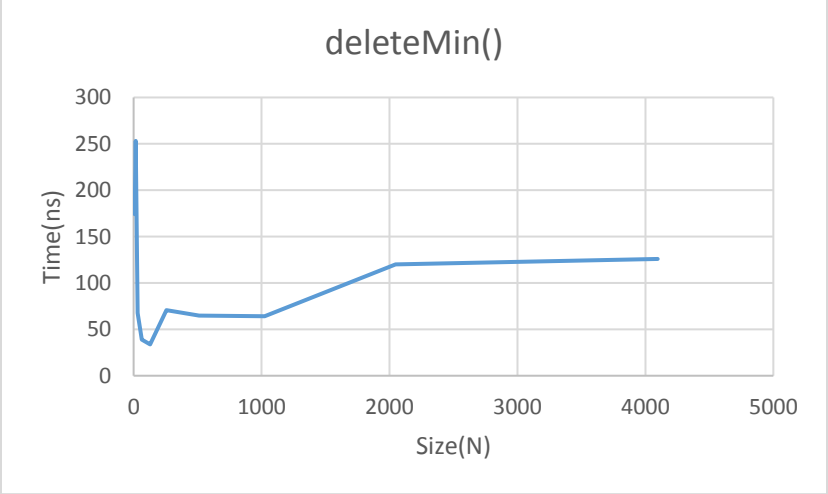
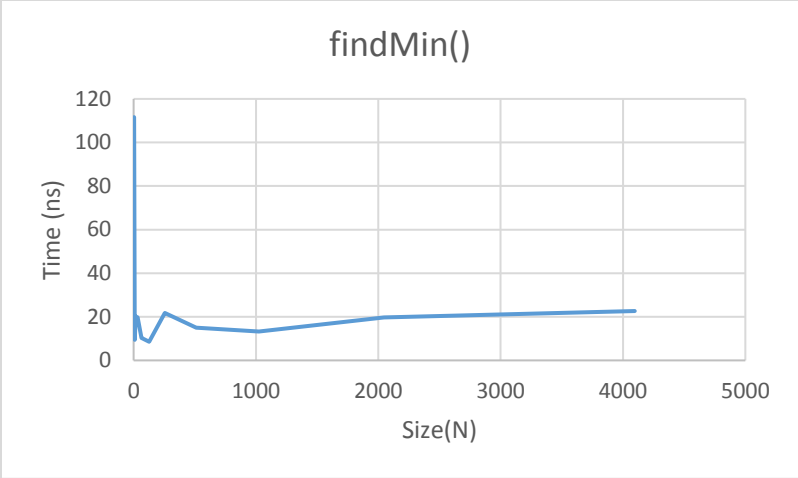
Assignment 10

1. The experiment was designed to test the priority queue methods, add(), deleteMin() and findMin(). Each experiment iterates over an exponential size. This generated different sizes of the priority queue. Multiple iterations allowed a better average time to be established. An array list is generated and filled. The array list is shuffled using the Java's Collection. Then a timer method is called. The timer method spins up the timer. For each iteration the method adds the array list to the priority queue. Then the time is taken and the method is called. The time is taken again and the difference in time is added to a total. When the timer has completed all iterations the average time is calculated.
   add() worst adds the Integer '0' which is not in the priority queue and would be the minimum.
   add() average adds a random Integer that is in the priority queue. Because priority queues can hold duplicates.
   add() best adds the Integer of the size which is not in the priority queue. It's also guaranteed to be the largest item in the priority queue.

   deleteMin() appears O(logN). There is a noticeable spin up cost in the beginning. Then the method appears to have inconsistent growth. The trend of the method is not linear but appears O(logN).

