1. Assume that you execute the randomized selection algorithm searching for the kth smallest element. As assumed in class and in the book, whenever you partition the kth smallest element ends up on the larger side, except there is a twist: There is a probability p, for some constant $0 \leq p \leq 1$, that at each partition you get lucky and the pivot actually is the kth smallest element. So, if $p = 0$ this is exactly the same as before, and if $p = 1$ the kth smallest element will be found after the first partition. Otherwise, if $0 < p < 1$, the kth smallest element will be found after some random partition.
   (a) Analyze randomized selection with this new parameter, p.
   (b) What is the result if $p = 1/3$?
   (c) What is the result if $p = 2/3$?

2.
   (a) Assume you use Selection Sort to find the median of 11 elements. Exactly how many comparisons do you use (in the worst case)?
   (b) Assume you use Mergesort to find the median of 11 elements. Exactly how many comparisons do you use (in the worst case)?

3. It turns out that you can find the Median of 11 elements with 18 comparisons. You can use this information to develop a (worst case) linear time Selection algorithm based on columns of size 11, rather than the columns of size 5 that we used in class.
   (a) Using columns of size 11 exactly how far from either end of the array is the median of medians guaranteed to be. Just give the high order term. (Recall that with columns of size 5 we got $\frac{3n}{10}$.)
   (b) Write down the recurrence for a Selection algorithm based on columns with 11 elements each. (You can ignore floors and ceilings, as we did in class.) You do not have to give the algorithm, but state where each of the terms in your recurrence comes from. (For example, you might say that the $n - 1$ term comes from partition.)
   (c) Solve the recurrence, and give the high order term exactly.