**National University of Computer and Emerging Sciences**



**Laboratory Manual**

*for*

**Object Oriented Programing Lab**

|  |  |
| --- | --- |
| Course Instructor | Mr. Uzair Naqvi |
| Lab Instructor(s) | M Hashir, Seemab Ayub |
| Section | BCS-2B |
| Date | Tuesday, 30 Jan 2024 |
| Semester | Spring 2024 |

**Department of Computer Science**

FAST-NU, Lahore, Pakistan

**Objectives:**

In this lab, students will practice:

1. Revision of searching and sorting
2. Pointers

# Exercise- **Linear Search**

Write a program that performs a linear search on an array of integers. The program should take user input for the array elements and the target value to search. Display whether the target value is found in the array or not.

# Exercise- **Binary Search**

Implement a program that performs binary search on a sorted array of integers. The program should take user input for the array elements, sort the array, and then ask the user for a target value to search. Display whether the target value is found in the array or not.

# Exercise- **Sorting**

Write a program to perform Bubble sort, Selection Sort and Insertion Sort algorithm on an array of integers. The program should take user input for the array elements, perform the sortings, and display the sorted array.

# Exercise No. 1 – Basic Pointers

Follow these steps:

1. Declare:
   1. Int variables num1, num2 and sum
   2. Int\* pointer variables xPtr, yPtr and sumPtr
2. Set num1, num2 and sum to 5, 7 and 0 respectively
3. Initialize all pointers to 0 (nullptr)
4. Print values of variables num1, num2 with labels as shown in the required output below:

**Required Output:**

Num1 = 5

Num2 = 7

1. Print the addresses of Num1 and Num2 using Address Operator (&). Required Output (assuming addresses starting from 0x10 for Num1 and so on) is shown below. (Note that addresses will be different on different machines) **Required Output:**

Num1 = 5

Num2 = 7

Address of Num1 = 0x10 //(This will be different on your machine) Address of Num 2 = 0x14//(This will be different on your machine)

1. Point xPtr to num1 and yPtr to num2
2. Print values of Num1 and Num2 by dereferencing xPtr and yPtr **Required Output:**

Num1 = 5

Num2 = 7

Address of Num1 = 0x10 //(This will be different on your machine)

Address of Num 2 = 0x14//(This will be different on your machine) \*xPtr = 5

\*yPtr = 7

1. Point sumPtr to sum and print sum by dereferencing sumPtr.

**Required Output:**

Num1 = 5

Num2 = 7

Address of Num1 = 0x10 //(This will be different on your machine)

Address of Num 2 = 0x14//(This will be different on your machine) \*xPtr = 5

\*yPtr = 7

\*sumPtr = 0

1. Add num1 and num2 using \*xPtr and \*yPtr and save the result in integer sum
2. Again Print sum using sumPtr **Required Output:**

Num1 = 5

Num2 = 7

Address of Num1 = 0x10 //(This will be different on your machine) Address of Num 2 = 0x14//(This will be different on your machine) \*xPtr = 5 \*yPtr = 7

\*sumPtr = 12

1. Print the values of xPtr and yPtr (cout<<”xPtr = ”<<xPtr<<endl) **Required Output:**

Num1 = 5

Num2 = 7

Address of Num1 = 0x10 //(This will be different on your machine) Address of Num 2 = 0x14//(This will be different on your machine) \*xPtr = 5 \*yPtr = 7 \*sumPtr = 12 xPtr = 0x10 //This output should be same as address of num1 i.e. &num1 yPtr = 0x14 //This output should be same as address of num2 i.e. &num2

**Help:**

cout<<”Num1 = ”<<num1<<endl; // Prints Num1 = 5

sum = \*xPtr + \*yPtr // Add num1 and num2 using \*xPtr and \*yPtr and save the result in integer sum