CSCE 221 Cover Page

Programming Assignment #3

First Name: Seth Last Name: Barberee UIN: 324009275

Any assignment turned in without a fully completed coverpage will receive ZERO POINTS.

Please list all below all sources (people, books, webpages, etc) consulted regarding this assignment:

CSCE 221 Students	Other People	Printed Material	Web Material (URL)	Other
1.	1. Timothy Ebringer	1.	1.	1.
2.	2. Nathan Leake	2.	2.	2.
3.	3.	3.	3.	3.
4.	4.	4.	4.	4.
5.	5.	5.	5.	5.

Recall that University Regulations, Section 42, define scholastic dishonesty to include acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion. Please consult the Aggie Honor System Office for additional information regarding academic misconduct – it is your responsibility to understand what constitutes academic misconduct and to ensure that you do not commit it.

I certify that I have listed above all the sources that I consulted regarding this assignment, and that I have not received nor given any assistance that is contrary to the letter or the spirit of the collaboration guidelines for this assignment.

Today's Date: 3/21/18

Printed Name (in lieu of a signature): Seth Barberee

1. Introduction - In this section, you should describe the objective of this assignment. **The purpose of this assignment was to compare and contrast the different implementations of sorting algorithms using Priority Queues and Heaps. Two of them use different forms of a Priority Queue while the other uses a Heap to sort its contents.**
2. Theoretical Analysis -In this section, you should provide an analysis of the complexity of an insert operation and the complexity of the sort (inserting all of the items and then removing all of the items). Describe the advantages and disadvantages of the three strategies. What is the complexity of an insert(on average) for the different implementations? What is the complexity of the sort (on average) for the different implementations?

Insert:

- **Unsorted:** $O(1)$
- **Sorted:** $O(n^2)$
- **Heap:** $O(nlog(n))$

Insert/Remove:

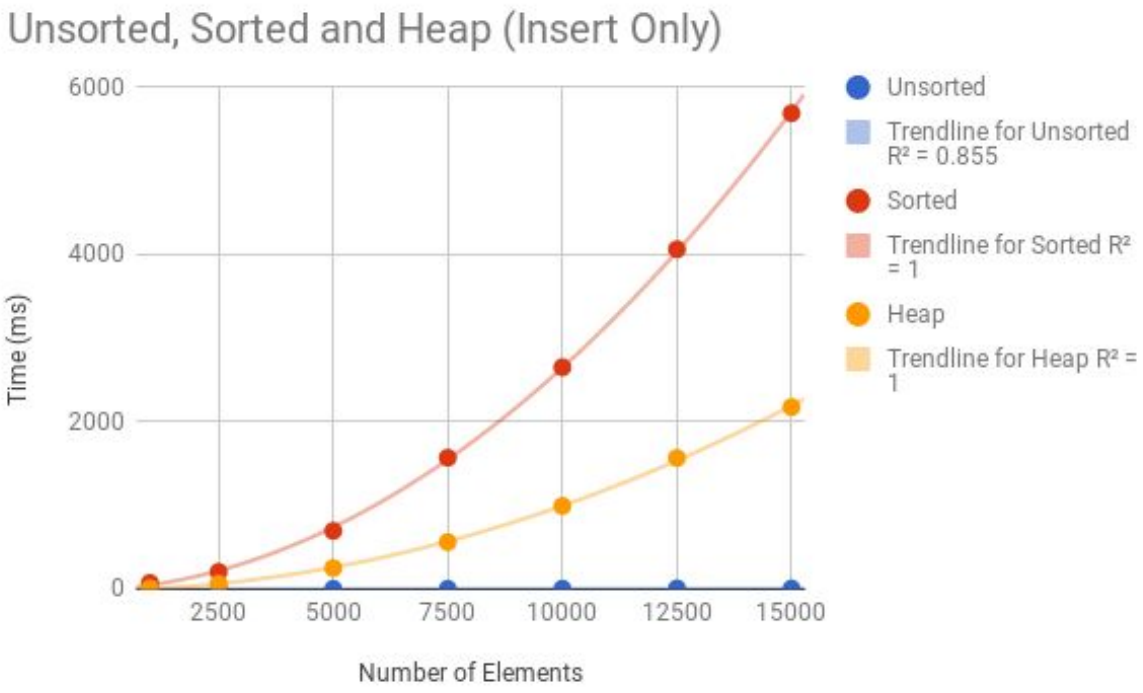
- **Unsorted:** $O(n^2)$
- **Sorted:** $O(n^2)$ **, Removal:** $O(1)$
- **Heap:** $O(nlog(n))$ **, Removal:** $O(1)$

