CPSC 121: Models of Computation
Assignment #5

Due: Wednesday, November 26, 2014 at 5:00pm **Total Marks: 27**

<u>Submission Instructions</u>-- read carefully

All assignments should be done in groups of 2. It is very important to work with another student and exchange ideas. **Each group should submit ONE assignment**. Type or write your assignment on clean sheets of paper with question numbers prominently labelled. Answers that are difficult to read or locate may lose marks. We recommend working problems on a draft copy then writing a separate final copy to submit.

Your submission must be **STAPLED** and include the **CPSC 121 assignment cover page** – located at the Assignments section of the course web page. Additionally, include your names at the top of each page. We are not responsible for lost pages from unstapled submissions.

Submit your assignment to the appropriately marked box in room **ICCS X235** by the due date and time listed above. **Late submissions are not accepted**.

Note: the number of marks allocated to a question appears in square brackets after the question number.

<u>A Note on the Marking Scheme</u>

Most items (i.e., question or, for questions divided into parts, part of a question) will be worth 3 marks with the following general marking scheme:

- **3 marks:** correct, complete, legible solution.
- **2 marks**: legible solution contains some errors or is not quite complete but shows a clear grasp of how the concepts and techniques required apply to this problem.
- **1 mark**: legible solution contains errors or is not complete but shows a clear grasp of the concepts and techniques required, although not their application to this problem or the solution is somewhat difficult to read but otherwise correct.
- **0 marks**: the solution contains substantial errors, is far from complete, does not show a clear grasp of the concepts and techniques required, or is illegible.

More specifically, for this assignment we assign 9 marks for each induction proof: 3 marks for the basic cases, 3 marks for the induction step and 3 marks for proving the induction step.

This marking scheme reflects our intent for you to learn the key concepts and techniques underlying computation, determine where they apply, and apply them correctly to interesting problems. It also reflects a practical fact: we have insufficient time to decipher illegible answers. At the instructor's discretion, some items may be marked on a different scale. TAs may very occasionally award a bonus mark for exceptional answers.

Question 1 [9]

Find a formula for

$$S(n) = \frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \frac{1}{3\cdot 4} + \ldots + \frac{1}{n\cdot(n+1)}$$

Use mathematical induction to prove your result. *[Hint: Try a few terms first to get an idea of the formula.]*

Question 2 [9]

The sequence of *Fibonacci numbers* has very interesting properties and is used in many great problems in science and mathematics. The Fibonacci sequence is defined as following:

$F_1 = 1$

$F_2 = 1$

$F_n = F_{n-1} + F_{n-2}$ for any $n \geq 3$

That is, the first two numbers of the sequence are 1 and every other number is the sum of the previous two numbers in the sequence. So, the sequence looks like the following:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, . . .

Using induction, prove the following property of the Fibonacci numbers:

For any $n \geq 1$, $F_n < 2^n$

[For more fun with the Fibonacci numbers you can watch a video at: https://www.youtube.com/watch?v=ahXIMUkSXX0]

Question3 [9]

Consider the following Racket function which determines whether some number occurs in a list of numbers.

```
;; search : number list-of-numbers  ->  boolean
(define (search n alist)
  (cond
    [(empty? allist) false]
    [else (or (= (first alist) n) (search n (rest alist)))]))
```

Prove by mathematical induction that :

For every number n and any list of numbers L, (search n L) returns true if and only if n is in L.

[*Hint:* Use induction on the length of the input list]

FOR PRACTICE, NOT FOR MARKING

The following questions are for practice. Do not submit solutions to them.

Question P1 [0]

Write a regular expression describing the strings of characters from *{a, b, c}* which do not contain two consecutive letters that are both *a* or both *b*. That is, That is, strings which do not contain any substring of the form *aa* or *bb* (but may contain substrings *ab* and *ba*).

Question P2 [0]

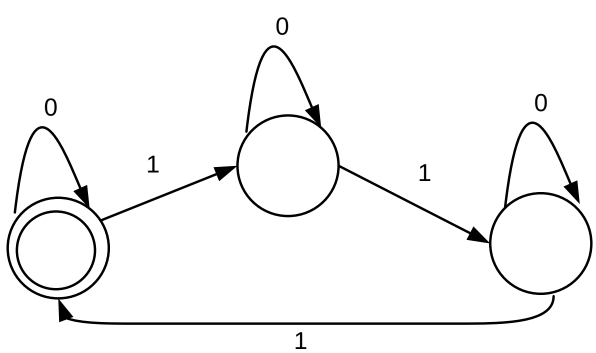
Consider again the Fibonacci numbers which were defined in question 2.

Using induction, prove the following property of the Fibonacci numbers:

For any n ≥ 1, $\sum_{i=1}^{n} F_i^2 = F_n \times F_{n+1}$

Question P3 [0]

Write a regular expression which describes exactly the set of strings of 0's and 1's which are accepted by the following DFA.



Question P4 [0]

Consider again the Racket function of question 5 which determines whether some number occurs in a list of numbers.

```
;; search : number list-of-numbers  ->  boolean

(define (search n alist)

  (cond

    [(empty? allist) false]

    [else (or (= (first alist) n) (search n (rest alist)))]))
```

Let `C(k)` = the number of comparison operations that `(search n L)` performs on a list `L` whose size is `k`. A comparison occurs when the program compares n with an element in the list. That is, when the operation `(= (first alist) n)` is executed. Prove by mathematical induction that in the worst case, `C(k)=k`, where `k` is the length of the list being searched. Note that the worst case for this algorithm is when n is not in the list.