Name: Brian Sampson Date: 09/25/22

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)

$Using\ \ c = 2$
$0 \leq n^4 + 10n^2 + 5 \leq 2n^4$
$Solving\ for\ n_0$
$n_0 = 4$

So this means that the upper bound is $O(n^4)$

2. Find an asymptotically tight bound for $f(n) = 3n^3 - 2n$. Write your answer here: $\Theta(n^3)$ (4 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. (6 points)

   $c_1 = 2$
   $c_2 = 3$
   $n_0 = 2$

   $2n^3 \leq 3n^3 - 2n \leq 3n^3$
   $when\ n_0 = 2$
   $16 \leq 20 \leq 24$

3. Is $3n - 4 \in \Omega\left(n^2\right)$? 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 not, derive a contradiction. (4 points)

   $3n - 4\ cannot\ be\ \ within\ \Omega(n^2)\ because\ it's\ better\ than\ the\ best\ case\ which\ does\ not\ make\ sense.$

   $cn^2 \leq 3n - 4$
   $cn^2 \leq 3n - 4 \leq 3n$
   $cn^2 \leq 3n$
   $cn \leq 3$

$n \leq \frac{3}{c} \forall n \geq n_0$

$This\ \ shows\ how\ 3n\ -\ 4\ is\ not\ within\ the\ best\ case\ of\ \ n^2\ because\ n\ \leq \frac{3}{c}\ is\ not\ \ possible.$

$Proved\ by\ contradiction$

4. Write the following asymptotic efficiency classes in **increasing** order of magnitude. (2 points each)

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

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

5. Determine the largest size *n* of a problem that can be solved in time *t*, assuming that the algorithm takes *f(n)* milliseconds. Write your answer for *n* as an integer. (2 points each)

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

   b. *f(n)* = $nlg(n)$, *t* = 1 hour 204,094

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

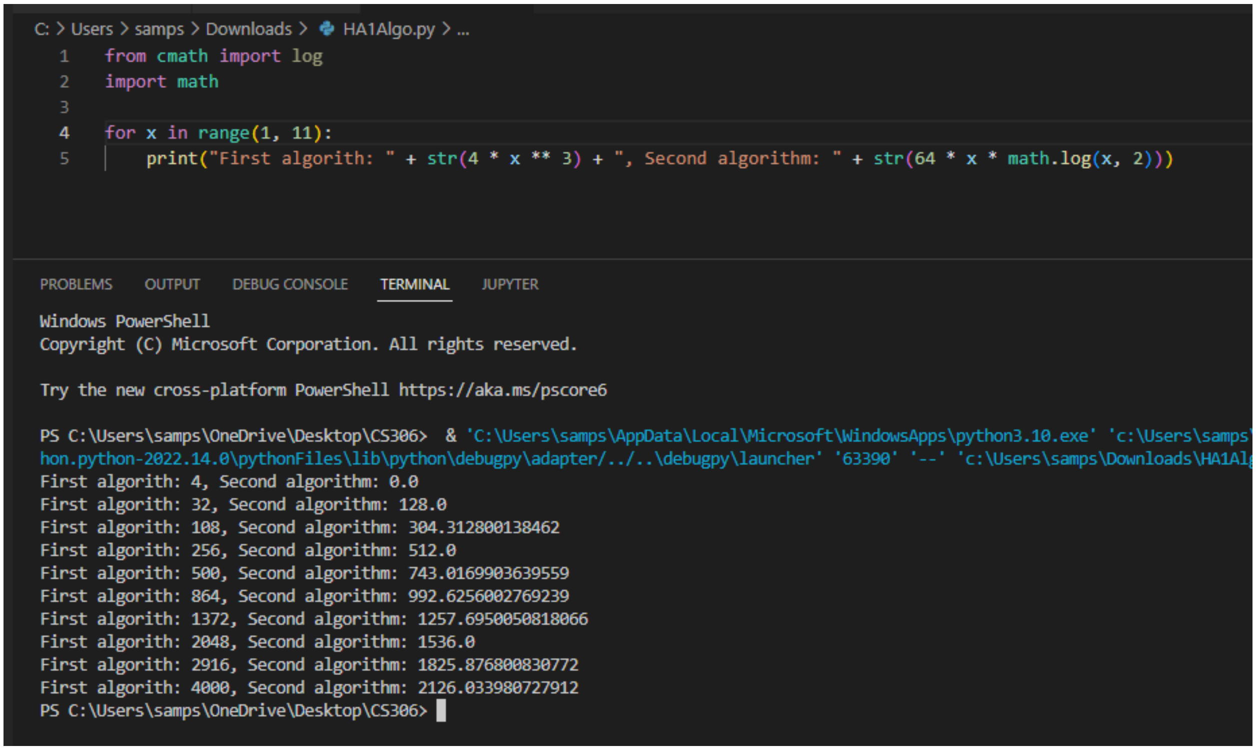
   d. *f(n)* = n^3, *t* = 1 day 442

   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?

   $2 \leq n \leq 6$

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



C: > Users > samps > Downloads > HA1Algo.py > ...

```python
from cmath import log
import math

for x in range(1, 11):
    print("First algorith: " + str(4 * x ** 3) + ", Second algorithm: " + str(64 * x * math.log(x, 2)))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\samps\OneDrive\Desktop\CS306>  & 'C:\Users\samps\AppData\Local\Microsoft\WindowsApps\python3.10.exe' 'c:\Users\samps
hon.python-2022.14.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '63390' '--' 'c:\Users\samps\Downloads\HA1Al
First algorith: 4, Second algorithm: 0.0
First algorith: 32, Second algorithm: 128.0
First algorith: 108, Second algorithm: 304.312800138462
First algorith: 256, Second algorithm: 512.0
First algorith: 500, Second algorithm: 743.0169903639559
First algorith: 864, Second algorithm: 992.6256002769239
First algorith: 1372, Second algorithm: 1257.6950050818066
First algorith: 2048, Second algorithm: 1536.0
First algorith: 2916, Second algorithm: 1825.876800830772
First algorith: 4000, Second algorithm: 2126.033980727912
PS C:\Users\samps\OneDrive\Desktop\CS306>
```

As we can see, the value for the first algorithm is smaller only on intervals from 2 to 6 (inclusive). Everything outside of these bounds, the second algorithm is faster.

7. Give the complexity of the following methods. Choose the most appropriate notation from among O , Θ, and Ω. (8 points each)

```c
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: $\Theta(nlog(n))$

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

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

```c
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: $\theta(n)$