CSCI 24000 – Fall 2017
Assignment #6 – Polymorphism, Sorting, Big-O My!

Due: 12/11/2017

This sixth and final assignment will allow you to put together all of your knowledge you have obtained this semester into one project. For this project we will use all four pillars of the Object-Oriented paradigm – with concentrated focus on Polymorphism. You will also explore how to evaluate, in terms of runtime performance, your program by providing Big-O analysis. If this hasn't overwhelmed you yet – read on…

Your job for this assignment is to produce a program that sorts a list of numbers in descending order. Your program will need to read-in, from a file, a list of integers – at which point you should allow the user an option to choose to sort the numbers in descending order via one of the two Sorting algorithms that we have explored (Insertion/Bubble). Your program should use the concept of Polymorphism to provide this sorting feature. As output, you will display the sorted list back to the user.

A secondary task of this assignment is to provide runtime analysis of your program. As discussed in lecture, Big-O notation provides us a way to evaluate the performance of our code. You will provide in a separate text file the Big-O of the two sorting algorithms that we have implemented in this assignment. This program will be written in **C++** and must compile and run on **Tesla** (tesla.cs.iupui.edu).

Your program will be menu driven in which you will provide the following prompts to the user:

```
1. Load Integers (From File)
2. Exit Program
```

Once the user has loaded the list of integers the following options should be provided:

```
1. Insertion Sort
2. Bubble Sort
3. Exit Program
```

The text file containing integer data will be provided to you. The filename will be **data.txt** – this file can be found (and downloaded) on Canvas. The file will contain **fifty integers** (50) and they will be comma separated on a single line.

Development Process:

All development must take place on the **master branch** in a **private** GitHub repository. You must add Me (rrybarcz) and all four (4) TA's as collaborators. It is strongly recommended that you commit and push often! We will be checking to make sure that you are actively pushing changes to your repository – failure to do so will result in a loss of points. You are also required to include the Honor Pledge and Digital Signature in each one of your source files (any file that you have written "code" in – you need to include the Honor Pledge and Digital Signature).

This is a challenging assignment – it is essential you start early on it in order to not only complete it in time but also to earn a good score on it.

You are required to implement the following items as part of this assignment:

- Your program must contain at-least **four** (4) classes.
    - Driver
    - Sort
    - Insertion Sort
    - Bubble Sort
- You must use the concepts of **Inheritance** and **Polymorphism** in order to complete this assignment.
    - Your `Sort` class **<u>must</u>** be a pure virtual class.
    - You should use proper OO approaches to implementing Inheritance.
- It is recommended you use `stringstream` to read-in and parse the data file.
- Your sample output **<u>must</u>** match that shown below – note that in order to "re-sort" the list you must also "re-load" the data file.
- You are **required** to use an `Array` to store the data read-in from the file.
- You are **required** to use the Heap to store your Objects in memory.
- You program should have **<u>no memory leaks</u>** – a memory leak will result in a deduction of points.
    - Make sure to run Valgrind against your submission to check for memory leaks and include this report with your submission.
- You are **required** to provide a brief (about one paragraph per algorithm) overview/analysis of the Big-O runtime of your program with respect to the two sorting algorithms we are implementing as part of this assignment – we will cover this in brief overview in lecture, you will need to provide some additional external research on your own to complete this assignment.

Below is the output of what your program should display when executed:

```
1. Load Data (From File)
2. Exit Program
Please enter your selection: 1

Unsorted Array: 39, 14, 100, 16, 93, 24, 62, 68, 52, 76, 86, 48, 15, 41, 83, 55, 18, 30, 74, 7, 31, 44, 67, 81,
70, 27, 53, 59, 61, 19, 56, 35, 88, 58, 72, 98, 38, 64, 94, 69, 50, 46, 78, 6, 57, 89, 26, 20, 79, 49

1. Insertion Sort
2. Bubble Sort
3. Exit Program
Please enter your selection: 1

Insertion Sort: 100, 98, 94, 93, 89, 88, 86, 83, 81, 79, 78, 76, 74, 72, 70, 69, 68, 67, 64, 62, 61, 59, 58,
57, 56, 55, 53, 52, 50, 49, 48, 46, 44, 41, 39, 38, 35, 31, 30, 27, 26, 24, 20, 19, 18, 16, 15, 14, 7, 6

1. Load Data (From File)
2. Exit Program
Please enter your selection: 1

Unsorted Array: 39, 14, 100, 16, 93, 24, 62, 68, 52, 76, 86, 48, 15, 41, 83, 55, 18, 30, 74, 7, 31, 44, 67, 81,
70, 27, 53, 59, 61, 19, 56, 35, 88, 58, 72, 98, 38, 64, 94, 69, 50, 46, 78, 6, 57, 89, 26, 20, 79, 49

1. Insertion Sort
2. Bubble Sort
3. Exit Program
Please enter your selection: 2

Bubble Sort: 100, 98, 94, 93, 89, 88, 86, 83, 81, 79, 78, 76, 74, 72, 70, 69, 68, 67, 64, 62, 61, 59, 58, 57,
56, 55, 53, 52, 50, 49, 48, 46, 44, 41, 39, 38, 35, 31, 30, 27, 26, 24, 20, 19, 18, 16, 15, 14, 7, 6

1. Load Data (From File)
2. Exit Program
Please enter your selection: 2

Goodbye!
```

Submission:

All assignments must be submitted on IU GitHub (github.iu.edu) in your master branch by the due date. The name of your IU GitHub repository must be as follows: **csci24000_fall2017_A6**. You should be submitting the source files (e.g., ***.cpp**), header files (e.g., ***.h**), a **Makefile**, a Valgrind file (e.g., **valgrind.txt**), and a file (**analysis.txt**) that contains your discussion of the Big-O analysis of your program. Please be sure to have us added as collaborators to your assignment repository prior to your submission. If you submit late, it is your duty to notify me (Dr. Rybarczyk) of the late submission – we will not be checking for late submissions without proper notification. Failure to follow any of these rules will result in a **zero** (0) on your assignment submission.

Blackbelt:

For this project the Blackbelt extension is as follows:

- Write this program in Java and compare the C++ version with the Java version as part of your algorithm analysis discussion. (+10)

OR

- Implement the Quick Sort algorithm in addition to the two sorting algorithms that are part of the regular standard submission and randomly select which sort is performed. (+10)

Important Note:

I fully realize that these are standard algorithms and well published and used. It is very easy to Google solutions to this problem…I know that. ☺ Remember the Honor Pledge & Digital Signature! If you need assistance please contact myself or one of the TA's for help, **do not** use Stack Overflow.