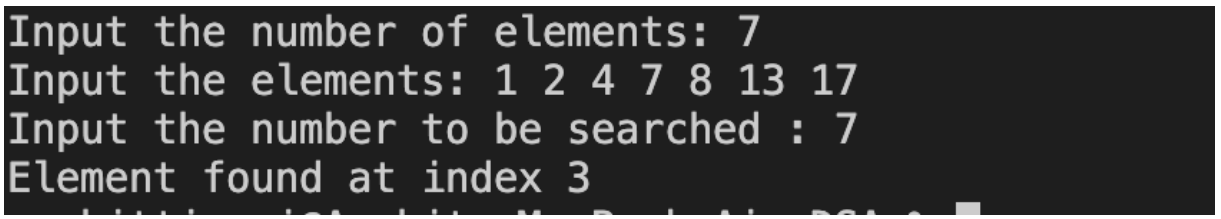
15B17CI371 – Data Structures Lab
ODD 2024
Week 5-LAB A

15B17CI371 – Data Structures Lab
ODD 2024
Week 5-LAB A
Practice Lab

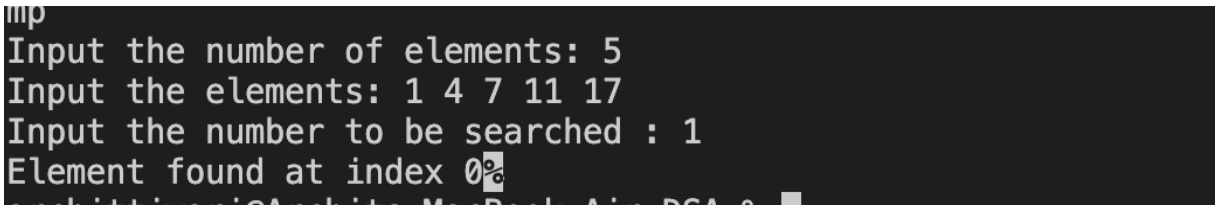
1. Write a program using linear search to check whether the inputted element belong to the it or not.

```
#include <iostream>
using namespace std;
int main()
{
   int n;
   cout<<"Input the number of elements: ";
   cin>>n;
   int *arr=new int[n];
   cout<<"Input the elements: ";
   for(int i=0;i<n;i++)
      cin>>arr[i];
   int key;
   cout<<"Input the number to be searched : ";
   cin>>key;
   for(int i=0;i<n;i++)
      if(arr[i]==key)
         cout<<"Element found at index "<<i<<endl;
}
```



2. Implement the binary search using iterative method.

```
#include <iostream>
using namespace std;
int main()
{
   int n;
   cout<<"Input the number of elements: ";
   cin>>n;
   int *arr=new int[n];
   cout<<"Input the elements: ";
   for(int i=0;i<n;i++)
      cin>>arr[i];
   int key;
   cout<<"Input the number to be searched : ";
   cin>>key;
   int start=0,end=n-1,mid;
   while(start!=end)
   {
      mid=start+(end-start)/2;
      if(arr[mid]==key)
         break;
      else if(arr[mid]<key)
         start=mid+1;
      else
         end=mid-1;
   }
   if(arr[mid]==key)
      cout<<"Element found at index "<<mid;
   else
      cout<<"Element not found ";
}
```



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

```
#include <iostream>

using namespace std;


void swap(int& a, int& b) {

   int temp = a;

   a = b;

   b = temp;

}


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

   int pivot = arr[high];

   int i = low - 1;


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

      if (arr[j] < pivot) {

         ++i;
```

```
            swap(arr[i], arr[j]);
        }
    }
    swap(arr[i + 1], arr[high]);
    return i + 1;
}

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

        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

void printarray( int arr[], int size) {
    for (int i = 0; i < size; ++i) {
        cout << arr[i] << " ";
    }
    cout << endl;
}
```

```
int smallest(int arr[],int k)
{
   return arr[k-1];
}

int greatest(int arr[],int k,int n)
{
   return arr[n-k];
}
int main() {
int size;
cout << "Enter the number of elements: ";
cin >> size;
int arr[size];
cout << "Enter the elements: ";
for (int i = 0; i < size; ++i) {
cin >> arr[i];
}
int key;
cout << "Enter which smallest and greatest element u want to search for: ";
cin >> key;
   cout << "Original array: ";
```

```
    printarray(arr, size);

    quickSort(arr, 0, size - 1);

    cout << "Sorted array: ";

    printarray(arr, size);

    int x=smallest(arr,key);

    cout<<key<<" th smallest element is "<<x<<endl;

    int y=greatest(arr,key,size);

    cout<<key<<" th greatest element is "<<y;

    return 0;

}
```

```
Enter the number of elements: 5
Enter the elements: 11 7 4 15 10
Enter which smallest and greatest element u want to search for: 2
Original array: 11 7 4 15 10
Sorted array: 4 7 10 11 15
2 th smallest element is 7
2 th greatest element is 11%
```

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.

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cout<<"Input the number of elements: ";
    cin>>n;
    int *arr=new int[n];
    cout<<"Input the elements: ";
```

```
   for(int i=0;i<n;i++)
      cin>>arr[i];
   int key;
   cout<<"Input the number to be searched : ";
   cin>>key;
   int start=0,end=n-1,pos;
   while(start!=end)
   {
      pos=start+((key-arr[start])*(end-start)/(arr[end]-arr[start]));
      if(arr[pos]==key)
         break;
      else if(arr[pos]<key)
         start=pos+1;
      else
         end=pos-1;
   }
   if(arr[pos]==key)
      cout<<"Element found at index "<<pos;
   else
      cout<<"Element not found ";
}
```

```
Input the number of elements: 7
Input the elements: 7 11 17 22 36 44 51
Input the number to be searched : 7
Element found at index 0%
```

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”

```
#include <iostream>
#include <string>
using namespace std;
int binarySearch(const string arr[], int size, const string& x) {
    int left = 0;
    int right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == x) {
            return mid;
        } else if (arr[mid] < x) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }

    return -1;
}

int main() {
    string arr1[] = {"Hiiiiiiiiii", "Folks", "ide", "for", "practice"};
    int size1 = sizeof(arr1) / sizeof(arr1[0]);
    cout<<"Size of array:"<<size1<<endl;
    string x1 = "ide";
    int index1 = binarySearch(arr1, size1, x1);
    cout << "Index of '" << x1 << "': " << index1 << endl;

    string arr2[] = {"Hiiiiiiiiii", "Folks", "ide", "for", "practic"};
    int size2 = sizeof(arr2) / sizeof(arr2[0]);
        cout<<"Size of array:"<<size2<<endl;
    string x2 = "zz";
    int index2 = binarySearch(arr2, size2, x2);
    cout << "Index of '" << x2 << "': " << index2 << endl;

    return 0;
}
```

```
Size of array:5
Index of 'ide': 2
Size of array:5
Index of 'zz': -1
architiwari@Archits-MacBook-Air DSA %
```