Project 1

(Due Mar 5/12, 2017)

Description:

In this project, we implement the *insertion sort* in two versions: the *linear-search-based* version and the *binary-search-based* version. Then we will compare their performances and see how much the second version is improved over the first version.

Requirements:

1. (*Programming Languages*) You can use either C/C++ or Java to write this program. If you want to use other programming languages, you may need to talk to me first.

2. (*Generate Data*) First you need to generate the data. Our data is a 1000-element array. You can use a random function to generate an integer array containing 1000 random integers with values between 1 and 1000 inclusive. The array elements do not have to be distinct.

3. (*Display Data*) Before you sort the array, you need to display it on the screen. You display these 1000 integers in 50 lines with each line containing 20 numbers, separated by commas.

4. (*Develop*) You need to develop one function/method for the *linear-search-based* insertion sort and one function/method for the *binary-search-based* insertion sort. For each function/method, the input is the array to be sorted, and the output is the sorted array.

5. (*Count operations*) In this algorithm, you will count two types of operations: *comparisons* and *assignments*. In each of your insertion sort implementation function/method, create two counters to count the number of those two operations. For these counters, you can use global variables or instance variables when necessary.

6. (*Display results*) Run two versions of the insertion sort one after another. When you run each version, print out the number of comparisons, the number of assignments, and then the sorted array in the same way as above.

7. (*Display running time*) When you run each version, clock the running time, and display it for comparison. We want to see if the second version has significant improvement in running time.