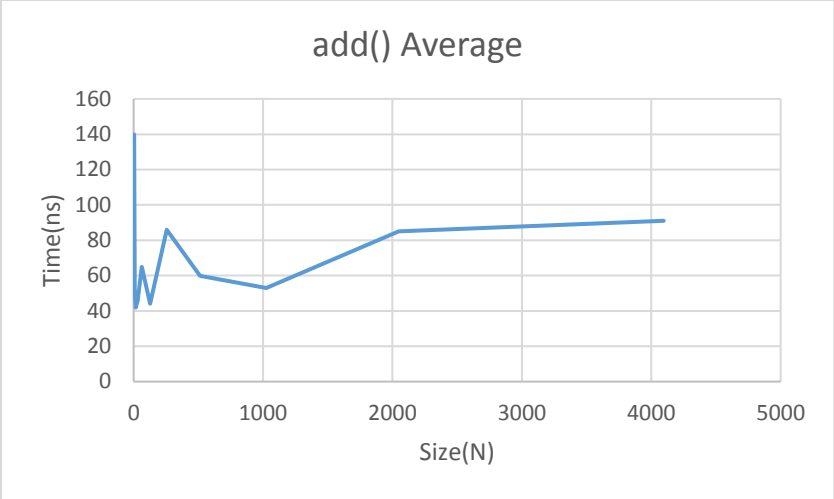


   findMin() appears O(c). There is a noticeable spin up cost in the beginning. After spinup the method appears to flatten out to a constant value.

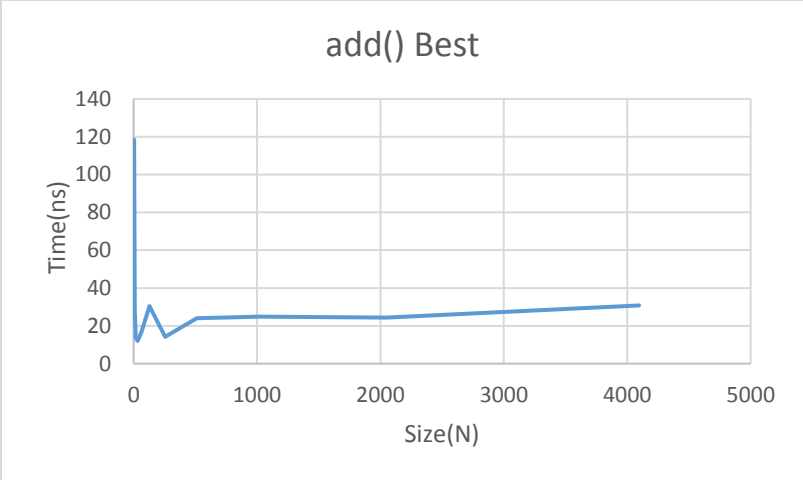


add() Worst appears O(logN). There is some spin up cost in the beginning. Then the method appears to have inconsistent growth periods. This can be due to resizing the array. The trend of the method is not linear but appears O(logN).

add() Average appears O(logN). There is more spin up cost in the beginning. Then the method appears to have inconsistent growth periods. This can be due to resizing the array. The trend of the method is not linear but appears O(logN). Though there is a noticeable difference between the Worst case and the Average case.



add() Best looks to be O(c). There is a considerable spin up cost. Then the method is constant time for a period before increasing. This increase is due to a single point of data. There could be a number of reasons for this increase, the most likely of which is resizing the array.



2. deleteMin() removes the min value and percolates down through the array. This method looks at every values child. Thus the expected big O complexity is O(log N).
   findMin() returns the first value in the underlying array structure. Access into the array structure is O(c). Thus the expected big O complexity is O(c).
   add() Worst case adds the minimum value in the array. Then it percolates all the way up the array to replace the current min value. Percolating complexity is O(log N) and swapping values is O(c). The expected complexity is O(long N).
   add() Average case adds a random value to the array. Then it percolates at least partially up the array and swaps with some values. Percolating complexity is O(log N) and swapping values is O(c). The expected complexity is O(long N).
   add() Best case adds the largest value to the data structure. It percolates up one value. Percolating complexity is O(c) and no swapping occurs. The expected complexity is O(c).
   Each method performed as expected.
3. In terms of data structures priority queues are important for use in path optimization. These data structures can be used conjointly with graphs to implement methods like Dijkstra's algorithm which finds the best route on a graph. This can be implemented in a min or max heap depending on which optimization is being used.

Priority queues are also used in calendars, scheduling and process sequencing. Any time one event precedes another it may be put into a priority queue. Then items are returned in the order that they need to occur. A subgroup of this is discrete event simulation. Thus runs simulation processes in order. However this application of priority queues can also be extended to things like Google Calendar. Google Calendar may use a priority queue to notify users of upcoming events. Each even is an object containing an alarm time, an event time and an event. A notification is received as a result of popping an alarm time priority queue.

4. I spent about 6 hours on this assignment.