15B17CI371 – Data Structures Lab

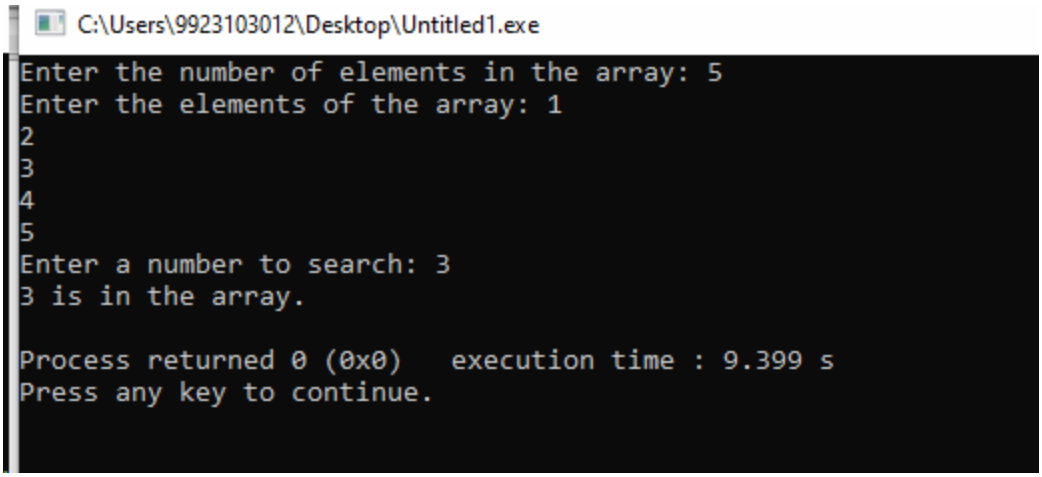
ODD 2024

Week 5-LAB A

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

```
#include <iostream>

using namespace std;



bool findElem(const int arr[], int size, int target) {
   for (int i = 0; i < size; ++i) {
      if (arr[i] == target) {
         return true;
      }
   }
   return false;
}

int main() {
   int size;

   cout << "Enter the number of elements in the array: ";
   cin >> size;
```

```
    int* numbers = new int[size];

    cout << "Enter the elements of the array: ";
    for (int i = 0; i < size; ++i) {
        cin >> numbers[i];
    }

    int num;
    cout << "Enter a number to search: ";
    cin >> num;

    if (findElem(numbers, size, num)) {
        cout << num << " is in the array." << endl;
    } else {
        cout << num << " is not in the array." << endl;
    }

    delete[] numbers;

    return 0;
}
```



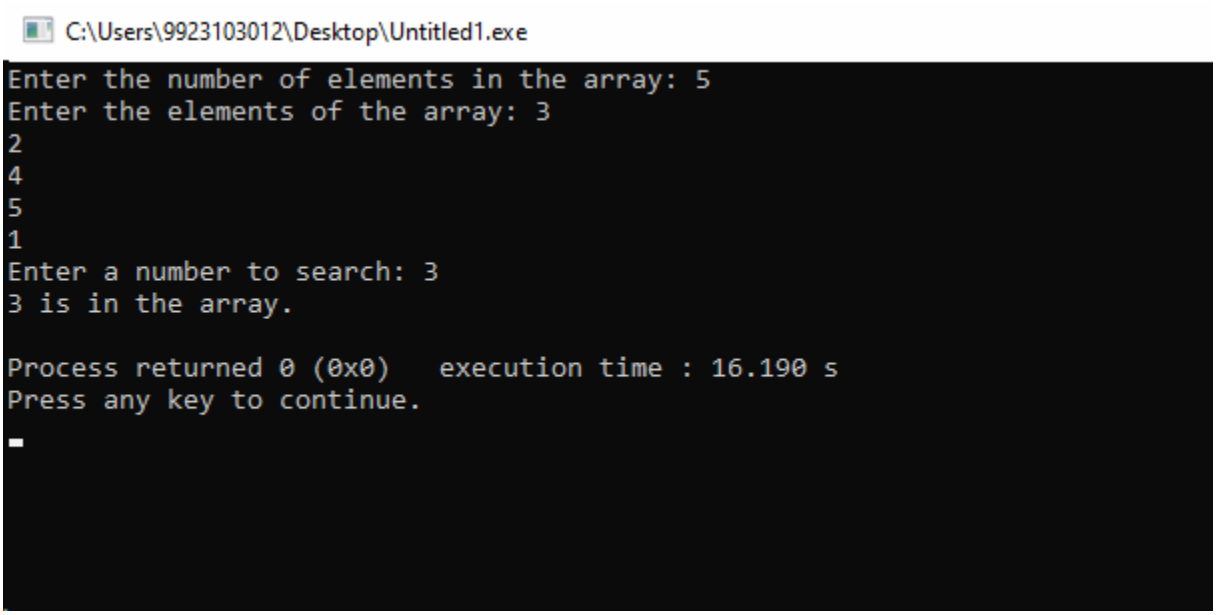
2. Implement the binary search using iterative method.

```
#include <iostream>

using namespace std;



void sortArr(int arr[], int size) {
   bool swapped;
   for (int i = 0; i < size - 1; ++i) {
      swapped = false;
      for (int j = 0; j < size - i - 1; ++j) {
         if (arr[j] > arr[j + 1]) {
            swap(arr[j], arr[j + 1]);
            swapped = true;
         }
```

```
        }
        if (!swapped) break;
    }
}

bool binSearch(const int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (arr[mid] == target) {
            return true;
        }
        if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return false;
}

int main() {
```

```
int size;

cout << "Enter the number of elements in the array: ";
cin >> size;

int* numbers = new int[size];



cout << "Enter the elements of the array: ";
for (int i = 0; i < size; ++i) {
    cin >> numbers[i];
}



sortArr(numbers, size);

int num;
cout << "Enter a number to search: ";
cin >> num;

if (binSearch(numbers, size, num)) {
    cout << num << " is in the array." << endl;
} else {
    cout << num << " is not in the array." << endl;
}
```

```
    delete[] numbers;


    return 0;
}
```



