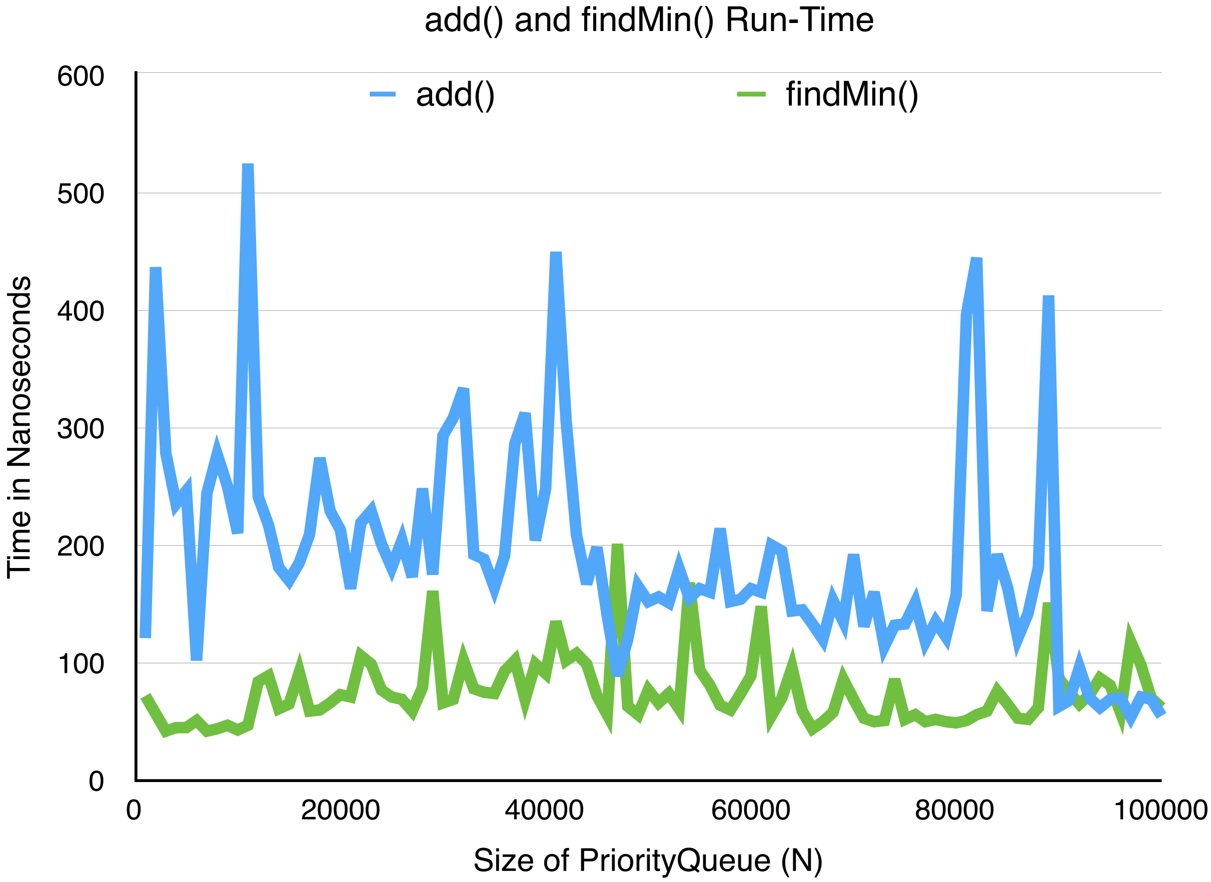
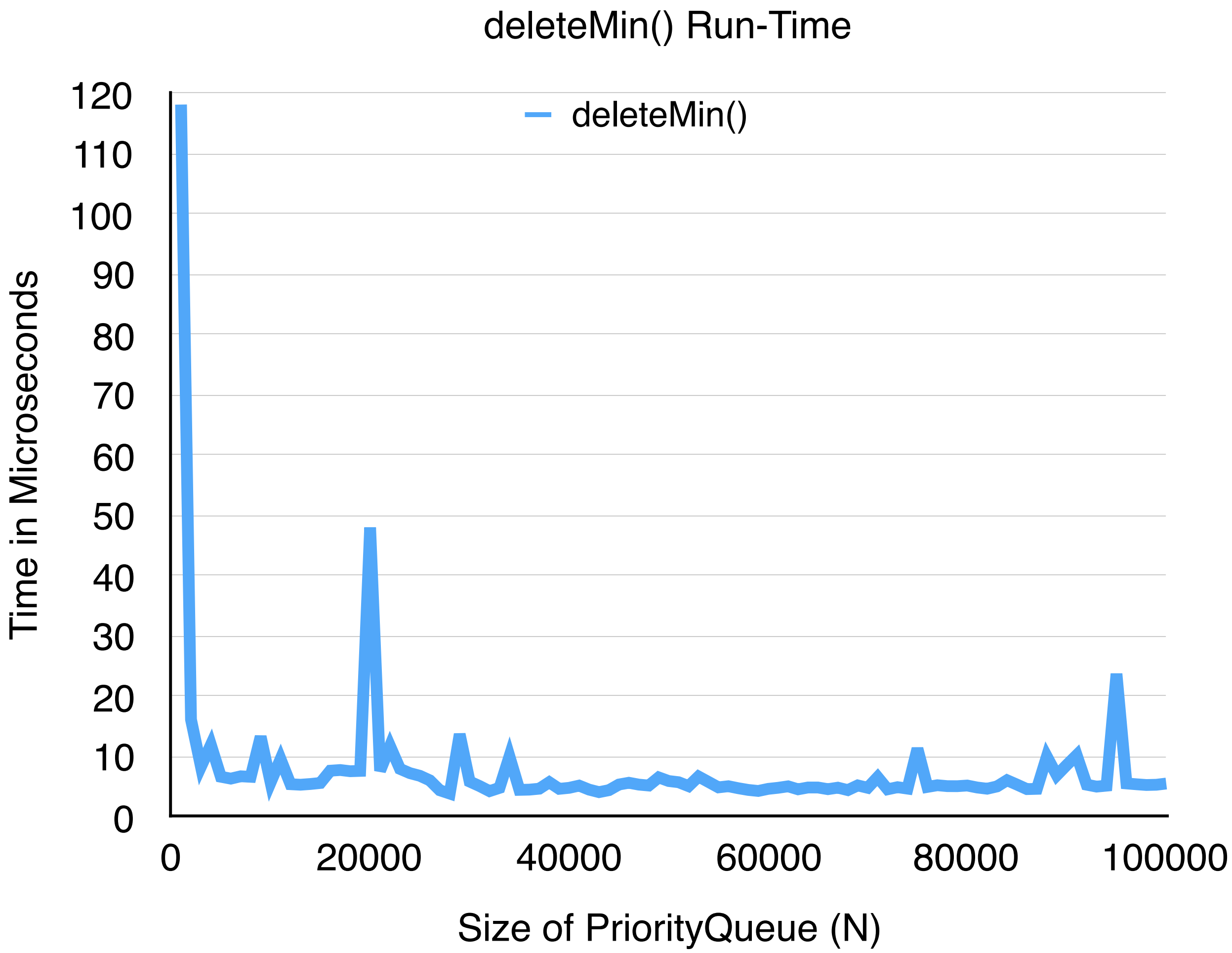
Samuel Teare
UID: u0663592
Assignment 11 Analysis

