1. Regular Expressions.

a) Give an algorithm to find a shortest string in a regular set with a given regular expression.

b) Apply the algorithm you described in part a) to the regular expression $00 + 11 + (01 + 10)(00 + 11)^* (01 + 10)$. Show how your algorithm applies.

Solution for a). Regular expressions are built recursively, so we should expect that any such algorithm will be recursive. Let $R_i$, for $i \geq 1$ denote a Regular Expression and make use of associativity of union and concatenation, along with precedence of Kleene closure over concatenation and of concatenation over union. We assume that $A^+$ will be written as $AA^*$. Shorter_of will return the shorter of two strings, with a random pick if they are of the same length.

Min-String (R)

if $R = Empty_Set$ return Error
if $R = Empty_String$ return Empty_String
if $R = Singleton_String$ return Singleton_String
if $R = R_1 + R_2$ return Shorter_of (Min_String (R_1), Min_String (R_2))
if $R = R_1 \cdot R_2$ return Concat (Min_String (R_1), Min_String (R_2))
if $R = Kleene_Closure (R_1)$ return Empty_String
Solution for b). What we will get is

\[
\text{Shorter}_\text{o}(\text{Min}_\text{String}(00), \text{Min}_\text{String}(11 + (01 + 10)(00 + 11)^*(01 + 10))) = \\
\text{Shorter}_\text{o}(00, \text{Shorter}_\text{o}(11, \text{Min}_\text{String}((01 + 10)(00 + 11)^*(01 + 10)))) 
\]

where

\[
\text{Min}_\text{String}((01 + 10)(00 + 11)^*(01 + 10)) = \\
\text{Concat}(\text{Min}_\text{String}(01 + 10), \text{Min}_\text{String}((00 + 11)^*(01 + 10))) = \ldots = 0101
\]

and

\[
\text{Shorter}_\text{o}(00, \text{Shorter}_\text{o}(11, 0101)) = \text{Shorter}_\text{o}(00, 11) = 00.
\]

Where choices among equal length strings are random.

2. **Closure of Regular Languages.** Let \( A \) be a regular language over some alphabet \( \Sigma \). Prove that the language \( A_x = \{ y \mid xy \in A \} \), where \( x \) is a fixed string, is regular.

Solution. Let \( M = \{ Q, \Sigma, \delta, q_0, F \} \) be the DFA for \( L \). Let, for any string \( x \in \Sigma^* \), \( M_x = \{ Q, \Sigma, \delta, \delta(q_0, x), F \} \), where \( \delta(q_0, x) \) denotes the state of \( M \) reached at the end of \( x \) starting from \( s \). Then \( M_x \) is the DFA for \( A_x \), and thus \( A_x \) is regular.

3. **NFA to DFA.** Given the NFA \( M = (\{ p, q, r \}, \{ 0, 1 \}, \delta, p, \{ q, r \}) \), where:

<table>
<thead>
<tr>
<th>( \delta )</th>
<th>0</th>
<th>1</th>
<th>( \epsilon )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( { p, q } )</td>
<td>( { p } )</td>
<td>-</td>
</tr>
<tr>
<td>( q )</td>
<td>-</td>
<td>( { r } )</td>
<td>-</td>
</tr>
<tr>
<td>( r )</td>
<td>-</td>
<td>-</td>
<td>( { p } )</td>
</tr>
</tbody>
</table>

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

Solution. We construct the transition table:

<table>
<thead>
<tr>
<th>( \delta_D )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q_0 = { p } )</td>
<td>( { p, q } )</td>
<td>( { p } )</td>
</tr>
<tr>
<td>( Q_1 = { p, q } )</td>
<td>( { p, q } )</td>
<td>( { p, r } )</td>
</tr>
<tr>
<td>( Q_2 = { p, r } )</td>
<td>( { p, q } )</td>
<td>( { p } )</td>
</tr>
</tbody>
</table>

from which we construct the graph:
   a) Define the relation $R_L$ used by the theorem.

   b) Find all equivalence classes of $R_L$ for the language \( \{ x \in \{0,1\}^* \mid \#_0(x) = \#_1(x) \} \), where $\#_a(w)$ is the number of occurrences of the symbol $a$ in $w$. Make sure you explain the construction.

   c) What does the Myhill-Nerode theorem tell you about this language and why? Start by quoting the Theorem and the appropriate Corollary...

