Exam Time: 1h & 15m. Each problem is worth 10 points. 50 pts total. Choose any 5 problems.

1. Regular Expressions and Nondeterministic Finite Automata.
   a) Give a regular expression for the set of all strings over \( \Sigma = \{0, 1\} \) having a number of zeros that is a multiple of 3.

   **Solution:** \( 1^* (01*01*01^*)^* \). You would expect the empty string and strings of just 1s to meet the condition.

   b) Construct an NFA that recognizes the language defined by this regular expression. Use the labeled digraph method.

   **Solution:** This asked for a specific method, and that the method be applied to the R.E. of part 1 - since you need to have an R.E. to apply this method to. We will take this through a few of the steps. This also reflects the R.E. chosen - others are possible. The character \& will stand for \( \epsilon \).

   ![Diagram 1](image1.png)

   **Constructing an NFA - labeled digraph method. Step 1**

   ![Diagram 2](image2.png)

   **Constructing an NFA - labeled digraph method. Step 2**
Finishing should now be trivial.

2. **Closure of Regular Languages.** Let $A$ be a regular language over some alphabet $\Sigma$. Prove that the language $B = \{x \mid xx^R \in A\}$ is regular.

**Solution:** since $A$ is regular, there exists a DFA $M$ accepting $A$. Let $M'$ be the NFA accepting $A^R$ constructed as in Example 2.34, and consider the NFA $M \times M'$. The only problem is now the determination of the set of accepting states: let $F = \{(q_i, q_i) \mid q_i \in M\}$. Explain why we are done.

3. **NFA to DFA.** Given the NFA:

convert it into an equivalent DFA, showing all intermediate steps.

**Solution:** we begin by setting up a table with the $\epsilon$-closure of the set $\{s\}$, and the input characters we expect. We then compute the new state sets and their $\epsilon$-closures.

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_e = {s, q_1}$</td>
<td>$Q_0 = {s, q_1, q_2}$</td>
<td>$Q_1 = {s, q_1, q_2}$</td>
</tr>
<tr>
<td>$Q_0 = {s, q_1, q_2}$</td>
<td>$Q_00 = {s, q_1, q_2} = Q_0$</td>
<td>$Q_{01} = {s, q_1, q_2}$</td>
</tr>
<tr>
<td>$Q_1 = {s, q_1, q_2}$</td>
<td>$Q_{10} = {s, q_1, q_2} = Q_0$</td>
<td>$Q_{11} = {s, q_1, q_2}$</td>
</tr>
</tbody>
</table>

Since $F = \{q_1\}$ for the original NFA, the set of final states for the DFA is $\{Q_e, Q_0\}$. We get the DFA:
This can be further reduced, but that is not part of the problem.

   a) Define the relation $R_L$ used by the theorem.
   b) Find all equivalence classes of $R_L$ for the language \( \{ x \in \{0,1\}^* \mid \#_0(x) \geq 3 \} \), where \( \#_a(w) \) is the number of occurrences of the symbol \( a \) in \( w \).
   c) What does the Myhill-Nerode theorem tell you about this language and why? Start by quoting the theorem...

**Solution:** part a) on p.70, line 7 and following.

Part b): we must find the equivalence classes.

\( [\epsilon]_{R_L} \) - this is the class of the empty string. Concatenating the empty string with any string with three or more zeros gives an element of \( L \), while concatenating it with any string with 2 or fewer zeros gives a string not in \( L \), we conclude that every string of the form \( 1^* \) is in \( [\epsilon]_{R_L} \) and that only strings of that form belong to \( [\epsilon]_{R_L} \).

\( [0]_{R_L} \) - a string belongs to this class if and only if it has a single 0. Use the definition of \( R_L \).

\( [00]_{R_L} \) - a string belongs to this class if and only if it has 2 0s.

\( [000]_{R_L} \) - a string belongs to this class if and only if it has 3 or more 0s. Since all string in \( \Sigma^* \) fall into one of the categories, we have computed the full equivalence relation.

Part c) The Myhill-Nerode (as given on the slides) states: For any regular language \( L \), its minimum DFA has exactly \( \text{Index}(R_L) \) states. Since the number of states for the minimum DFA recognizing \( L \) is 4 (finite - required for a language to be regular), we conclude that \( L \) is a regular language.

5. Pumping Lemma. Show that the language \( \{ w \in \{0,1,2\}^* \mid \#_0(w) + \#_1(w) = 2\#_2(w) \} \) is not regular.

**Solution:** assume the language (name it \( L \)) to be regular. This implies that there exists an integer \( N > 0 \) (the number of states of the minimal recognizing DFA) such that any string \( \alpha \in L \) with \( |\alpha| \geq N \) and any way of breaking it into \( \alpha = xyz \) with \( |y| \geq N \), \( y \) can be written as \( y = uvw \) such that \( v \neq \epsilon \) and \( xuv^*wz \subseteq L \).

Consider the string \( \alpha = 0^N1^N2^N \in L \). Split it as \( x = \epsilon, y = 0^N, z = 1^N2^N \). Since \( |y| \geq N \), \( y \) can be decomposed as \( y = uvw \) with \( v \neq \epsilon \). Since \( v \) consists only of 0s, the Strong Form of
the Pumping Lemma would imply that \(0^N - |v|1^N2^N \in L\). Which is not possible, since this string, because of \(N - |v| < N\), does not satisfy the defining condition for \(L\).

6. **Minimum DFAs.** Construct the minimum DFA for the language \((0 + 1)^*01\). Use any of the methods discussed in the textbook, but be explicit.

**Solution:** see Example 2.53 for three distinct methods. The important part is explicitness: I have to know HOW you applied the method.