I pledge my honor that I have abided by the Stevens Honor System.
– Khayyam Saleem

CS 385, Homework 1: Analysis of Algorithms

Name: Khayyam Saleem Date: 9/26/2016

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

1. Find an upper bound for $f(n) = n^4 + 10n^2 + 5$. Write your answer here: O(n^4) (2 points)

   Prove your answer by giving values for the constants c and n_0. Choose the smallest integral value possible for c. (2 points)

   c = 2; n0 = 4

2. Find an asymptotically tight bound for $f(n) = 2n^2 - n$. Write your answer here: θ(n^2) (2 points)

   Prove your answer by giving values for the constants c_1, c_2, and n_0. Choose the tightest integral values possible for c_1 and c_2. (3 points)

   c1 = 1; c2 = 2; n0 = 1

3. Is $3n - 4 \in \Omega(n^2)$? Circle your answer: yes / no. (1 point)

   If yes, prove your answer by giving values for the constants c and n_0. Choose the smallest integral value possible for c. If no, derive a contradiction. (2 points)

   From the computation shown, a contradiction arises when n is determined to be bounded by 3/c if we assume that it is an element of the given efficiency class.

   3n-4 ≥ cn^2
   3n ≥ cn^2
   3 ≥ cn
   3/c ≥ n

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

   O(1), O(lg n), O(n), O(n * lg n), O(n^2), O(n^2*lg n), O(n^3), O(2^n), O(n^n), O(n!)

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

   a. *f(n)* = n, *t* = 1 second 1,000

   b. *f(n)* = $n \lg n$, *t* = 1 hour 204,095

   c. *f(n)* = n^2, *t* = 1 hour 1,897

   d. *f(n)* = n^3, *t* = 1 day 442

   e. *f(n)* = $n!$, *t* = 1 minute 8

6. Suppose we are comparing two sorting algorithms and that for all inputs of size n the first algorithm runs in $4n^3$ seconds, while the second algorithm runs in $64n \, lg \, n$ seconds. For which integral values of n does the first algorithm beat the second algorithm? n ≥ 7; n ∈ ℤ (2 points)

Explain how you got your answer or paste code that solves the problem (1 point):

```
i = 2
while 4*i**3 < 64*i*math.log*i, 2):
  i += 1
print i
```

7. Give the complexity of the following methods. Choose the most appropriate notation from among O, Θ, and Ω. (3 points 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: $\Theta(n\lg(n))$

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

Answer: $\Theta(\sqrt[3]{(n)})$

```
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: $\Theta(n^3)$

```
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: $\Theta(n)$