Assignment 7

R-2.8 Illustrate the performance of the selection-sort algorithm on the following input sequence (22, 15, 26, 44, 10, 3, 9, 13, 29, 25).

R-2.9 Illustrate the performance of the insertion-sort algorithm on the input sequence of the previous problem.

R-2.10 Give an example of a worst-case sequence with n elements for insertion-sort that runs in $\Omega(n^2)$ time on such a sequence.

1. Using the pseudo-code in today's notes implement insertionSort, ShellSort, and heapSort in JavaScript. Insert a counter in each of the algorithms to count the number of key comparisons and swaps for heapSort. Similarly, insert a counter for the number of key comparisons and shifts in insertionSort and ShellSort. Run several tests of small, medium, and large arrays to compare the algorithms; use the ArraySort.js file that imports your HW07-ArraySorter.js file and runs tests on your sort algorithms. What is your conclusion about running times?

Level 2:

2. Use one of these sorts from 1 above to implement in JavaScript another version of isPermutation(A,B). First start with the pseudo-code, then translate into JavaScript. Submit both the pseudo-code and the JavaScript program file with test cases.