1. In order to test the run-time efficiency of my Priority Queue, I created an experiment that would test the run-time of the add, findMin, and deleteMin methods. I first created an empty PriorityQueue. I added 100,000 objects to the queue and recorded the time it took to add every 1000th item, findMin at every 1000th size, and deleteMin at every 1000th size. I took these run times and plotted the two graphs below. The add and findMin methods kept a fairly constant run-time. This is apparent in the first graph below. The run-time in the first graph is shown in nanoseconds. The run-time of the add method will be affected by increasing the size of the array that backs the PriorityQueue, but the average run-time for adding an item will be O(c).



The second graph shows the run-time of the deleteMin method. This had a longer run-time but still seemed very constant. Its run-time is shown in microseconds. I attempted to test this method with larger PriorityQueue sizes, but it was becoming too taxing on my computer.



2. The cost of the add method for a priority queue is O(c) in best and average cases and O(logN) in the worst cases. The worst case for the add method is when an item is added that is smaller/lesser than the current min item. The findMin() method has a O(c) cost for all cases. This is because it only has to look at the first item in the array that backs the PriorityQueue. The deleteMin method has a cost of O(logN) in almost all cases. This is because it rarely stops before getting close to the last level or two of the queue.

My implementation appeared very consistent with the costs listed above. I tested my add method for both best and worst cases and it had very similar run-time. I did not see O(logN) with my add method. My findMin method was O(c) as was expected. My deleteMin method appeared to have a O(c) runtime although it took 10-100 times longer than the add or findMin method.

3. A priority queue can be applied to manage matters with a time oriented natures. An example of this would be a ticketing system that manages issues for an IT department. A priority queue would allow the system to return the most important issue to be resolved. It could also be used to return the issues that would require the most or the least number of man hours to accomplish. This would be helpful to optimize the efficiency of a department in improving systems or resolving issues.

4. I spend around 8 hours on this assignment.