CS 385, Homework 1: Analysis of Algorithms

Name: John Spicer Date: 9/23/15

Point values are assigned for each question. Points earned: ____ / 42, = ____ %

1. Use the definitions of $O$, Θ, and Ω to determine whether the following assertions are true or false. If true, give values for the appropriate constants. If false, explain the contradiction. (3 pts. each)

   a. $n(n + 1)/2 \in O(n^3)$ True: c = 1, $n_0 = 1$

   b. $n(n + 1)/2 \in O(n^2)$ True: c = 1, $n_0 = 1$

   c. $n(n + 1)/2 \in \Theta(n^3)$ False: for n >= 1, f(n) ! >= 1/2($n^3$)

   d. $n(n + 1)/2 \in \Omega(n^3)$ True: c = ½, $n_0 = 0$

2. Write the following asymptotic efficiency classes in **increasing** order of magnitude.
   $O(n^2)$, $O(2^n)$, $O(1)$, $O(n \lg n)$, $O(n!)$, $O(n^3)$, $O(\lg n)$, $O(n^n)$, $O(n)$ (1 pt. each)

   _O(1)_,_O(lgn)_,_O(n)_,_O(nlgn)_,_O(n^2)_,_O(n²lgn)_,_O(n³)_,_O(2ⁿ)_,_O(n!)_,_O(nⁿ)_

3. Determine the largest size *n* of a problem that can be solved in time *t*, assuming that the algorithm takes *f(n)* milliseconds. (1 pt. each)

   a. *f(n)* = $n$, *t* = 1 second _____1000_____

   b. *f(n)* = $n \lg n$, *t* = 1 hour $\sqrt{(2^{3600000})}$

   c. *f(n)* = n² , *t* = 1 hour $\sqrt{(3600000)}$

   d. *f(n)* = n³ , *t* = 1 day $\sqrt[3]{86400000}$

   e. *f(n)* = $n$!, *t* = 1 minute ___8___

4. (3 pts.) Suppose we are comparing two sorting algorithms and that for all inputs of size $n$ the first algorithm runs in $8n^2$ seconds, while the second algorithm runs in $48n \lg n$ seconds. For which integral values of $n$ does the first algorithm beat the second algorithm? _____{2,3}__

   Explain how you got your answer:

   I solved the system of equations by setting them equal to each other and found the points of intersection, which is the integral between which the first would beat the second algorithm.

5. Give the complexity of the following methods. Choose the most appropriate notation from among $O$, Θ, and Ω. (3 pts. each)

```
int function1(int n){
  int count = 0;
  for (int i = n/2; i <= n; i++){
    for (int j = 1; j <= n; j *= 2){
      count++;
    }
  }
  return count;
}
```

Answer: Θ(nlgn)

```
int function2(int n){
  int count = 0;
  for (int i = 1; i*i <= n; i++){
    count++;
  }
  return count;
}
```

Answer: Θ($\sqrt{(n)}$)

```
int function3(int n){
  int count = 0;
  for (int i = 1; i <= n; i++){
    for (int j = 1; j <= n; j++){
       for (int k = 1; k <= n; k++){
        count++;
      }
    }
  }
  return count;
}
```

Answer: Θ($n^3$)

```
int function4(int n){
  int count = 0;
  for (int i = 1; i <= n; i++){
    for (int j = 1; j <= n; j++){
      count++;
      break;
    }
  }
  return count;
}
```

Answer: Θ(n)