CMSC 341 Data Structures
Priority Queue Review

1. Define the following terms
   (a) priority
   (b) priority queue
   (c) min binary heap
   (d) partial ordering
   (e) null path length in a binary tree
   (f) leftist binary tree
   (g) leftist heap
2. Insertion and deletion (of the minimu element) in a (min) binary heap are $O(lg\, n)$ on average. Explain why this is so.
3. Finding the minimum element in a (min) binary heap is $O(1)$ in the worst case. Explain why this is so.
4. Although a binary heap is conceptually a binary tree, it can be implemented as an array. Explain why this is so.
5. In a min binary heap with N elements, what is the range of indicies in which the largest element will be found?
6. Describe, in English, an algorithm to find the largest element in a min binary heap. Wht is the asymptotic worst-case performance of your algorithm?
7. Assume that the array representing a min binary heap contains the values 2, 8, 3, 10, 16, 7, 18, 13, 15. Show the contents of the array after inserting the value 4.
8. Assume that the array representing a min binary heap contains the values 2, 8, 3, 10, 16, 7, 18, 13, 15. Show the contents of the array after deleting the minumum element.
9. Show the array representing the min binary heap constructed using the initial values 18, 2, 13, 10, 15, 3, 7, 16, 8.

10. Prove that the largest element in a min binary heap is a leaf.

11. Prove that a complete binary tree is a leftist tree.

12. Prove for any leftist tree with N nodes, the number of nodes, R, on the rightmost path to a non-full node is given by

$$R \leq lg(N+1)$$

13. Given the drawing of a binary tree, determine if the tree is a leftist tree and if it is a leftist heap. Give reasons why or why not.

14. Given the drawings of two leftist heaps, draw the leftist heap that results from merging them.

15. Describe how to perform the **findMin, insert,** and **deleteMin** operations on a leftist heap.

16. Describe a method for constructing a leftist heap from an intial set of N values. Your algorithm must run in $O(N)$ time.