PAK-AUSTRIA FACHHOCHSCHULE: INSTITUTE OF APPLIED SCIENCES AND TECHNOLOGY

<u>Department of IT & CS</u>

Course Instructor: ______________________ Dated: ___02, NOV, 2023__

Semester: Fall 2023

COMP-201L

Lab 02: C++ Review

		CLO1	CLO2	CLO3	
Name	**Reg. No.**	**Lab Tasks Marks 20**	**Report Marks 5**	**Viva Marks 5**	**Total Marks 30**
JAWAD UL ISLAM KHAN	**B22F1235SE109**				

Lab Task 1

Write a program to find out a number among all other numbers entered by user using Binary search technique and Linear Search Technique.

Lab Task 2

Find Sum of Fibonacci Series using Recursive Function.

Program:

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

int fibonacci(int n)
{
   if (n <= 1)
   {
      return n;
   }
   return fibonacci(n - 1) + fibonacci(n - 2);
}

int sum_Of_Fibonacci(int n)
{
   int sum = 0;
   for (int i = 0; i < n; i++)
   {
      sum = sum +  fibonacci(i);
   }
   return sum;
}

int main()
{
   int n;
   cout << "Enter the number of terms in the Fibonacci series: ";
   cin >> n;

   int sum = sum_Of_Fibonacci(n);
```

```
    cout << "Sum of the first " << n << " Fibonacci numbers is: " << sum << endl;

    return 0;
}
```

Lab Task 3

Given a sorted array of integers, find index of first or last occurrence of a given number. If the element is not found in the array, report that as well.

Program:

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

int find_First_Occurrence(int arr[], int size, int target)
{
    int left = 0;
    int right = size - 1;
    int result = -1;

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

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

    return result;
}
```

```
int find_Last_Occurrence(int arr[], int size, int target)
{
   int left = 0;
   int right = size - 1;
   int result = -1;

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

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

   return result;
}

int main()
{
   int arr[] = {1, 2, 2, 4, 4, 4, 5, 6, 7};
   int size = sizeof(arr) / sizeof(arr[0]);
   int target;

   cout << "Enter the number to search: ";
   cin >> target;

   int first_Occurrence = find_First_Occurrence(arr, size, target);
   int last_Occurrence = find_Last_Occurrence(arr, size, target);

   if (first_Occurrence != -1)
   {
      cout << "First occurrence of " << target << " is at index: " << first_Occurrence << endl;
      cout << "Last occurrence of " << target << " is at index: " << last_Occurrence << endl;
```

```
    }
    else
    {
        cout << target << " is not found in the array." << endl;
    }

    return 0;
}
```

Lab Task 4

Given a circularly sorted array of integers, find the number of times the array is rotated. Assume there are no duplicates in the array and the rotation is in clockwise direction.

Input: arr = [9, 10, 2, 5, 6, 8]

Output: The array is rotated 2 times

Program:

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

int find_Rotations(int arr[], int size)
{
    int left = 0;
    int right = size - 1;

    while (left <= right)
    {
        if (arr[left] <= arr[right])
        {
            return left;
        }

        int mid = left + (right - left) / 2;
        int next = (mid + 1) % size;
        int prev = (mid - 1 + size) % size;

        if (arr[mid] <= arr[next] && arr[mid] <= arr[prev])
        {
            return mid;
        }
       else if (arr[mid] <= arr[right])
```

```
        {
            right = mid - 1;
        }
        else if (arr[mid] >= arr[left])
        {
            left = mid + 1;
        }
    }

    return -1;
}

int main()
{
    int arr[] = {9, 10, 2, 5, 6, 8};
    int size = sizeof(arr) / sizeof(arr[0]);

    int rotations = find_Rotations(arr, size);

    if (rotations >= 0)
    {
        cout << "The array is rotated " << rotations << " times in a clockwise direction." << endl;
    }
    else
    {
        cout << "The array is not rotated." << endl;
    }

    return 0;
}
```