# Lab 2 continued

1. Prove by induction that for all `n > 7`, $F_n > (\sqrt{2})^n$.

   Hint #1. First verify that $(\sqrt{2})^n = 2^{\frac{n}{2}}$

   Hint #2. For the base case of your proof, note that $F_8 = 21$.

2. More complexity classes:
   a. True or false: $2^n$ is $\Theta(2^{n-1})$. Prove your answer.
   b. True or false: log n is $\Theta(\log_3 n)$. Prove your answer.

3. Below, pseudo-code is given for the recursive factorial algorithm `recursiveFactorial`.

```
Algorithm recursiveFactorial(n)
  Input: A non-negative integer n
  Output: n!
   if (n = 0 || n = 1) then
       return 1
   return n * recursiveFactorial(n-1)
```

   Do the following:
   A. Use the Guessing Method to determine the worst-case asymptotic running time of this algorithm. Then verify correctness of your formula.
   B. Prove the algorithm is correct.

4. Devise an iterative algorithm for computing the Fibonacci numbers and compute its running time. Prove your algorithm is correct.

5. Find the asymptotic running time using the Master Formula:

```
T(n) = T(n/2) + n; T(1) = 1
```

6. *Interview Question.* You are given a length-n array A consisting of 0s and 1s, arranged in sorted order. Give an o(n) algorithm that counts the total number of 0s and 1s in the array. Your algorithm may not make use of auxiliary storage such as arrays or hashtables (more precisely, the only additional space used, beyond the given array, is O(1)). You must give an argument to show that your algorithm runs in o(n) time.