Practice 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 may not use any electronic devices during the course of this exam without prior authorization from the course staff. Please write all of your solutions on this physical copy of the exam.

You are welcome to cite results from the problem sets or lectures 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 However, please do not cite results that are beyond the scope of what we've covered in CS103.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) Graph Theory	(6)	/ 6	
(2) Induction	(6)	/ 6	
(3) Finite Automata and Regular Expressions	(6)	/ 6	
(4) Regular Languages	(6)	/ 6	
	(24)	**/ 24**	

You've got this. Best of luck on the exam!

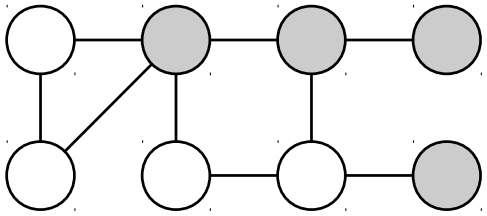
Problem One: Graph Theory (6 Points)

(Midterm Exam, Fall 2015)

If you'll recall from Problem Set Four, an ***independent set*** in a graph $G = (V, E)$ is a set $I \subseteq V$ such that no two nodes in I are adjacent to one another.

Let's begin with a new definition. A ***dominating set*** in G is a set $D \subseteq V$ with the following property: every node $v \in V$ either belongs to D or is adjacent to some node in D (or both).

For example, in the graph given below, the nodes in gray form a dominating set:



This question explores the interplay between independent sets and dominating sets.

i. **(3 Points)** Let $G = (V, E)$ be a graph with the following property: every node in G is adjacent to at least one other node in G. Prove that if I is an independent set in G, then $V - I$ is a dominating set in G. (Notice that we're asking you to show that $V - I$ is a dominating set, not that I is a dominating set.)

As a refresher, a ***dominating set*** in G is a set $D \subseteq V$ with the following property: every node $v \in V$ either belongs to D or is adjacent to some node in D (or both).

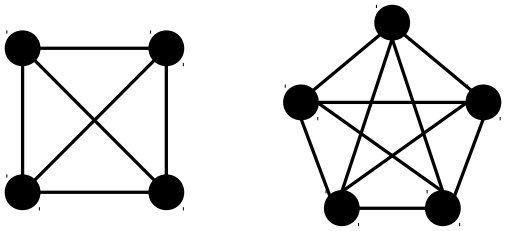
Now, let's introduce a new definition. We'll say that an independent set I in a graph G is a ***maximal independent set*** in G if there is no independent set I' in G where $I \subset I'$. (Here, $I \subset I'$ denotes that I is a strict subset of I'). Intuitively, a maximal independent set is one that can't be enlarged to form an even bigger independent set.

ii. **(3 Points)** Let $G = (V, E)$ be any undirected graph. Prove that if I is a maximal independent set in G, then I is also a dominating set in G.

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 Expressions (6 Points)

(Midterm Exam, Fall 2015)

Let $\Sigma = \{\texttt{a}, \texttt{b}\}$ and consider the following languages L_1 and L_2 over Σ:

$$L_1 = \{\ w \in \Sigma^* \ \mid\ w \text{ doesn't contain } \texttt{bb} \text{ as a substring } \}$$

$$L_2 = \{\ w \in \Sigma^* \ \mid\ |w| \geq 3 \text{ and the third-to-last character of } w \text{ is an } \texttt{a} \ \}$$

This problem concerns the language $L_1 \cap L_2$. As an example, the strings `aaa`, `baaba`, and `bababa` are all in $L_1 \cap L_2$, and the strings ε, `ba`, `abb`, `bbaab`, and `bab` are all not in $L_1 \cap L_2$.

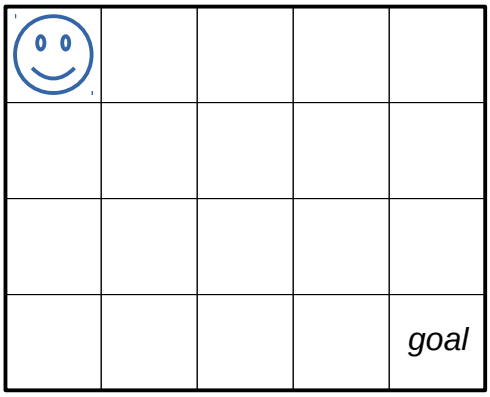
i. **(3 Points)** Design an NFA for $L_1 \cap L_2$. No justification is necessary.

ii. **(3 Points)** Write a regular expression for $L_1 \cap L_2$. No justification is necessary.

