1. Consider an array of size nine with the numbers in the following order $80, 60, 40, 20, 90, 50, 10, 70, 30$.
   (a) Form the heap using the algorithm described in class. Show the heap as a tree. Show the heap as an array. Exactly how many comparisons did heap creation use?
   (b) Finish heapsort: Start with the heap created in Part (a). Show the *array* after each element sifts down *after heap creation*. How many comparisons does each sift use? What is the total number of comparisons *after heap creation*?

2. Assume that there are n points in the plane with integer indices. We would like to find the k points that are closest to the origin, where k is a small integer, maybe around $\lg n$ or even less.
   (a) Describe an efficient algorithm very briefly in English.
   (b) Give the pseudo code.
   (c) Analyze its time complexity. Just give the answer in order notation.

3. Assume that we visit the nodes in a binary tree in the order that they are numbered in heapsort, but you may only move directly to nodes connected by tree edges. You want to start at the root and end at the root, and visit all nodes up to and including node n. You will have to visit intermediate nodes multiple times.

   For example, assume that you want to visit up to node 3 then you will visit the nodes in order 1, 2, 1, 3, 1. If you want to end at node 7 you will visit the nodes in order 1, 2, 1, 3, 1, 2, 4, 2, 5, 2, 1, 3, 6, 3, 7, 3, 1. In the first example you will cross edges 4 times, in the second example you will cross edges 16 times.

   We want to calculate how many edge crossings there are. To keep things simple we will only consider visiting up to the last node of a level. Then we can calculate by levels rather than node numbers. There are a number of ways of solving this problem but we want to do it with recurrences. Levels are normally numbered 0 for just the root (node 1), 1 for nodes 2 and 3, 2 for nodes 4, 5, 6, and 7, etc.
   (a) Write a recurrence for the number edges crossed when completing k levels (levels 0 to $k-1$). (Note that $n = 2^k - 1$.)
   (b) Solve the recurrence using the tree method. Show your work.
   (c) Solve the recurrence using the "master theorem" (posted). Show your work.
   (d) Rewrite your answer in terms of n.

4. CHALLENGE PROBLEM (not part of your grade). Not all of these problems have been solved by the instructor so you are on your own.
   (a) Calculate the number of edge crossings for ending at a general node n (not just completing a level).
   (b) Assume that every time you visit a node you add its value to a sum that starts at 0. For example, ending at node 1 will give sum 1, and ending at node 3 will give sum 8 (1+2+1+3+1). Calculate this sum when completing level k.
   (c) Calculate the above sum when completing (a general) node n.