Homework 2 – Due March 20th 23:59 KST

Instructions: Complete the report and turn it in before the due date. Any deviations from the instructed deliverable format will result in a deduction of grade. DO NOT COPY OTHER'S WORKS!

The answers to the following questions must be accompanied by appropriate justification. You will receive no credit if you do not show your work (drawing a graph does not count, except in #4).

1. Compute the tightest order, in big-O notation, of the following functions and codes. For A and B, you must follow the rigorous derivation steps we did in class. For C and D, first write the code before providing an analysis (5 pts x 4 = 20 pts).

   A. $\mathrm{f(n)} = \sin(\mathrm{n}) - 10000\mathrm{n}^3 + 10^{-10}\mathrm{n}^4$

   B. $\mathrm{f(n)} = \sqrt{\mathrm{n}} + n\log_2 n + \log_{10} n^2$ (It's okay to use graphing software to find where the logarithm and the square root functions cross each other. Just don't use the graph as the entire justification for your answer)

   C. A Java method that takes an integer array 'x' as input and returns the second-smallest element, which is guaranteed to exist in 'x'.

   D. A method that takes two 2D arrays as inputs and returns another 2D array that contains the element-wise sums of the two inputs.

2. Attach your code for HW1's moveTo() implementation. Based on YOUR code, analyze the big-O complexity of that method. The style of analysis should be similar to what we did for the analysis of InsertionSort. Please attach the code!!! We will NOT grade your work if you don't. By 'analysis', I mean accounting for every single line of code you have, not just the overall description (20 pts).

3. Open Complexity.java. Analyze the time complexities of the methods a() and b(). (10 pts)

4. Now execute Complexity.java multiple times with varying values for IN_SIZE to collect the running times of methods a() and b(). Draw a plot containing two curves, one for a() and one for b(), that depicts the actual running time (y-axis) in milliseconds vs. the size of the input (x-axis). What kind of trend can you find, and how does it compare to the theoretical analysis you gave in #3? Don't just say 'They're the same' or 'They're different' but try to describe the qualitative properties of the plots. Attach the plot as well. CAUTION: The overall running time will get larger as you increase IN_SIZE. Be patient! (10 pts)

5. Implement the following count() method that performs as follows: Given two arrays 'a' and 'b', this algorithm should return the number of items in 'a' whose value is larger than the

sum of elements in 'b'. (e.g., If a = {1, 12, 3, 6, 10, -1, 5} and b = {1, 2, 3}, then count(a, b) should return 2, because only 12 and 10 are greater than 6=1x2x3). Provide two versions of your algorithm: One running in linear time, and the other running in quadratic time. Justify your time complexities by analyzing your codes (20 x 2 = 40 pts).

```
public static int linearCount(int[] a, int[] b) {
    // Fill here
}
public static int quadraticCount(int[] a, int[] b) {
    // Fill here
}
```

Deliverables

- A single HW2.pdf file containing your name and SBU ID. The PDF file must contain all the answers to the problems above in a ***typed*** format. That is, don't hand-write your answers and attach the scanned version – it's very hard to read poorly handwritten text. Do not turn in .java files for the coding problems: just include the code in your final PDF file.
- All codes you provide must be error-free, including syntax errors.