Practice CS103 Midterm Exam II

This exam is closed-book and closed-computer. You may have a double-sided, 8.5” × 11” sheet of notes with you when you take this exam. You may not have any other notes with you during the exam.

You are welcome to cite lectures from the problem sets or lecture on this exam. Just tell us what you're citing and where you're citing it from. However, please do not cite results that are beyond the scope of what we've covered in CS103.

On the actual exam, there'd be space here for you to write your name and sign a statement saying you abide by the Honor Code. We're not collecting or grading this exam (though you're welcome to step outside and chat with us about it when you're done!) and this exam doesn't provide any extra credit, so we've opted to skip that boilerplate.

You have three hours to complete this practice midterm. There are 24 total points. This practice midterm is purely optional and will not directly impact your grade in CS103, but we hope that you find it to be a useful way to prepare for the exam. You may find it useful to read through all the questions to get a sense of what this practice midterm contains before you begin.

Question		Points	Grader
(1) The Pigeonhole Principle	(6)	/ 6	
(2) Induction	(6)	/ 6	
(3) Finite Automata and Regular Expressions	(6)	/ 6	
(4) Regular Languages	(6)	/ 6	
	(24)	**/ 24**	

Best of luck on the exam!

Problem One: The Pigeonhole Principle (6 Points)

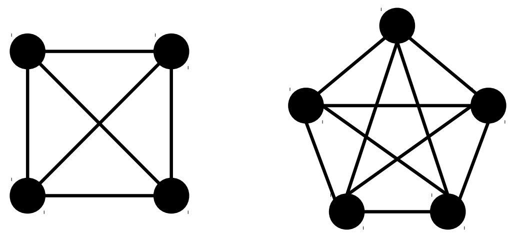
Suppose that there are six flavors of jelly beans and that you have eleven jelly beans of each flavor. You distribute those jelly beans across five jars. Prove that no matter how you distribute them, there will always be a jar with at least three jelly beans of one flavor and at least three jelly beans of a different flavor.

(Giving credit where credit is due: this excellent pigeonhole principle problem comes from a problem set given at MIT. I just thought it was such a good problem that I couldn't pass up on it. ☺)

Problem Two: Induction (6 Points)

In this problem, you'll see a new type of graph called the ***k-clique***, then will prove a useful property of k-cliques with applications to social network analysis.

A ***k-clique*** is a graph with k nodes where each node is connected to the k-1 other nodes in the graph. For example, here's a 4-clique and a 5-clique:



Now, suppose that you take a k-clique and color each edge either red or blue. Prove the following result by induction: if the k-clique contains an odd-length simple cycle made only of blue edges, then it must contain a simple cycle of length three with an odd number of blue edges (that is, a simple cycle of length three with exactly one blue edge or exactly three blue edges.) This result might seem pretty strange, but trust me, it's meaningful. We'll put details in the solution set. ☺

As a hint, try doing induction on the length of the cycle rather than the number of nodes in the graph.

(Extra space for your answer to Problem Two, if you need it.)

Problem Three: Finite Automata and Regular Languages (6 Points)

Let $\Sigma = \{\texttt{a}, \texttt{b}\}$ and let $L = \{ w \in \Sigma^* \mid w \text{ does not contain } \texttt{bbb} \text{ as a substring} \}$.

i. **(3 Points)** Design a DFA for *L*.

ii. **(3 Points)** Write a regular expression for *L*.

Problem Four: Regular Languages (6 Points)

The number of characters in a regular expression is defined to be the total number of symbols used to write out the regular expression. For example, `(a ∪ b)*` is a six-character regular expression, and `ab` is a two-character regular expression.

Let Σ = {a, b}. Find examples of all of the following:

- A regular language over Σ with a one-state NFA but no one-state DFA.
- A regular language over Σ with a one-state DFA but no one-character regular expression.
- A regular language over Σ with a one-character regular expression but no one-state NFA.

Prove that all of your examples have the required properties.