CS435 Homework Assignment 2

Due date: __________________

Turn in all code on both paper and by email (to dbahr@Regis.edu with “CS435 Homework” in the subject line). *Please thoroughly comment your code* so that I can tell what it is doing.

___________________________________________________________________________________________

**Problem #1:** Write a method (inside of a complete program) that finds $\frac{1}{4^i}$ for any specified $i$. Calculate the run time, $T(N)$, and the growth rate (big-O) of this method.

Note: Please do not use any pre-written methods for calculating $4^i$ (e.g., Math.pow(4,i)). Why? Because you don’t know how long those methods take to run. Instead use something like $4*4$ for $4^2$ and $4*4*4$ for $4^3$, etc. Similarly, inside your method, do not use a JOptionPane, System.out.println, or any other Java methods because we don’t know how long it takes them to run. Instead, put all of your input and output and other Java methods in the main.

Also note: There is no summation in this problem.

**Problem #2:** Write a method (inside of a complete program) that finds $\sum_{i=0}^{N} i^2$ for any specified $N$. Calculate the run time, $T(N)$, and the growth rate (big-O) of this method.

Note: Please do not use any pre-written functions for calculating $i^2$.

**Problem #3:** An algorithm takes 0.5ms for input size 100. How long will it take for input size 500 if the running time is the following?

(a) $N$
(b) $N^2$
(c) $N^3$
(d) $N \log_{10} N$

Hint: This is a really cool application of big-O. You can run a test with a small number of inputs and then predict how long it will take for a real data set with lots
of values. For $N^5$ you could set up the problem this way:

$$\frac{(100 \text{ inputs})^5}{\text{takes } 0.5 \text{ ms}} = \frac{(500 \text{ inputs})^5}{\text{takes } x \text{ ms}}.$$ Now solve for $x$.

**Problem #4:** An algorithm takes 0.5ms for input size 100. How large an input can be solved in 1 min if the running time is the following?

(a) $N$
(b) $N^2$
(c) $N^3$

Hint: Set this up similarly to problem #3.

**Problem #5:** Calculate the growth rates (big-O) for the following code fragments.

(a) 
```
sum = 0;
for(i=0; i<n; i++)
    sum++;
```

(b) 
```
sum = 0;
for(i=0; i<n; i++)
    for(j=0; j<n, j++)
        sum++;
```

(c) 
```
sum = 0;
for(i=0; i<n; i++)
    for(j=0; j<n*n, j++)
        sum++;
```

**Problem #6:** Order the following functions by growth rate: $N$, $\sqrt{N}$, $N^{1.5}$, $N^2$, $N \log N$, $N \log(\log N)$, $N \log^2 N$, $N \log(N^2)$, $2/N$, $2^N$, $2^{N/2}$, 37, $N^2 \log N$, $N^3$. Indicate which functions grow at the same rate.

Hint: You can plug in numbers (or plot these) to compare, but they had better be very, very, very, VERY big numbers. Why? Because growth rates are only concerned about the answer at very large values. At “smaller” values, you might get the wrong answer. For example, you know that $N$ is better than $N^2$, but $N^2$ gives the larger value at 0.5.

So what should you do? Suppose you are trying to decide if one of the functions is bigger or smaller than another. When the answers aren’t obvious, your best
approach is to do some math. For example, use the fact that $(\log N)^k = O(N)$. Recall that this means $(\log N)^k \leq N$ for big enough values of $N$. That can be really handy! You can also compare two functions directly. For example, set $N \log(\log N) \leq N$ and reduce. Is that equation true or false for all really big values of $N$? If true, then $N \log(\log N)$ has a smaller growth rate than $N$. If false, then $N$ has the smaller growth rate.