CS435 Homework Assignment 9

Due date: ____________________

Turn in all code on both paper and by email (to dbahr@regis.edu with “CS435 Homework” in the subject line).

Please include comments in all of your code. Describe what your function does, and describe anything that is tricky or unclear. More comments are better than fewer comments.

Full credit requires comments.

Problem #1: By hand (not code), sort the sequence 3, 1, 4, 1, 5, 9, 2, 6, 5 using insertion sort. Show your work.

Problem #2: What is the big-O running time for insertion sort if all of the elements are equal. Give a brief explanation why.

Problem #3: By hand (not code), sort the sequence 9, 8, 7, 6, 5, 4, 3, 2, 1 using Shellsort with the increments 7, then 3, then 1.

Problem #4: Copy the code presented in class for Shellsort. Now alter the code to use Hibbard’s increments 1, 3, 7, …, $2^k - 1$.

Problem #5: Write two sorting programs, one that uses insertion sort and one that uses Shellsort. For insertion sort, you may copy the code presented in class. For Shellsort, use the code from problem #4.

Problem #6: Generate a large array of random numbers (say N = 1000, 10000, or 100000) and sort the array using your programs for (1) insertion sort, and (2) Shellsort. Compare the run times of your two programs using a stopwatch (or code). Which was faster and by how much?

Hint: in Java, “System.currentTimeMillis()” will give the current time in milliseconds (measured from midnight, Jan. 1, 1970).

Note: Use the same array for each program. The random number generator in Java (and C) will generate the same array as long as you use the same seed each time. See the Java API for details.
Also, to compare the times, you will need to pick an array size that is large enough that it takes measurable time to run. E.g., an array of size 10 will be sorted too quickly to measure. On the other hand, an array of size $1 \times 10^9$ will take too long to sort.

**Problem #7:** This is the same as problem #6, but use an array that is already sorted and goes 1, 2, 3, 4, 5, 6, … Sort the array using your programs for (1) insertion sort, and (2) Shellsort. Compare the run times of your two programs using a stopwatch (or other method). Which was faster and by how much?