Name: __Daniel Detore____ Date: __9/25/2024______

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

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

1. Find a tight 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 integer value possible for c. (4 points)
   c = 2, n_0 = 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 integer values possible for c_1 and c_2. (6 points)

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

3. Is $3n - 4 \epsilon \Omega(n^2)$? Circle your answer: yes / $\boxed{\textbf{\textit{no}}}$. (2 points)

   If yes, prove your answer by giving values for the constants c and n_0. Choose the smallest integer value possible for c. If no, derive a contradiction. (4 points)

   If $3n - 4 \epsilon \Omega(n^2)$, then $\exists c \in \mathbb{Z}$ where $0 \leq cn^2 \leq 3n - 4$ $(\forall n \geq n_0)$.
   We also have $3n - 4 \leq 3n + 4n = 7n$
   Combining these statements gives us:

   $$cn^2 \leq 7n$$
   $$cn \leq 7$$
   $$n \leq \frac{7}{c}$$

   Because n can grow to infinity, there is no way to bound it by any constant $\frac{7}{c}$. $\therefore 3n - 4 \notin \Omega(n^2)$.

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(nlogn), O(n^2), O(n^2logn), 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 n = 1000

   b. *f(n)* = $n \lg n$, *t* = 1 hour n = 204094

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

d. *f(n)* = n^3, *t* = 1 day n = 9295

e. *f(n)* = $n!$, *t* = 1 minute n = 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 $64nlg(n)$ seconds. For which integer values of n does the first algorithm beat the second algorithm? $\boxed{\boldsymbol{n \geq 1}}$ (4 points)
Explain in detail how you got your answer or paste code that solves the problem (2 point):

```
import math

for n in range(1, 10000):
    if ((4*math.pow(n,3)) < (64*n*math.log2(n))):
        print(n)
        break
```

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: $\boldsymbol{\Theta(nlgn)}$

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

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

```
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: $\Theta(n)$