CSC165, Summer 2014
Assignment 6
Weight: 8%
Due Aug. 12th, 12:01 p.m.
No late assignments will be accepted

The goal of this assignment is for you to keep practicing writing proofs.**In your proofs, justify each step**. If you are asked to prove or disprove a claim, first determine whether the claim is true, and then prove that it is true or prove that it is false (i.e., that its negation is true), depending on which is correct.

You may work in groups of no more than two students, and **you should submit a TEX file named `a6.tex` and a PDF file named `a6.pdf` that was produced by compiling your `a6.tex`** and that contains the answers to the questions below. You should submit your Python code in `a6.py`. Submit your answer to the bonus question separately in `bonus6.py`. These files should be submitted using MarkUs.

For this assignment, you will **not** receive 20% of the marks for leaving questions blank or writing "I cannot answer this."

Note: it is possible to obtain 100% on this assignment by doing the non-bonus questions.

1. State whether the following claim is true, and then prove or disprove it. Give a detailed structured proof, justifying every step.

$$\forall n \in \mathbb{N}, [(\exists k \in \mathbb{N}, n = 4k) \vee (\exists k \in \mathbb{N}, n = 4k + 1)]$$

2. Let $\mathcal{F}$ be the set of all function from $\mathbb{N}$ to $\mathbb{R}^+$. Let $\lceil f \rceil$ be a function such that

$$\forall n \in \mathbb{N}, [(\lceil f \rceil)(n) = \lceil f(n) \rceil].$$

   State whether the following claim is true, and then prove or disprove it. Give a detailed structured proof, justifying every step.

$$\forall f \in \mathcal{F}, \forall g \in \mathcal{F}, [f \in \mathcal{O}(g) \Rightarrow \lceil f \rceil \in \mathcal{O}(g)]$$

3. The Fibonacci numbers $fib(n)$ are defined as follows:
   $fib(0) = 1, fib(1) = 1$, and $fib(n) = fib(n-1) + fib(n-2)$ for $n \in \{2, 3, 4, 5, ...\}$.
   Prove that

$$\forall n \in \mathbb{N}, fib(n) \leq 2^n.$$

   It will be helpful to prove, using induction, that $[\forall n \in \mathbb{N}, P(n)]$ where

$$P(n) : \forall k \in \mathbb{N}, [k \leqslant n] \Rightarrow fib(k) \leq 2^k.$$

   .

4. (a) Write a function `def lowest_terms(n,m)` that takes two integers as inputs, and returns True iff n/m is a fraction that is reduced to lowest terms. For example, `lowest_terms(2,3)` is True but `lowest_terms(4,6)` is False. In the comments to your code, explain how the code works, and argue informally (i.e., no formal proof is required) that it produces the desired output. In the comments, provide the output for 8 test cases.

   (b) In the comments in `a6.py`, give a tight upper bound on the total number of comparison operators ($==$, $<$, $>$, $<=$, $>=$) and arithmetic operators ($+$,$-$, `mod`, `/`, $*$,....) performed when running `def lowest_terms(n,m)`. Your answer should be one expression for the tight upper bound on sum of the number of comparison operators and the number of arithmetic operators. Justify your answer. (A formal proof is not required). Note: your answer will be an expression that may depend on both `m` and `n`.

5. **Bonus question, worth half the weight of the other questions:** Claims 5.3 and 5.4 in Section 5.4 in the notes present a method to list all the rational numbers. Note, however, that, if that method is used, every rational number is actually listed an infinite number of times (for example, the number 1 is listed as $1/1, 2/2, 3/3, 4/4, 5/5$...). It is possible to modify this method that produces a list of all the rational numbers such that every rational number appears in the list exactly once. Write a Python function `def r(n)` that takes an integer `n` as input and prints the n-th (starting from 0) rational number in such a list. For example, if the list begins with $\{0, 1, -1, 1/2, -1/2, 2, -2, ...\}$, `r(4)` should print "-1/2" (the output should not contain quotes). In the comments to your code, explain how the code works, and argue informally (i.e., no formal proof is required) that it produces the desired output. Also in the comments, include the output for r(0), r(1), r(2), ..., r(20). *Hints: (1) I suggest implementing code that prints a list using the method in Claim 5.3, then modifying that code to print a list using the method in Claim 5.4, and then thinking how to implement* ***`def r(n)`*** *by modifying the code from there. You only need to submit* ***`def r(n)`*** *and not anything else.* **Clearly documented** attempts at a solution that **run** and make progress towards the solution will get part marks. Submit your code and comments in `bonus6.py`.