Practice Midterm

- Don’t panic.
- The practice midterm has 9 problems.
- The real midterm has 3 problems.
Problem 1

Let $\Sigma = \{0, 1\}$. Let $\text{reverse} : \Sigma^* \rightarrow \Sigma^*$ be the reverse function. Prove that for every $k$, there exists a language $L$ where the minimal DFA for $L$ has $k + 2$ states, but the minimal DFA for $L^R = \{\text{reverse}(x) \mid x \in L\}$ has $2^k$ states.

Given $k$, define $L_k$

Prove $L_k$ has exactly $k + 2$ equivalence classes.

Prove $L_k^R$ has exactly $2^k$ equivalence classes.
Problem 2

Let $\Sigma = \{0, 1\}$. Let $\circ$ be the concat operator.
Prove there exists some regular language $L$ such that $L' = \{ w \circ w \mid w \in L \}$ is irregular.

Define $L$

Argue that $L$ is regular

Show that $L'$ has an infinite number of equivalence classes
Problem 3

Let $\Sigma = \{0, 1\}$. Let $\circ$ be the concat operator.
Prove that for every regular $L$, $L' = \{ w \circ w \in L \}$ is regular.

Let $L$ be any regular language. Show that $L'$ is also regular.

*Hint, take the DFA for $L$, construct a DFA for $L'$*
Problem 4
Let $\Sigma = \{0, 1\}$. For a NFA, an accepting path is any path from the starting state to any accepting state, consistent with the input. A 2NFA is an $\epsilon$-free NFA that accepts if and only if the total number of accepting paths is exactly 2.
Prove that 2NFAs only recognize regular languages.

Let $L$ be any language recognized by a 2NFA. Prove $L$ is regular.

*Hint: construct a DFA*
Problem 5

Let $\Sigma = \{0, 1\}$. For a NFA, an accepting path is any path from the starting state to any accepting state, consistent with the input. An odd-NFA is an $\epsilon$-free NFA that accepts if and only if the total number of accepting paths is odd. Prove that odd-NFAs only recognize regular languages.

Let $L$ be any language recognized by an odd-NFA. Prove $L$ is regular.

*Hint: construct a DFA*
Problem 6

Let $\Sigma = \{0, 1\}$. The Turing Machines we have seen in class are 1D-TM since they operate on a 1D-tape. A 2D-TM operates on a 2D-grid and has transition function $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{LEFT, RIGHT, UP, DOWN\}$. (The input is written along the x-axis). Prove that a 2DTM is equivalent in power to a 1DTM.

Design an algorithm executable on a 1D-TM for simulating 2D-TMs.
Problem 7

Let $\Sigma = \{0, 1\}$. For $s, r \in \Sigma^*$, we say that $s \prec r$ if (1) $|s| < |r|$ or (2) $|s| = |r|$ and $s$ appears ahead of $r$ in alphabetical order.

A sequence of strings $s_1, s_2, \ldots$ is in lexicographical order if for all $i < j$, we have $s_i \prec s_j$.

A language is lexicographically enumerable if there is an enumerator that outputs the language in lexicographical order.

Prove that a language is lexicographically enumerable if and only if it is decidable.

Assume $L$ is lexicographically enumerable. Prove $L$ is decidable.

Assume $L$ decidable. Prove $L$ is lexicographically enumerable.
Problem 8

For every integer $n$, consider the streaming complexity of the problem of deciding whether a graph
on $n$ vertices, given by a stream of edges, is bipartite.

That is, for a set of $n$ vertices $V$, our alphabet $\Sigma = \{\{x, y\} \mid x, y \in V, x \neq y\}$ is all possible
(undirected) edges between these vertices and our stream is a sequence of these edges. If we call
the set of each edge in this stream $E$, then $G = (V, E)$ is the undirected graph defined by it. We
want a streaming algorithm that takes the stream and computes whether or not $G$ is bipartite.
Show that every streaming algorithm for this problem requires $\Omega(n)$ bits of memory.

Define $Q$, the set of distinguishable strings.

Argue that for any $s, r \in Q$, they are distinguishable.

Argue that $|Q| = 2^{\Omega(n)}$
Problem 9

For every integer \(n\), consider the streaming complexity of the problem of deciding whether a graph on \(n\) vertices, given by a stream of edges, is connected.

That is, for a set of \(n\) vertices \(V\), our alphabet \(\Sigma = \{\{x, y\} \mid x, y \in V, x \neq y\}\) is all possible (undirected) edges between these vertices and our stream is a sequence of these edges. If we call the set of each edge in this stream \(E\), then \(G = (V, E)\) is the undirected graph defined by it. We want a streaming algorithm that takes the stream and computes whether or not \(G\) is connected. Show that every streaming algorithm for this problem requires \(\Omega(n \log n)\) bits of memory.

Define \(Q\), the set of distinguishable strings.

*Hint: you are not expected to solve this problem. We have put it here in case the rest of the practice midterm was not entertaining.*

Argue that for any \(s, r \in Q\), they are distinguishable.

Argue that \(|Q| = 2^{\Omega(n \log n)}\)