University of Central Punjab

Faculty of Information Technology

Data Structures and Algorithms
Spring 2021

Lab 01	
Topic	• Simple sorting algorithms — Selection sort • Simple searching algorithms — Linear search, binary search • Working with classes and multiple files.
Objective	The basic purpose of this lab is to revise some preliminary concepts of C++ that has been covered in the course of Introduction to Computing and Programming Fundamentals and Object Oriented Programming.

Instructions:

- Indent your code.
- Comment your code.
- Use meaningful variable names.
- Plan your code carefully on a piece of paper before you implement it.
- Name of the program should be same as the task name. i.e. the first program should be Task_1.cpp
- **void main() is not allowed. Use int main()**
- **You have to work in multiple files. i.e separate .h and .cpp files**
- **You are not allowed to use system("pause")**
- **You are not allowed to use any built-in functions**
- **You are required to follow the naming conventions as follow:**
    - **Variables:** firstName; (no underscores allowed)
    - **Function:** getName(); (no underscores allowed)
    - **ClassName:** BankAccount (no underscores allowed)

Students are required to complete the following tasks in lab timings.

Selection Sort

Selection sort is a sorting algorithm, in which we repeatedly find the next largest (or smallest) element in the array and move it to its final position in the sorted array. Assume that we wish to sort the array in increasing (ascending) order, i.e. the smallest element at the beginning of the array and the largest element at the end. We begin by selecting the smallest element and moving it to the lowest index position. We can do this by swapping the element at the lowest index and the smallest element. We then reduce the effective size of the array by one element and repeat the process on the greater (sub)array. The process stops when the effective size of the array becomes 1 (an array of 1 element is already sorted).

Pseudo code

```
Input: An unsorted array, A of N elements
Output: Sorted array, A

For I = 0:N-1
    SmallSub = I
    For J = I+1:N-1
       If A[J] < A[SmallSub]
          SmallSub = J
       End-If
    End-For
    Swap ( A[I], A[SmallSub] )
End-For
```

Linear Search

In computer science, linear search or sequential search is a method for finding a particular value in a list that consists of checking every one of its elements, one at a time and in sequence, until the desired one is found.

Pseudo code

```
Input: An unsorted array, A of N elements and value to be searched
Output: Index of searched element or -1 if not found

For I = 0:N-1
   If ( A[I] == Value )
     Return [I];
   End-if
End-For
return -1;
```

Binary Search

Binary search is another searching algorithm, used to search a specific value (or index of value) from the *sorted array*.

In binary search, we first compare the *value to be searched* with the item in the middle position of the array. If there's a match, we can return immediately. If the key is less than the middle key, then the item sought must lie in the lower half of the array; if it's greater than the item sought must lie in the upper half of the array. So we repeat the procedure on the lower (or upper) half of the array.

Pseudo Code

```
Input: An Sorted array, A of N elements and value to be searched
Output: Index of searched element or -1 if not found

low = 0, high = N-1;
while (low <= high)
   mid = (low + high)/2;
   If ( A[mid] == Value )
     Return mid;
   Else-if ( A[mid] < Value )
     low = mid + 1;
   Else
     high = mid - 1;
   End-if
End-For
return -1;
```

Task 1

Implement the Selection Sort Algorithm with the help of pseudo code given above.

Test the above program in main.

Task 2

Implement the Binary Sort Algorithm with the help of pseudo code given above.

Test the above program in main.

Task 3

Implement the Linear Sort Algorithm with the help of pseudo code given above.

Test the above program in main.

Task 4

A. Write a swap function and use it in main.

B. Now convert the above swap function using function templates and use it in main. Test the program on “int” and “float” datatypes.

Task 5

Create a C++ class named **Calculator**, with the following private attributes:

1. num2
2. num1

You have to design a simple calculator using class templates.

A **Calculator** must have following functions: add, subtract, multiply and divide two numbers using class template.

A **Calculator** must have display() function to display the data and other methods to perform above basic functions.

Your task is to instantiate the Calculator with different data types in main.

Task 6

Create a C++ class named **Student**, with the following private attributes:

3. regNo: char*
4. CGPA: double

Your task is to instantiate array of 4 Students. Initialize all objects of the array with different values.

The above task no. 6 must be done in separate header and .cpp files.