Problem 1. Consider the following recursive procedure for bubble Sort (which might be similar to your solution to Problem 3 on Homework 1).

```
procedure BubbleR(A,n)
    if n>1 then
        for j = 1 to n-1 do
            if A[j] > A[j+1] then A[j] ↔ A[j+1]
        end for
        BubbleR(A,n-1)
    end if
end procedure
```

(a) Write a recurrence for the exact number of comparisons.

(b) Solve the recurrence for $n = 3$. Show your work.

Problem 2. Assume that you are merging two sorted lists, each of size m. Note that this will use exactly $2m - 1$ comparisons in the worst case (since at some point one list will become empty and the other list will not be), and exactly m comparisons in the best case.

Assume the lists are random in the following sense: You are executing Merge Sort on a random array (or more precisely a random permutation), and you are about to do a merge.

(a) What is the probability that the algorithm does exactly $2m - 1$ comparisons. Justify. Simplify.

(b) What is the probability that the algorithm does exactly $2m - 2$ comparisons. Justify. Simplify.

(c) What is the probability that the algorithm does exactly m comparisons. Justify. Simplify.

(d) **Challenge Problem. Will not be graded.** What is the average case number of comparisons?

Problem 3. When running Merge Sort, it is quite likely advantageous to use a quadratic sorting algorithm, say Selection Sort, rather than continue the recursive calls of Merge Sort, when the size of the list being sorted gets small enough. Let m be a constant. Assume we run Merge Sort on a list of size n, but when the list has size $\leq m$ we use Selection Sort. Merge Sort could be considered the special case of this algorithm for $m = 1$.

(a) Write pseudo-code for this generalized Merge Sort algorithm. You can assume that the Merge and Selection Sort routines are given to you (so you do not have to write them).

(b) Write a recurrence for the number of comparisons.

(c) Solve the recurrence using the tree method, assuming that the list size is exactly m times a power of 2 (i.e. $n = 2^k m$, for some integer k). Simplify. Show your work.

Ideally, your final answer should only involve n and m. You should write your final answer as a term involving $n \lg n$, followed by a linear term (in n), followed by a constant term. (You can check that your answer is correct by substituting $m = 1$.)

(d) **Challenge Problem. Will not be graded.** Assume that there is an additive constant α associated with each (recursive) procedure call, a multiplicative constant β associated with Merge, and a multiplicative constant γ associated with Selection Sort. Calculate the optimal value of m.