Advantages/Disadvantages:

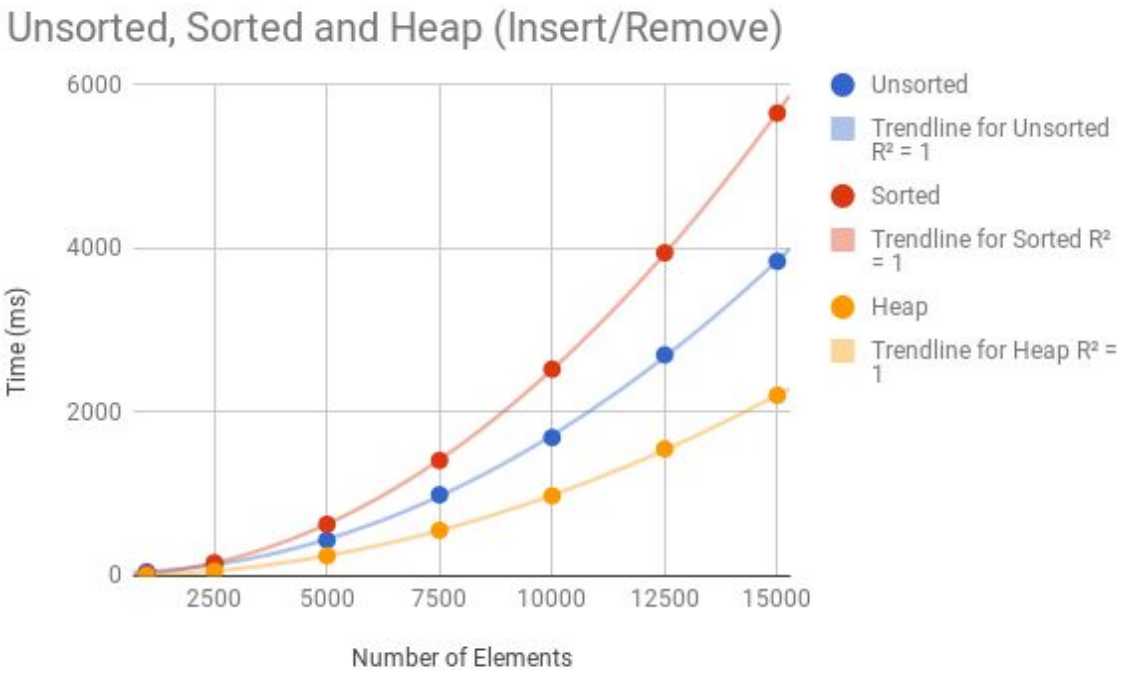
- **Unsorted: Quick insert but slowest removal**
- **Sorted: Slowest insert but quick removal**
- **Heap: Quick removal and moderate insertion**

3. Experimental Setup - In this section, you should provide a description of your experimental setup, which includes but is not limited to
    a. Machine specification
    - **Intel Core i5 6600K at 4.2 GHz**
    - **16 GB Memory**
    - **Arch Linux x86_64 (Kernel Version here)**
    b. How did you generate the test inputs? What input sizes did you test? Why? **Test inputs were generated with the random math function in C++. The code for this is in TimeTest (commented in the code where it is)**
    c. What data structures did you use for your List implementation of the Priority Queue? **STL List**
    d. How many times did you repeat each experiment? **2 times for each implementation for which an average was taken for the data point for the graph below.**

4. Experimental Results - In this section, you should compare the performance (running time) of the insert() operation and the sort in the three different implementations to one another and to their theoretical complexity.
    a. Make a plot showing the running time (y - axis) vs. the number of insert operations (x - axis).



b. Make a plot showing the running time (y-axis) vs. the number of items inserted and removed (x-axis). You must use some electronic tool (matlab, gnuplot, excel, ...) to create the plot



c. Provide a discussion of your results, which includes but is not limited to:
   i. Which of the three Priority Queue implementations performs the best? Does it depend on the input?
   Heap Sort. It did not depend on the input size.
   ii. To what extent does the theoretical analysis agree with the experimental results? Attempt to understand and explain any discrepancies you note
   All implementations matched with the theoretical results by the data points shown above.