in the

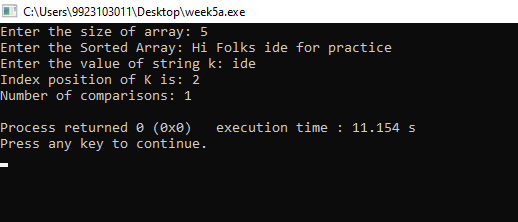
array.

Examples:

Input : arr[] = {”Hi”, ”Folks”, ”ide”, ”for”, ”practice”}, x = ”ide”

Output : 2, The String x is present at index 2.

Input : arr[ ] = {”Hi”, ”Folks”, ”ide”, ”for”, ”practic”}, x = ”zz”



class q5

{

public:

int count=0;

int bsearch(string arr[],int n)

{

string k;

cout<<"Enter the value of string k: ";

cin>>k;

int low=0;

int high=n-1;

while(low<=high)

{

int mid=(low+high)/2;

if(arr[mid]==k)

{

count++;

return mid;

}

else if(arr[mid]>k)

{

count++;

high=mid-1;

}

else

{

count++;

low=mid+1;

}

}

return -1;

}

};

int main()

{

int n;

cout<<"Enter the size of array: ";

cin>>n;

string \* arr=new string[n];

cout<<"Enter the Sorted Array: ";

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

{

cin>>arr[i];

}

q5 obj;

int res=obj.bsearch(arr,n);

if(res==-1)

{

cout<<"Element K does not exist in the array."<<endl;

}

else

{

cout<<"Index position of K is: "<<res<<endl;

}

cout<<"Number of comparisons: "<<obj.count<<endl;

}