For these problems, assume that you are using Insertion Sort with a sentinel (as done in class). Assume that the first and second smallest elements are next to each other (somewhere in the input), the third and fourth smallest elements are next to each other, the fifth and sixth smallest elements are next to each other, etc. For example,

50, 60, 40, 30, 90, 100, 10, 20, 80, 70

(The algorithm does not know this, and executes without this extra information.) Assume the problem size n is even. For each problem show your work.

1. Assume each of the above pairs are out of order (so that the second smallest element comes before the smallest, the fourth smallest element comes before the third smallest, etc.). For example,

   60, 50, 40, 30, 100, 90, 20, 10, 80, 70

   What is the exact number of comparisons in the best case (for general n)?

2. Assume each of the above pairs are in order (so that the smallest element comes before the second smallest, the third smallest element comes before the fourth smallest, etc.). For example,

   50, 60, 30, 40, 90, 100, 10, 20, 70, 80

   What is the exact number of comparisons in the worst case (for general n)?

3. Assume each of the above pairs are in a random order. What is the exact number of comparisons in the average case (for general n)?

4. Insertion Sort can be thought of as a recursive algorithm as follows: Sort the first $n-1$ elements recursively, and then put the nth element into its correct location. Write the pseudo code for this recursive version. Use the method without the sentinel to insert elements.