**CSE 373 Homework 3 Write up**

**<Your name, UW Net ID and student number:>**

1. **What is the worst case asymptotic running time of isEmpty, size, insert, findMin, and deleteMin operations on all three of your heap implementations? *For this analysis you should ignore the cost of growing the array. That is, assume that you have enough space when you are inserting a value.***

<your answer here>

1. **Timing your code: Perform several timing experiments (similar to what you did in Homework 2, where you timed pieces of code), to examine the running time of all three of your heap implementations. An experiment will include running the same client code (that uses a Priority Queue) for your three different heap implementations *for at least four different values of N* and timing this. It is up to you to write and to determine what this client code should be. Just be sure that it exercises your insert and deleteMin operations in a reasonable manner, including eventually deleting everything that has been inserted into the heap. You are not required to explicitly measure calls to findMin, size, and isEmpty but feel free to do so if interested. Similar to Homework 2, graphing your results is recommended, but a table of results will work also. Please note that similar to Homework 2, you are required to turn in the code you used to do your timing experiments.**

<your answer here>

1. **Compare what you see in your experiments, to what you expected to see based on a big-Oh analysis. (This is also similar to what you did in Homework 2.) In your discussion, answer these questions:**
	1. **How useful was the asymptotic analysis for predicting the measured run time of insert and deleteMin for your three implementations?**

<your answer here>

* 1. **If your predictions differed substantially from your measured times, gives reasons why this might have occurred.**

<your answer here>

* 1. **Which of your three implementations would you recommend to someone who needs to use a heap? Why is that one preferred? Are there any conditions under which you might suggest using your other implementations?**

<your answer here>

1. **Briefly discuss how you went about testing your three heap implementations. Feel free to refer to your testing files, which you should submit.**

<your answer here>

1. **You implemented a binary-heap and a three-heap. We could have also asked you to implement a four-heap, a five-heap, etc.**
	1. **In a short table, indicate for a binary heap, a three-heap, a four-heap and a five-heap, where the children for the node at array index i are. For example, the first row of the table would indicate that for a binary heap, the two children would be at i\*2 and i\*2+1.**

<your answer here>

* 1. **For a *d*-heap where *d* is a variable representing the number of children (like two, three, four, five, ...), give an arithmetic formula for calculating where the left-most child for the node at array index i are. For example, a *wrong* answer in the right format would be *i\*d+14*. Naturally, your formula should produce the right answer for all the rows in your table from part (a).**

<your answer here>

**Appendix**

<place anything that you want to add here.>