Problem 1. (10 points)

Prove that the following are not regular languages.

(a) \( \{0^n1^m 2^n | n \text{ and } m \text{ are arbitrary integers} \} \).

(b) \( \{0^{2^n} 1^n | n \geq 1 \} \)

(d) Proof. Assuming the language \( L \) is regular, let \( p \) be the pumping-lemma constant. Pick \( w = 0^p1^p2^p \). Then when we write \( w = xyz \), we know that \( |xy| \leq p \), and therefore \( y \) consists of only 0’s. Thus, \( xz \), which must be in \( L \) if \( L \) is regular, consists of fewer than \( p \) 0’s, followed by a 1 and exactly \( p \) 2’s. That string is not in \( L \), so we contradict the assumption that \( L \) is regular. □

(f) Proof. Assuming the language \( L \) is regular, let \( p \) be the pumping-lemma constant. Pick \( w = 0^{2p}1^p \). Then when we write \( w = xyz \), we know that \( |xy| \leq p \), and therefore \( y \) consists of only 0’s. Thus, \( xyyz \), which must be in \( L \) if \( L \) is regular, consists of more than \( 2p \) 0’s, followed by exactly \( p \) 1’s. That string is not in \( L \), so we contradict the assumption that \( L \) is regular. □
Problem 2. (10 points)

Prove that the following are not regular languages:

The set of strings of 0’s and 1’s that are of the form $ww$, that is, same string repeated.

Proof. Assuming the language $L$ is regular, let $p$ be the pumping-lemma constant. Pick a string $0^p1^p0^p1^p$. Then when we write it as $xyz$, we know that $|xy| \leq p$, and therefore $y$ consists of only 0’s. Thus, $xz$, which must be in $L$ if $L$ is regular, consists of fewer than $p$ 0’s, followed by exactly $p$ 1’s, then exactly $p$ 0’s, and another $p$ 1’s. Clearly this string is not of the form $ww$, so we contradict the assumption that $L$ is regular. □
Problem 3. (Exercise 4.2.3, 10 points)

If $L$ is a language, and $a$ is a symbol, then $a \setminus L$ is the set of string $w$ such that $aw$ is in $L$. For example, if $L = \{a, aab, baa\}$, then $a \setminus L = \{\epsilon, ab\}$. Prove that if $L$ is regular, so is $a \setminus L$. 

*Hint:* Remember that the regular languages are closed under reversal and under the quotient operation of Exercise 4.2.2.

**Proof.** If $L$ is regular, so is $L^R$ (the regular languages are closed under reversal). According to Exercise 4.2.2, we know $L^R/a$ is also regular. Since it is easy to prove $a \setminus L = (L^R/a)^R$, we conclude that $a \setminus L$ is regular.

□
Problem 4. (10 points)

Give an algorithm to tell whether a regular language \( L \) contains at least 100 strings.

Algorithm 1 \textsc{NumberOfStrings}(D, n)

\textbf{Input:} \( D \): a black box that tests if a string is in \( L \). \( n \): pumping lemma constant.

\textbf{Output:} Return “yes” if \( L \) contains at least 100 strings, otherwise return “no”

\begin{enumerate}
\item for \( i \leftarrow n \) to \( 2n - 1 \) do \\
\item \hspace{1em} for all string \( w \) of length \( i \) do \\
\item \hspace{2em} if \( D(w) = \) accept then return “yes” // the language is infinite // \\
\item \hspace{1em} \text{count} \leftarrow 0 \\
\item \hspace{1em} for \( i \leftarrow 0 \) to \( n - 1 \) do \\
\item \hspace{2em} for all string \( w \) of length \( i \) do \\
\item \hspace{3em} if \( D(w) = \) accept then \\
\item \hspace{4em} \text{count} \leftarrow \text{count} + 1 \\
\item \hspace{1em} if \( \text{count} \geq 100 \) then \\
\item \hspace{2em} return “yes” \\
\item \hspace{1em} else \\
\item \hspace{2em} return “no”
\end{enumerate}

Suppose, however, that there are no strings in \( L \) whose lengths are in the range \( n \) to \( 2n - 1 \). We claim there are no strings in \( L \) of lengths \( 2n \) or more, and thus testing all strings of lengths between 0 and \( n - 1 \) is sufficient for us to tell whether \( L \) contains at least 100 strings. In the proof, suppose \( w \) is the shortest string in \( L \) of length at least \( 2n \). Then the pumping lemma applies to \( w \), and we can write \( w = xyz \), where \( xz \) is also in \( L \). How long could \( xz \) be? It can’t be as long as \( 2n \), because it is shorter than \( w \), and \( w \) is the shortest string in \( L \) of length \( 2n \) or more. It can’t be shorter than \( n \), because \( |y| \leq n \). Thus, \( xz \) is of length between \( n \) and \( 2n - 1 \), which is a contradiction, since we assumed there were no strings in \( L \) with a length in that range.

Clearly, the blackbox \( D \) and constant \( n \) can be easily determined if the input regular language is represented as a DFA (or NFA or regular expression). That is, \( D \) is basically the membership algorithm and \( n \) could be fixed as the size of the DFA.
Problem 5. (Exercise 4.4.2, 20 points)

The following figure is the transition table of a DFA.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \rightarrow A )</td>
<td>B</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>F</td>
</tr>
<tr>
<td>( \ast C )</td>
<td>D</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>I</td>
</tr>
<tr>
<td>( \ast F )</td>
<td>G</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>I</td>
<td>C</td>
</tr>
<tr>
<td>( \ast I )</td>
<td>A</td>
<td>E</td>
</tr>
</tbody>
</table>

(a) Draw the table of distinguishabilities for this automaton.

(b) Construct the minimum-state equivalent DFA.

(a) Equivalent classes: \( \{A, D, G\} \), \( \{B, E, H\} \), \( \{C, F, I\} \).