Quiz 2 Practice Questions

1. Given a heap and a number k, design an efficient algorithm that outputs the top-k largest element from the heap. What is the running time of the algorithm? Can you design an algorithm that runs faster than O(k*log n)?
2. Insert the following numbers into an initially empty red-black tree: 8 2 4 7 5 3 1 6. (To solve this problem correctly, you will need to refer to the text-book for details about red-black trees in addition to the main ideas discussed in class).
3. Design a "worst-case" red-black tree with 10 nodes, i.e., a red-black tree with the longest possible path from the root to a leaf.
4. Instead of binary heaps, suppose you had to implement ternary heaps (i.e., where each node has up to three children). Explain, how you would implement such heaps using arrays, and how you can determine child and parent pointers. What are the advantages/disadvantages of ternary heaps over binary heaps?
5. Can you use a binary search tree to simulate heap operations? What are the advantages/disadvantages of doing so?
6. Design an algorithm to convert a given binary search tree into a heap efficiently. What is the running time of your algorithm?
7. Assume a heap is arranged such that the largest element is on the top. Suppose you are given two heaps S and T, each with m and n elements respectively. What is the running time of an efficient algorithm to find the 10th largest element of S U T.
8. What is minimum possible number of nodes in a red-black tree which contains two black nodes from every root-to-leaf path?
9. Suppose we define a red-black tree where along each path the number of black nodes is the same, and there cannot be more than three consecutive red nodes.
    i. What is the ratio of the longest possible path length to the shortest possible path length in this tree?
    ii. For a tree that has b black nodes along each path, what ratio of the maximum possible number of nodes to the minimum possible number of nodes in the tree?

10. Suppose you want to implement a heap (with the largest element on top) in an array. Consider the different arrays below, determine which of these arrays cannot possibly be a heap:
    a. 7 6 5 4 3 2 1
    b. 7 3 6 2 1 4 5
    c. 7 6 4 3 5 2 1
    d. 7 3 6 4 2 5 1
    e. 7 4 6 3 2 1 5

11. Can you use any of the previously studied data structures (e.g. heaps, red-black trees) for the Union-Find problem? Explain your answer.
12. Suppose we are working with a Union-Find data structure as described in class (this time path compression is allowed for Find operations). Node z is at a distance of 4 from the root r of its tree (i.e.4 edges away). We now perform a Find on z.
    a. By how much will the degree of r change? Explain your answer.
    b. By how much will the degree of z change? Explain your answer.

13. What is the worst-case performance of the Union-Find data structure we discussed (with union by rank and path-compression)?
14. Suppose you are given a set S = {1, 2, 3……9}. Show the steps of Union-Finding by union by size and union by rank (no path compression) for the sequence of following operations: Union (1,2), Union (3,4), Union (2, 4), Union (5,6), Union (6, 7), Union (7, 8), Union (8, 9), union (4, 9).