HW3, Growth Functions and Big O Notation
20 points

Assignment: Complete the exercises below.

Due: Tuesday October 01 at 11:59 pm

Turn In: Submit a file containing your answers to Blackboard Vista.

Note: The first 5 exercises are taken from EX 2.1–EX 2.5 on pages 26-27 of the 7th edition of the Lewis and Chase Java Software Structures book. Questions 6 and 7 are discussed in the book in section 2.4.

1. EX 2.1 What is the order of the following growth functions?
   Note we're using $\log_2$ not $\log_{10}$

   a) $10n^2 + 100n + 1000$
   b) $10n^3 - 7$
   c) $100n^3 + 2^n$
   d) $n^2 \log_2 n$

2. EX 2.2 Arrange the growth functions of the previous exercise in ascending order of efficiency for n=10 and again for n=1,000,000.

3. EX 2.3 Write the code fragment necessary to find the largest element in an unsorted array of integers. What is the time complexity of this algorithm? (don't worry about writing a runnable program – just write the algorithm that finds and prints out the largest value in the unsorted array. You can assume that you're handed an array of unsorted integers to work with. We'll be looking at your algorithm and the time complexity you give for an answer, more than syntax, but it would be nice to use proper syntax for the fragment.

4. EX 2.4 Determine the growth function and order of the following code fragment:

```
for (int count=0; count<n; count++)
{
    for (int count2=0; count2<n; count2=count2+2)
    {
        System.out.println(count + ", " + count2);
    }
}
```

5. EX 2.5 Determine the growth function and order of the following code fragment:

```
for (int count=0; count<n; count++)
{
    for (int count2=1; count2<n; count2=count2*2)
    {
        System.out.println(count + ", " + count2);
    }
}
```

6. Determine the growth function and order of the following code fragment that prints the sum of numbers from i to n :

Note that you need to look at the order of the method called as well as the order of the for loop

```
for (int count = 0; count < n; count++)
 {
      printsum(count);
 }
```

Here's the method:

```
public void printsum(int count) {
        int sum = 0;
        for (int i=1; i<count; i++)
        {
          sum +=i;
        }
        System.out.println(sum + ": " + sum);

}
```

7. Determine the growth function and order of the following code fragment that prints the sum of numbers from i to n :

Note that you need to look at the order of the method called as well as the order of the for loop. ***Note that this method is different from above printsum method.***

We know from past examples that the sum of all numbers from i to n is an arithmetic progression which can be expressed in *closed form* as n(n+1)/2
We have rewritten the code fragment above that prints the sum of all numbers from i to n:

```
for (int count = 0; count < n; count++)
 {
      printsum(count);
 }
```

Here's the method:

```
public  void printsum(int count) {
                int sum = 0;
                sum = count * (count + 1)/2;
                System.out.println(“sum : " + sum);
 }
```