"I pledge my honor that I have abided by the Stevens Honor System." JNelson

CS 385, Homework 1b: Analysis of Algorithms

Name: Julia Nelson Date: 02/11/2020

Point values are assigned for each question. Points earned: ____ / 100, = ____ %

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

   Prove your answer by giving values for the constants c and n_0. Choose the smallest integral value possible for c. (4 points)

   $n^4+10n^2+5 \le 2n^4$ $(\forall n \geq 4)$

   $n^4+10n^2+5 \le C \cdot n^4$

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

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

   $\boxed{C=2 \quad n_0=4}$

   $n_0=1$ $15 \le 1$ ✗
   $=2$ $45 \le 16$ ✗
   $=3$ $95 \le 81$ ✗
   $=4$ $165 \le 256$ ✓

2. Find an asymptotically tight bound for $f(n) = 3n^3 - 2n$. Write your answer here: $\Theta(n^3)$ (4 points)

   $C_1 \cdot (n^3) \le 3n^3 - 2n \le C_2 \cdot (n^3)$

   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. (6 points)

   let $C_1=2$

   $C_1 \cdot n^3 \le 3n^3-2n$

   $2n^3 \le 3n^3-2n$ $(\forall n \geq 2)$

   $n=1$ $2 \le 1$ ✗

   $n=2$ $16 \le 20$ ✓

   $n_0=2$

   let $C_2=3$

   $3n^3-2n \le C_2 \cdot n^3$

   $3n^3-2n \le 3n^3$

   $\boxed{C_1=2 \quad C_2=3 \quad n_0=2}$

   $n=0$ ✗
   $n=1$ $3-2 \le 3$ ✓
   $=2$ $24-4 \le 3 \cdot 8$ ✓
   $=3$ $81-6 \le 81$ ✓

   $n_0=1$

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

   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. (4 points)

   For n^2 to be lower bound

   $n^2 \le 3n-4$

   if $n=2 \rightarrow 4 \le 2$ ✗
   $n=1 \rightarrow 1 \le -1$ ✗
   $n=1/2 \rightarrow 1/4 \le -2.5$
   $n=1/4 \rightarrow 1/16 \le$

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

   $O(n^2)$, $O(2^n)$, $O(1)$, $O(n \lg n)$, $O(n)$, $O(n!)$, $O(n^3)$, $O(\lg n)$, $O(n^n)$, $O(n^2 \lg n)$ (2 points each)

   $O(1)$, $O(\lg n)$, $O(n)$, $O(n \lg n)$, $O(n^2)$, $O(n^2 \lg n)$, $O(n^3)$, $O(2^n)$, $O(n!)$, $O(n^n)$

5. Determine the largest size n of a problem that can be solved in time t, assuming that the algorithm takes $f(n)$ milliseconds. n must be an integer. (2 points each) 1 s = 1000 ms

   a. $f(n) = n$, t = 1 second 1000 n milliseconds

   b. $f(n) = n \lg n$, t = 1 hour 204094 1 hr = 3600000 ms

   $n \lg n \le 3600000$

   $2^{n \lg n} = 2^{3600000}$

c. $f(n) = n^2$, $t = 1$ hour 1897

$n^2 = 3600000$
$= \sqrt{3600000}$

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

$n^3 = 86400000$
$= \sqrt[3]{86400000} =$

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

$1 \text{min} = 60000 \text{ ms}$
$n! = n \times (n-1)!$

$1 \times 2 \times 3 \times 4 \times 5 = 120$
$120 \times 6 \cdot 7 = 5040 \cdot 8 \checkmark$
$= 40320 \times$
$\times 9 = 362880$

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? ~~[illegible]~~ $2 \leq n \leq 6$ (4 points)

Explain how you got your answer or paste code that solves the problem (2 point):

~~graphed [illegible] and looked for which~~

graphed and looked for where they crossed

7. Give the complexity of the following methods. Choose the most appropriate notation from among O, Θ, and Ω. (8 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;
}
```

(cuberoot)

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: $\Theta(n)$

```
int function5(int n) {
    int count = 0;
    for (int i = 1; i <= n; i++) {
        count++;
    }
    for (int j = 1; j <= n; j++) {
        count++;
    }
    return count;
}
```

Answer: $\Theta(n)$