Name: Natalie Zoladkiewicz Date: September 25

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)
   $n^4 + 10n^2 + 5 \leq$ C(n^4)
   C = 2, n0 = 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(n^3) <= 3n^3 – 2n <= c_2(n^3)
   c_1 = 2
   c_2 = 3
   c_3 = 2

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 integer value possible for c. If no, derive a contradiction. (4 points)

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. Write your answer for *n* as an integer. (2 points each)

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

   b. *f(n)* = $n \lg n$, *t* = 1 hour $\sqrt{2^{3600000}}$

   c. *f(n)* = n^2, *t* = 1 hour $\sqrt{3600000}$

   d. *f(n)* = n^3, *t* = 1 day $\sqrt[3]{86400000}$

   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 integer values of n does the first algorithm beat the second algorithm? **Between 2 and 6 inclusive** (4 points)
   Explain in detail how you got your answer or paste code that solves the problem (2 point):
   I drew a visual of the two algorithms and then found the values where the second equation was under the first equation.

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: O (n log 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: $\Theta(n)$

```
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: $\Theta(n)$