Problem Four: Regular Languages (6 Points)

(Midterm Exam, Fall 2015)

In Problem Set Six, two of the questions asked you to reason about someone walking their dog from the perspective of the regular languages. Our hope was to show you that it's possible to represent real-world processes as languages and to build DFAs that model the behavior of those processes. In this problem, we're going to ask you to think about how you might model a particular process – in this case, a robot moving around a grid world – as a DFA. Because the resulting DFAs will be quite large, we're not going to ask you to directly design DFAs for these languages, but will instead ask you questions about what those DFAs would look like.



Suppose there is a robot that moves around an 5 × 4 grid of squares. The robot can move around the grid by moving up exactly one square, moving down exactly one square, moving left exactly one square, and moving right exactly one square. The robot is not facing any particular direction, so the robot's motion is "absolute" rather than "relative." (If you've taken CS106A, this means that the robot is ***not*** like Karel the Robot.)

If the robot tries to move off of the grid (for example, moving right when up against the right side of the grid), the robot just says in its current position with no ill effects.

The robot begins in the upper-left corner of the world. Its goal is to arrive at the bottom-right corner of the world. A sample world like this is shown above.

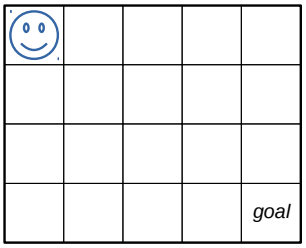
We can model a path that the robot takes as a string over the alphabet $\Sigma = \{$ `l`, `d`, `r`, `u` $\}$, where `l` means "move left one square," `r` means "move right one square," `u` means "move up one square," and `d` means "move down one square." For example, the string `ddrrul` means "go down two squares, then move right two squares, then move up one square, and then move left one square."

Consider the following language L_1:

$$L_1 = \{\ w \in \Sigma^* \mid w \text{ represents a series of steps that moves the robot so that it ends in the bottom-right corner of the world } \}$$

For example, the string `rrrrddd` is in L_1, as is `lldrdrrrrrd`.

i. **(2 Points)** The language L_1 is regular, meaning that it's possible to build a DFA for it. How many states are there in the smallest possible DFA for L_1? What pieces of information do those states correspond to? No proof is necessary – just tell us how many states there are and what those states mean.



Previously, we assumed that if the robot tried to move off of the grid, the robot would just stay in place and not move. Now, suppose we change this world so that if the robot tries to move off the grid, rather than staying in place, the robot instead crashes into the wall and breaks. (Life is hard when you're a robot.)

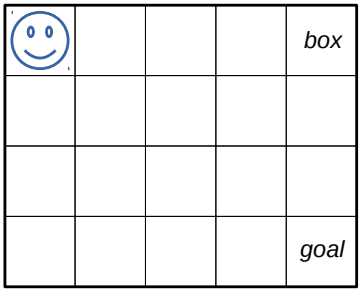
Consider this language L_2:

$$L_2 = \{\ w \in \Sigma^* \mid w \text{ represents a series of steps that moves the robot so that it ends in the bottom-right corner of the world without crashing and breaking in the process } \}$$

For example, the string `rrrrddd` is in L_2, as is `rddudrrrdlr`. However, the string `dddddrrrrr` is not in L_2, because at some point in that string the robot crashes into the bottom wall. Similarly, the string `dddrrrrr` is not in L_2, since although the robot ends in the bottom-right corner, it crashes into the right wall in the proccess.

ii. **(2 Points)** The language L_2 is regular, meaning that it's possible to build a DFA for it. How many states are there in the smallest possible DFA for L_2? What pieces of information do those states correspond to? No proof is necessary – just tell us how many states there are and what those states mean.

Now, let's suppose that we want to give the robot a task. The robot needs to move from the upper-left corner to the lower-right corner, but along the way needs to pick up a box located in the upper-right corner of the world. We'll assume that if the robot enters that square, it automatically picks up the box. As before, if the robot tries to move into a wall, the robot crashes and breaks.



Consider the following language L_3:

$$L_3 = \{\ w \in \Sigma^* \mid w \text{ represents a series of steps that moves the robot so that it ends in the bottom-right corner of the world, without crashing and breaking in the process, and picking up the box along the way } \}$$

For example, `dddrrrr` is not in L_3 because the robot doesn't pick up the box before ending in the bottom corner. Similarly, `rrrr` is not in L_3 because the robot picks up the box but doesn't end in the bottom corner. The string `rrrrdddd` is also not in L_3, since while the robot does pick up the box and does end up in the bottom-right corner, the robot crashes in the process. However, `rrrrddd` is in L_3, as are the strings `drrrruddd`, `rrrrllllldddrrrr`, and `dddrrrruuuddd`.

iii. **(2 Points)** The language L_3 is regular, meaning that it's possible to build a DFA for it. How many states are there in the smallest possible DFA for L_3? What pieces of information do those states correspond to? No proof is necessary – just tell us how many states there are and what those states mean.