Problem 1. At time $T = 0$ someone told you a joke. You told it to two of your friends at time $T = 1$, subsequently at every minute each person told the joke to two of their friends, and so on. At time $T = 5$, how many people, including yourself, (but not the one who told this to you), have heard the joke? If you were to construct a binary tree with each node being a person what would be its height? (by the way this is how rumors spread). Show your work without actually constructing a binary tree.

Problem 2. You are given an array of $n$ distinct unsorted numbers. Write a combination of pseudo-code and concise English description to find the second smallest number in this array using at most $n + lg\, n - 2$ comparisons. Note: You may assume $n$ is a power of 2.

Problem 3. Consider the following recurrence equation for the runtime of some algorithm:

$$T(n) = 3T(n/5) + 2n + 1, \qquad T(1) = 4$$

(a) Calculate $T(25)$ by hand. Show your work.

(b) Use the tree method to solve the recurrence exactly, assuming $n$ is a power of 5. For each subpart briefly justify and/ or show your work when appropriate.

1. Draw the tree. You should show at least three levels at the top and at least two levels at the bottom (as done in class).
2. What is the height of the tree? (Note that a tree with one node has height 0, a tree with a root and some children are at level 1, etc.)
3. How many leaves are there?
4. What is the total work done by the leaves?
5. What is the size of each subproblem at level $i$?
6. How much work does each subproblem at level $i$ do?
7. What is the total work for level $i$?
8. Write a summation for the total work not including the leaves?
9. Simplify the summation.
10. What is the total work for the entire algorithm?

Problem 4. For this question, we will have to modify Merge sort algorithm. In the divide and conquer step, you will divide the array into thirds, recursively sort each third, and finally combine the results using a three-way merge routine. Answer the following question:

1. Write pseudo code for the modified Merge sort algorithm
2. What is the recursion equation?
3. What is the base case?
4. Show the recursion tree with at least 3 top levels and 2 bottom levels.
5. Solve the recursion equation using the recursion tree approach. Exactly how many comparisons does this modified merge sort algorithm use in the worst-case? Show your work.

6. What is the value of T(3)?
7. Verify the base case
8. In terms of runtime, would this modified Merge sort be faster or slower than the regular algorithm?