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

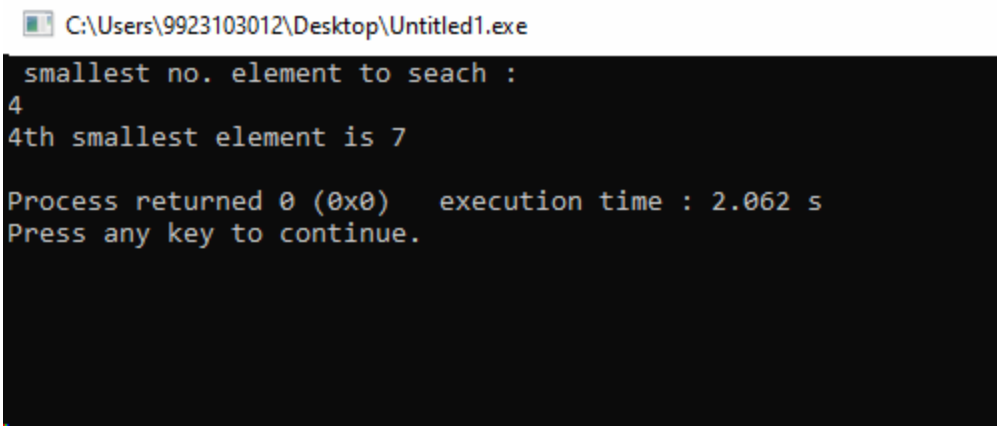
Input: arr[ ] = {7, 10, 4, 3, 20, 15}, k = 4

Output: 10

```
#include <iostream>

#include <cstdlib>

using namespace std;
```

```
int median(int A[], int l, int r, int k) {
    int ind = r - l + 1;
    int index = l + rand() % ind;
    int i, j = 0, m = 0, n = 0;
    int S1[10], S3[10], S2[1];
    for (i = l; i <= r; i++) {
        if (A[i] < A[index]) {
            if (j < 10) {
                S1[j] = A[i];
                j++;
            }
        } else if (A[i] == A[index]) {
            if (n < 1) {
                S2[n] = A[i];
                n++;
            }
        } else {
            if (m < 10) {
                S3[m] = A[i];
                m++;
            }
        }
    }
```

```
    if (j >= k) {
        return median(S1, 0, j - 1, k);
    } else if ((j + n) >= k) {
        return A[index];
    } else {
        return median(S3, 0, m - 1, k - (j + n));
    }
}

int main() {
    int arr[] = {7, 10, 4, 3, 20, 15, 8, 12, 6};
    int n = sizeof(arr) / sizeof(arr[0]);
    int k ;
    cout<< " smallest no. element to seach : "<< endl;
    cin>>k;

    int element = median(arr, 0, n - 1, k);

    cout << k << "th smallest element is " << element << endl;

    return 0;
}
```



