Assignment #8 (Marked Lab)

Problem Solving and Programming in C++

Department of Computer Science

Old Dominion University

Objectives: In this assignment students will practice manipulating lists of data and arranging items in an ascending/descending order. Students will also explore storing lists/arrays using different sorting algorithms including, the selection sort, bubble sort, and insertion sort algorithms. Comparison between the three algorithms are made based on the number of comparisons and item assignments (basic operations) each algorithm executes.

Important notes:

- This assignment should be implemented during the lab time of this week.
- Any submission after the lab time will not be accepted for marking.
- Please note that this assignment is only worth 30 marks.

Background: Ordering the elements of a list is a problem that occurs in many computer science contexts. For example, in a telephone directory it is necessary to alphabetize the names of subscribers. Similarly, creating a dictionary requires words be put in alphabetical order. While there are multiple sorting algorithms, some are better than others depending on the information being sorted. One thing in common which all sorting algorithms have is the sorting parameter. This "function" is what decides if two items in a list need to switch places. While this is easy for the standard data types (`int`, `char`, `double`, …) since the compiler already knows how to compare these, it might be tricky for data types created by you. Structs and classes you create will require implementing comparison functions if you are trying to sort them. For example in order to compare two dates you can write something like:

```
bool compDate(Date d1, Date d2){
      if (d1.year < d2.year
            return true;
      else if (d1.year == d2.year && d1.month < d2.month)
            return true;
      else if (d1.year == d2.year && d1.month == d2.month && d1.day < d2.day)
            return true;
      else
            return false;
}
```

The selection sort, bubble sort, and insertion sort algorithms are the most widely used algorithms. The selection sort repeatedly picks the smallest element to append to the result. The insertion sort algorithm splits the list into two portions (sorted and unsorted) and repeatedly adds new element to the sorted portion of the list. The bubble sort algorithm repeatedly compares neighbor pairs and swap if needed.

Problem Description: (30 marks)

Write a C++ program to create 2 identical arrays, `list1`, and `list2` – each list of 5000 elements. The program then sorts `list1` using bubble sort, `list2` using selection sort and outputs the number of comparisons and item assignments made by each sorting algorithm.

Hint: You need to create a random array and then copy it to end up with two identical lists (`List1`, `List2`). These two lists must be identical. You will need to sort `List1` using bubble sort, and sort the second list `List2` using the selection sort algorithm. Finally, you will need to output the number of comparisons and item assignments made by each sorting algorithm. Plea se note that the array should be randomly generated and you can do that by using the random function: `rand()` from the `<cstdlib>` library.

Submission notes:

- Submit all files from your project including the .cpp file(s).
- Zip all files and name it as "Assg8_cslogin", where the cslogin is your login ID for the computers at the Department of Computer Science at ODU.
- Submit the .zip file in the respective Blackboard link.

Extra Task (Not Graded):

Note: *Students have the option to complete this task during or after the lab time.*

You need to create one more list, `List3` – of 5000 elements. The three lists (`List1`, `List2`, and `List3`) must be identical. Modify your program to sort `list3` using the insertion sort algorithm. Next, compare the performance of this algorithm with the other sorting algorithms which you implemented during the lab time.