Solution

   a) - Textbook, p.70, l. 7 and following.

   b) Consider the following strings and their classes:

   \[
   [\varepsilon]_{R_L} = \{ y \mid \forall w \in \{0,1\}^*, \varepsilon w \in L \iff y w \in L \}. \]

   Since $w \in L \iff \#_0(w) = \#_1(w)$, $y$ must have exactly the same number of 0s as 1s, and thus be in $L$. So $[\varepsilon]_{R_L} = L$.

   Similar observations lead us to the conclusions that $[0^n]_{R_L} = \{ x \mid x \in \{0,1\}^*, \#_0(w) = \#_1(w) + n \}, \forall n > 0$ and $[1^n]_{R_L} = \{ x \mid x \in \{0,1\}^*, \#_0(w) = \#_1(w) - n \}, \forall n > 0$. It should be trivial to see that all these classes are distinct. We thus have infinitely many equivalence classes.

   c) The Myhill-Nerode Theorem and its Corollary state that a Language $L$ is regular iff $\text{Index}(R_L) < \infty$, where $\text{Index}(R_L)$ is both the number of equivalence classes of $R_L$ and the number of states of a minimal DFA for $L$ (if $L$ is regular). Since $\text{Index}(R_L) = \infty$, the language above is not regular.

5. Pumping Lemma &... Show that the language \( \{ w \in \{0,1,2\}^* \mid \#_0(w) + \#_1(w) \neq 3\#_2(w) \} \) is not regular.

Solution. Although it may be possible to prove the result directly, that direction requires some care and extra arguing - requiring the setting up of a whole family of strings at least one of which will give us a way to violate the condition. The simplest way is to first use the fact that the complement of any regular language is regular, and attempt to prove non-regularity.
for $\bar{L} = \{ w \in \{0, 1, 2\}^* \mid \#_0(w) + \#_1(w) = 3\#_2(w) \}$: we have just changed an inequality into an equality, which should be easier to deal with.

Assume $L$ regular. There exists a positive integer $s$ such that any string $\alpha \in \{0, 1, 2\}^*$, $|\alpha| \geq s$ can be decomposed according to one of the Pumping Lemmas. Since we will need some control over where the cycle will appear, let $\alpha = 0^{2s}1s2^s$, which is clearly in $\bar{L}$. The strong form of the Pumping Lemma allows us to break $\alpha = xyz$, with $x = 0^{2s}$, $y = 1^s$, $z = 2^s$, and to conclude that $y$ can be further decomposed as $y = uvw$, with $1 \leq |v| \leq s$ and satisfying $xuv^*wz \subseteq \bar{L}$. From this we have $0^{2s}1^sz2^s \in \bar{L}$, which is a contradiction, since $2s + (s - |v|) = 3s - |v| \neq 3s$.

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

**Solution.** Using the ”checker method” and a little care, we construct the DFA:

```
A DFA For the Language.
```

To determine the minimal DFA, the easiest method is the third: split the states into Final $\{q_2\}$ and Non-Final $\{s, q_1\}$. Since $q_2 \rightarrow_0 1 q_2$ there is no further splitting of F. We observe that $s \rightarrow 1 s$ and $s \rightarrow 2 q_1$, remaining in the Non-final states, while $q_1 \rightarrow^0 q_1$ and $q_1 \rightarrow 1 q_2$ splitting the Non-Final states into $\{s\}$ and $\{q_1\}$. Since the three sets $\{s\}, \{q_1\}, \{q_2\}$ are all singletons, no further splitting is possible and the minimal DFA must have 3 states. Since the DFA we constructed has 3 states, it must be minimal.