You will submit your solution to this assignment to the Curator System (as `HW03`). Your solution must be either a plain text file (e.g., NotePad++) or a typed MS Word document; submissions in other formats will not be graded.

If you work with a partner, be sure to put both the name and PID of each partner at the beginning of the file you submit.

Except as noted, credit will only be given if you show relevant work.

In all questions about complexity, functions are assumed to be nonnegative.

1. [32 points] Suppose that an algorithm takes 50 seconds for an input of 2^{20} elements. How long would the same algorithm, running on the same hardware, take if the input contained 2^{24} elements, and the algorithm's complexity function is:

a) $\Theta(N)$
b) $\Theta(\log N)$
c) $\Theta(N^2)$
d) $\Theta(N^3)$

Assume that the low-order terms of the complexity functions are insignificant, and state your answers to the nearest tenth of a second. Be sure to show supporting work.

2. [12 points] There are two algorithms for solving the same problem. We know that algorithm A is $\Theta(N^2)$ and algorithm B is $\Theta(N \log N)$. Moreover, we know that the exact complexity functions have dominant terms $N^2/2$ and $1000N \log N$, respectively.

Haskell Hoo IV says that since the coefficient for the dominant term is so much larger for algorithm B than for algorithm A, we should choose algorithm A. Under what precise conditions, if any, would Haskell Hoo IV be correct? Justify your conclusion carefully.

3. [24 points] Use theorems from the course notes to solve the following problems. Show work to support your conclusions.

a) Find the "simplest" function g(n) such that
$$f(n) = 17n^2 + 3n\log n + 1000 \text{ is } \Theta(g(n))$$

b) Find the "simplest" function g(n) such that
$$f(n) = 5n^2 \log n + 8n\log^2 n \text{ is } \Theta(g(n))$$

c) Find the "simplest" function g(n) such that
$$f(n) = \sqrt{n} + \log n \text{ is } \Theta(g(n))$$

4. [16 points] Using the counting rules from the course notes, find the exact-count complexity function T(n) for the following algorithm. Show details of your analysis, and simplify your answer.

```
for (r = 1; r <= n; r++) {
   a[r][1] = 1;
   for (c = 2; c <= 2*n; c += 2) {
      a[r][c] = c * c;
   }
}
```

5. [16 points] Use the definitions of O and Ω to prove that: $f(n)$ is $O(g(n))$ if and only if $g(n)$ is $\Omega(f(n))$