This is an open text and open notes exam.

Write your answers on your own paper. The instructor will have blank paper in case you have none.

Each question is worth 10 points.

1. Suppose we are given an unsorted array of salaries - $A[i]$ contains the salary of person $i$, and $T = \sum_{i=1}^{n} A[i]$ is the total amount of money earned by the population. We want to be able to answer simple economics questions, such as “What is the percentage of the income earned by the top quintile (20%) of all earners?”.

To deal with these questions, you should provide an $O(n)$ procedure `percTot(A, k, j)` which will add up all salaries made by those people whose salaries have rank between $k$ and $j$ (inclusive), and divide it by the total of all salaries.

Thus, the answer to the “top quintile” question above would be `percTot(A, .8n, n)`.

2. Provide solutions to the following recurrence relations

   (a) $T(n) = 8 \cdot T(n/2) + 10 \cdot n^{10}$
   (b) $T(n) = 8 \cdot T(n/2) + 10^{10} \cdot n$
   (c) $T(n) = 16 \cdot T(n/4) + 313 \cdot n^{2}$

3. Into an initially empty red-black tree

   (a) insert the values 11, 15, 17, 5, 9, 14, 12, 13, 3, 2, 1
   (b) then delete 11

4. Consider an array containing the values 6, 4, 11, 2, 9, 1, 8, 3, 10, 5, 7 in locations 1-11. Illustrate the build-heap method converting this array into a max-heap.

5. Into an initially empty binomial heap (which, as in the text, is a MIN-heap):

   (a) insert the values 11, 15, 17, 5, 9, 14, 12, 13, 8, 7, 18
   (b) then remove the min.

6. What are the worst-case and average-case run-times of the following pieces of code?

   (a) ```
   for i = 1 to n*n
       for j = 1 to i*i
           sum++
   ```
   (b) Given unsorted array A
7. Some lower bound questions:

(a) We want to take a list of $n$ integers and create a BST containing these values. Show that this cannot be done in $O(n)$ time. (In fact, $\Omega(n \log n)$ time is required. As a hint, recall that an inorder traversal of a BST takes $O(n)$ time.)

(b) Consider the problem of looking in a sorted array $A$ for an element $x$. We would use binary search, which uses $O(\log n)$ time. Show that any comparison based method must make $\Omega(\log n)$ comparisons.

8. Let $T$ be a RB tree and $p$ a pointer to a node on the left branch of $T$ (that is, it can be reached by following some number of left-child pointers starting at the root - it is not necessarily a leaf). Create two new RB trees $U$ and $V$ (you may destroy $T$) such that $U$ contains values $\leq p.key$ and $V$ contains values $\geq p.key$. This should be accomplished in $O(\log n)$ time.

Total: 80 points