# CS:3330 Homework 1, Spring 2018

## Due in class on Thu, Jan 25

---

1. *Fibonacci numbers* are defined by the recurrence relation: $F_0 = 0$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$ for all $n > 1$. We observed in class (Tue, Jan 16th) that $F_{n+1} \geq 2 \cdot F_{n-1}$ for all $n \geq 1$. Now, to make things convenient for ourselves, let us stipulate that $F_n = 0$ for all integers $n < 0$. With this stipulation, it follows that $F_{n+1} \geq 2 \cdot F_{n-1}$ for all integers $n$. Using this fact, *prove by induction* that

   $$F_{n+1} \geq 2^t \cdot F_{n-(2t-1)}$$

   for all integers $n$ and all integers $t \geq 0$.

   **Hint:** Fix an arbitrary integer $n$ and use $t = 0$ as the base case of your proof. Then assume (as your inductive hypothesis) the claim to be true for some non-negative integer $t = k$ and using this prove the claim for $t = k + 1$.

2. Consider the problem ModuloPower defined below.

   ModuloPower
   Input: Positive integers $a$, $r$, and $p$
   Output: $a^r \bmod p$

   Using ideas discussed in class, write an *efficient* function called `moduloPower`, in a high level programming language of your choice, that takes arguments $a$, $r$, and $p$, and returns $a^r \bmod p$. Then complete your solution by writing a program that reads input $a$, $r$, and $p$ from a textfile called `hw1Prob2.txt` and outputs (to standard output) the answer $a^r \bmod p$. Assume that the input file contains $a$ in the first line, $r$ in the second, and $p$ in the third, with nothing else in the file.

   Finally, use your program to compute $a^r \bmod p$, where $a$ is your university ID, $r$ is the 300-digit number obtained by concatenating the string "8901234567" thirty times and $p$ is the 300-digit number obtained by concatenating the string "2345678901" thirty times.

3. Suppose that algorithm $A$ (for some problem) takes $2^{n/2}$ basic computer steps for an input of size $n$.

   (a) Suppose you run algorithm $A$ on a computer whose processor can complete $10^9$ basic computer steps in 1 second. What is the largest input size for which $A$ can solve the problem in 1 hour of processor time?

   (b) You anticipate that next year you will be able to buy a new computer whose processor is 10 times faster. If you used this new computer next year, what is the largest input size for which $A$ can solve the problem in 1 hour of processor time?

   (c) Suppose that you thought about your problem carefully and were able to come up with a new algorithm $B$ that solved the problem using $n^2$ basic computer steps. What is the largest input size for which $B$ can solve the problem in 1 hour of processor time on your current computer?

4. Solve Problem 0.1 (a)-(j) from the textbook.

5. Analyze the following code fragment and write down the number of basic computer steps performed as a function of $n$. Please show your work in order to receive partial credit.

$j \leftarrow n$
**for** $i \leftarrow 1$ **to** $n$ **do**
    $j \leftarrow 1$
    **while** $j \leq n$ **do**
        print(`"hello"`)
        $j \leftarrow 2 \times j$