Problem 1. Find values for v and n₀ (according to the definition of $O()$) for $f(n)$ is $O(g(n))$, where:

a. $f(n) = 7n$  
   $g(n) = \frac{n}{10}$

b. $f(n) = 1000$  
   $g(n) = 3n^3$

c. $f(n) = 7n^2 + 3n$  
   $g(n) = n^4$

d. $f(n) = n + 2n \log n$  
   $g(n) = n \log n$

Problem 2. True or false?

a. $f(n)$ is $\Theta(g(n))$ implies $f(n)$ is $O(g(n))$

b. $f(n)$ is $\Theta(g(n))$ implies $g(n)$ is $\Theta(f(n))$

c. $f(n)$ is $\Omega(g(n))$ implies $f(n)$ is $O(g(n))$

Problem 3. Find functions $f(n)$ and $g(n)$ such that $f(n)$ is $O(g(n))$ and the constant $c$ for the definition of $O()$ must be $> 1$. That is, find $f$ and $g$ such that $c$ must be greater than 1, as there is no sufficient $n₀$ when $c = 1$.

Problem 4. Write the $O()$ run-time of the functions with the following recurrence relations:

a. $T(n) = 3 + T(n - 1)$, where $T(0) = 1$

b. $T(n) = 3 + T(n/2)$, where $T(1) = 1$

c. $T(n) = 3 + T(n - 1) + T(n - 1)$, where $T(0) = 1$
Problem 5. Prove by induction that
\[
\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}
\]

Problem 6. What’s the \(O()\) run-time of this code fragment in terms of \(n\):

a. int \(x = 0;\)
   for(int \(i = n; i >= 0; i--\))
       if((i % 3) == 0) break;
       else \(x += n;\)

b. int \(x = 0;\)
   for(int \(i = 0; i < n; i++\))
       for(int \(j = 0; j < (n * n / 3); j++\))
           \(x += j;\)

c. int \(x = 0;\)
   for(int \(i = 0; i <= n; i++\))
       for(int \(j = 0; j < (i * i); j++\))
           \(x += j;\)

Problem 7

Find a recurrence for the worst case runtime, then find a closed form for its recurrence.
Problem 8

An algorithm takes 50 steps for input size 100. How many steps will it take for input size 500 if the running time is the following (assume low-order terms are negligible):

1) Linear
2) O(n lg n)
3) quadratic
4) cubic

Extra Credit (30 points)! Must compute!

Write a recursive algorithm to count the number of 1s in the binary representation of an integer... Then prove using induction that your algorithm works.