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

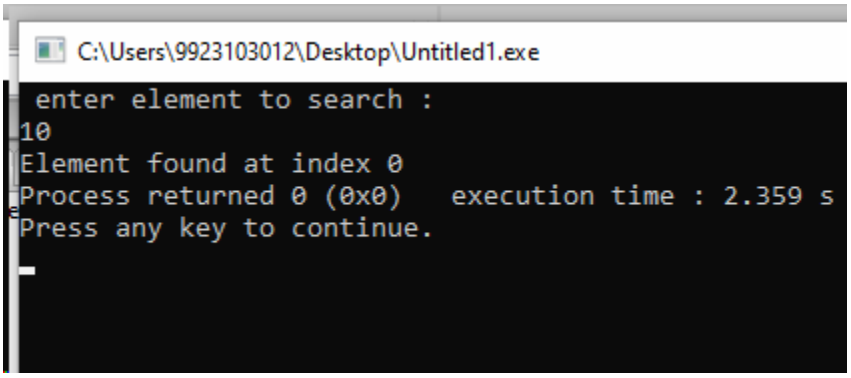
Output: 3

Explanation: 4 appears at index 3.

```
#include <iostream>

using namespace std;

int interpolationSearch(int arr[], int lo, int hi, int x) {
   int pos;

   if (lo <= hi && x >= arr[lo] && x <= arr[hi]) {
      pos = lo + (((double)(hi - lo) / (arr[hi] - arr[lo])) * (x - arr[lo]));

      if (arr[pos] == x)
```

```
            return pos;



        if (arr[pos] < x)
            return interpolationSearch(arr, pos + 1, hi, x);



        if (arr[pos] > x)
            return interpolationSearch(arr, lo, pos - 1, x);
    }

    return -1;
}



int main() {
    int arr[] = {10, 12, 13, 16, 18, 19, 20, 21, 22, 23, 24, 33, 35, 42, 47};
    int n = sizeof(arr) / sizeof(arr[0]);



    int x ;
    cout << " enter element to search : "<<endl;
    cin >>x;

    int index = interpolationSearch(arr, 0, n - 1, x);
```

```
    if (index != -1)
        cout << "Element found at index " << index;
    else
        cout << "Element not found.";

    return 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”

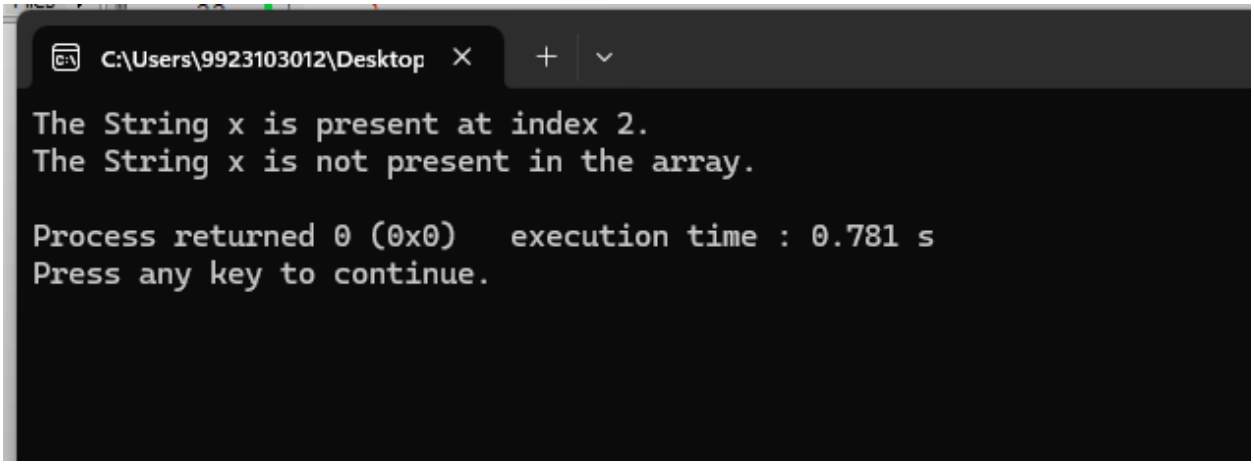
Output : 2, The String x is present at index 2.

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 low = 0;
    int high = size - 1;

    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == x) {
            return mid; // x found at index mid
        }
        if (arr[mid] < x) {
            low = mid + 1; // Search in the right half
        } else {
            high = mid - 1; // Search in the left half
        }
    }

    return -1; // x not found in the array
}

int main() {
```

```
    const int size1 = 5;
    string arr1[size1] = {"Hi", "Folks", "ide", "for", "practice"};
    string x1 = "ide";
    int index1 = binarySearch(arr1, size1, x1);

    if (index1 != -1) {
        cout << "The String x is present at index " << index1 << ".\n";
    } else {
        cout << "The String x is not present in the array.\n";
    }

    const int size2 = 5;
    string arr2[size2] = {"Hi", "Folks", "ide", "for", "practic"};
    string x2 = "zz";
    int index2 = binarySearch(arr2, size2, x2);

    if (index2 != -1) {
        cout << "The String x is present at index " << index2 << ".\n";
    } else {
        cout << "The String x is not present in the array.\n";
    }

    return 0;
}
```


C:\Users\9923103012\Desktop
The String x is present at index 2.
The String x is not present in the array.
Process returned 0 (0x0)   execution time : 0.781 s
Press any key to continue.