CS 385, Homework 1: Analysis of Algorithms

Name: Max Copeland Date:9/23/17

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)

   $n^4 + 10n^2 + 5 \leq cn^4$

   $n^4 + 10n^2 + 5 \leq 2n^4$

   $c = 2$ for all values $n \geq 4$

   $n_0 = 4$

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

   $c_1n^2 \leq 2n^2 - n$

   $c_1 = 1$ for all values $n \geq 1$

   $2n^2 - n \leq c_2n^2$

   $c_2 = 2$ for all values $n \geq 1$

   $n_0 = 1$

3. Is $3n - 4 \; \epsilon \; \Omega\left(n^2\right)$? 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)

   $0 \le cn^2 \le 3n - 4 \le 3n$

   $cn^2 \le 3n \rightarrow n \le \frac{3}{c}$

   Contradiction because $\frac{3}{c}$ will never be an integer except when $c = 1{,}3$

4. Write the following asymptotic efficiency classes in **increasing** order of magnitude.

   $O(n^2)$, $O(2^n)$, $O(1)$, $O\left(n\lg n\right)$, $O(n)$, $O(n!)$, $O(n^3)$, $O\left(\lg n\right)$, $O\left(n^n\right)$, $O\left(n^2\lg n\right)$ (1 point each)

   $O(1), O(lgn), O(n), O(nlgn), O(n^2), O(n^2lgn), O(n^3), O(2^n), O(n!), O(n^n)$

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 —> 1000

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

   c. *f(n)* = n^2, *t* = 1 hour —> 1897

   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? (2 points)

   For all $n \geq 1$

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

   I plugged in numbers until the second algorithm took more time than the first (it didn't take long)

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

Answer: $O(nlgn)$

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

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

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

```
}
```

Answer: $O(n^3)$

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

Answer: $O(n)$