Due Wednesday, October 4th, 4:00 pm in 2131 Kemper
For inductive proofs, you must state when you use your inductive hypothesis in your proof.

1. (2 points) Let \( F_i \) be the Fibonacci numbers, where \( F_0 = 1, F_1 = 1, F_2 = 2, F_3 = 3, F_4 = 5 \) …. Prove \( \sum_{i=1}^{N-2} F_i = F_N - 2 \), where \( N > 2 \) using induction. (Lipschutz, HW 3)

2. (2 points) Use induction to prove that for all natural numbers \( x > 1 \) and \( n \), \( x^n - 1 \) is divisible by \( x - 1 \). (Heileman, p.415)

3. (2 points) Prove by induction that
\[
1^2 + 2^2 + 3^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}.
\]
(Lipshutz)

4. (2 points) Prove that \( \sum_{k=0}^{n} k \cdot 2^k = (n-1)2^{n+1} + 2 \). (Heileman, p. 415)

5. (2 points) Assuming \( a \) and \( b \) are arbitrary constants, and \( 0 < a < 1 < b \), order the following functions by growth rate:
\[
\log n, \log(\log n), n \log n, n^b, n^a, \frac{1}{n}, \frac{1}{\log n}, n^n, b^n, I, n^{\log n}, b^{b^n}, \frac{1}{b^n}.
\]
(Heileman p. 27)

6. (2 points) The number of operations executed by algorithms A and B is \( 8n \log n \) and \( 2n^2 \), respectively. Determine an \( n_0 \) such that \( A < B \) for \( n > n_0 \). (Goodrich, p.185)

8. (4 points) Prove that \( x^2 + 3x - 10 \) is \( \Theta(N^2) \). This actually involves two proofs. (Lipshutz HW 3)

9. (4 points, 2 points each) Assuming that \( f_i(n) \) is \( O(g_i(n)) \) and \( f_2(n) \) is \( O(g_2(n)) \): (Drozdek, p.71)
   a. Find a counterexample to refute that \( f_1(n) - f_2(n) \) is \( O(g_1(n) - g_2(n)) \) by supplying \( f_1, f_2, g_1, \) and \( g_2 \).
   b. Find a counterexample to refute that \( f_1(n) / f_2(n) \) is \( O(g_1(n) / g_2(n)) \) by supplying \( f_1, f_2, g_1, \) and \( g_2 \).

10. (2 points) Show that \( \log(n + 1) = O(\log n) \). (Heileman p.28)

11. (12 points, 2 points each) Find the computational complexity for the following code fragments: (a Nyhoff p. 364, b-e Drozdek pp. 72-73, f Weiss, p. 72)
   a. for(int x = 1, count = 0, i = 0; i < n; i++)
      {
         for(int j = 0; j <= x; j++)
            count++;
         x = 2;
      }
   b. for(int count = 0, i = 0; i < n; i++)
      for(int j = 0; j < n; j++)
         count++;
   c. for(int count = 0, i = 0; i < n; i++)
      for(int j = 0; j < i; j++)
         count++;
   d. for(int count = 0, i = 1; i < n; i *= 2)
      for(int j = 0; j < n; j++)
         count++;
   e. for(int count = 0, i = 1; i < n; i *= 2)
      for(int j = 0; j < i; j++)
         count++;
for(int count = 0, i = 0; i < n * n; i++)
    if( i % n == 0)
        for(int j = 0; j < i; j++)
            count++;

12. (4 points, 2 points each) Let \( p(x) \) be a polynomial of degree \( n \), that is, \( p(x) = \sum_{i=0}^{n} a_ix^i \). (Goodrich, p. 190)
   a. Describe a simple \( O(n^2) \) time method for computing \( p(x) \).
   b. Now consider a rewriting of \( p(x) \) as
      \[ p(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \ldots + x(a_{n-1} + xa_n) \ldots))) \]
      which is known as Horner’s method. Using the big-Oh notation, characterize the number of arithmetic operations this method executes.

13. (6 points, 2 points each) Evaluate the following sums: (Weiss, p. 47)
   a. \( \sum_{i=0}^{\infty} \frac{1}{4^i} \)
   b. \( \sum_{i=0}^{\infty} \frac{i}{4^i} \)
   c. \( \sum_{i=0}^{\infty} \frac{i^2}{4^i} \)

Sources of questions: