# COMP 250: Introduction to Computer Science
# Assignment 2

## Posted Tuesday, February 4, 2014
## Due Friday, February 14, 2014

Please submit the homework through myCourses before midnight on the day it is due.

1. [20 points] **Proofs by induction**

   (a) Prove by induction on $n$ the formula for the geometric series: for any natural numbers $b > 1$, $n > 1$,

   $$\sum_{i=0}^{n} b^i = \frac{b^{n+1} - 1}{b - 1}$$

   (b) Prove by induction that for all positive integers $n$:

   $$\sum_{k=1}^{n} \frac{1}{k(k+1)} = \frac{n}{n+1}$$

   (c) Prove by induction that $n^2 < 2^n, \forall n \geq 4$

   (d) Prove by induction that $8^n - 2^n$ is divisible by 6.

2. [20 points] **Big-Oh**

   (a) Prove that $O(10000n + 10^6) = O(n)$.

   (b) Prove that $3^n$ is not $O(n^3)$.

   (c) Is $O(n \log_2 n)$ in $O(n^2)$? Prove your answer

   (d) Give an example of two function $f$ and $g$ such that $f \notin O(g)$ and $g \notin O(f)$.

3. [15 points] bf More Big-oh

   For the following pieces of code, give the tightest $O()$ estimate that you can, and justify your answer.

   (a) int $sum = 0$;
   for (int $i = 0; i < n; i = i + 2$);
   for (int $j = 0; j < 10; j++$)
   $sum = sum + i + j$;

   (b) int $sum = 0$;
   for (int $i = n; i > n/2; i--$);
   for (int $j = 0; j < n; j++$)
   $sum = sum + i + j$;

(c) int $sum = 0$;
for (int $i = n; i > n - 2; i - -$);
for (int $j = 0; j < n; j+ = 5$)
$sum = sum + i + j$;

4. [25 points] **Recursion**

Write, in Java, a recursive method countBinaryStrings that has one integer parameter $n$ and returns the number of binary strings of length $n$ that do not have two consecutive 0's. For example, for n = 4, the number of binary strings of length 4 that do not contain two consecutive 0's is 8: 1111, 1110, 1101, 1011, 1010, 0111, 0110, 0101. For this problem, your method needs to return *only the number of such strings*, not the strings themselves. You may assume that the integer specified in the parameter is positive. Looking at the example above will give you a hint about how such strings start.

The method should be static and embedded in a class called Recursion. This class should also have a main method. In this case, we will call the main method with an argument, the number of bits $n$. This argument will be in args[0]. You should convert it to an int using the Integer.parseInt method. Look this method up in the Java documentation to see what it does.

5. [20 points] **More recursion**

Suppose we want to compute an exponential function $b^n$ (where $b$ is some base and $n$ is an integer. There is a simple $O(n)$ algorithm for this (multiply $b$ by itself $n$ times). However, in this question you would have to devise a faster algorithm.

(a) [10 points] Devise an algorithm for solving this problem that works in $O(\log_2(n))$.

(b) [10 points] Prove by induction that your algorithm works correctly.