1. We define an operation BACKANDFORWARD such that:

\[
    \text{BACKANDFORWARD}(A) = \{ w \in A | w \in A \text{ and } w^R \in A \}.
\]

That is, given any language \( A \), \( \text{BACKANDFORWARD}(A) \) is a new language containing all elements in \( A \) whose reverse is also in \( A \). Prove that the class of regular languages is closed under the operation \( \text{BACKANDFORWARD} \).

**Solution Sketch.** We have shown in homework 2 that if \( A \) is a regular language, then \( A^R = \{ w | w^R \in A \} \) is also regular. Note that \( \text{BACKANDFORWARD}(A) = A \cap A^R \). Since the class of regular languages is closed under intersection, for any regular language \( A \), we have \( \text{BACKANDFORWARD}(A) \) is also regular.

2. (a) Prove that the class of regular languages is closed under set difference. That is, if \( A \) is regular and \( B \) is regular, show that \( A \setminus B = \{ w \in A | w \notin B \} \) is regular.

**Solution Sketch.** Note that \( A \setminus B = A \cap \overline{B} \). Thus it is regular because \( A \) and \( B \) are both regular, and the class of regular languages are closed under intersection and complementation.

(b) show that the class of regular languages is closed under symmetric set difference, defined as \( A \Delta B = \{ w | w \in A \text{ or } w \in B \text{ but not both} \} \).

**Solution Sketch.** Note that \( A \Delta B = (A \setminus B) \cup (B \setminus A) \). Thus it is regular because \( A \) and \( B \) are both regular, and the class of regular languages is closed under union and set difference.

3. Prove that the class of regular languages is closed under PREFIX where \( \text{PREFIX}(A) = \{ w | \exists x \in \Sigma^* \text{ such that } wx \in A \} \).

**Solution Sketch.** Assume a DFA \( M \) recognizes \( A \). We modify \( M \) to be the following DFA \( M' \). \( M \) and \( M' \) have the same set of states and transition functions, the only difference is the set of accept states. Specifically, for every state in \( M \) that can reach an accept state by reading some string in \( \Sigma^* \), we make it an accept state in \( M' \) (this includes the original states in \( M \), which can reach an accept state by just reading \( \epsilon \)). We claim that \( M' \) recognizes \( \text{PREFIX}(A) \).

To see this, note that for every \( w \in \text{PREFIX}(A) \), there is an \( x \) s.t. \( wx \in A \). Note that if \( wx \in A \), then by reading \( wx \), \( M \) will go through a path from the start state to an accept state. Since every state on this path is an accept state in \( M' \), \( w \) will be accepted by \( M' \).

Now for any string \( w \), assume \( w \) is accepted by \( M' \). When reading \( w \), \( M' \) starts at the start state and ends at an accept state in \( M' \). By our construction, there is a string \( x \) that makes \( M \) go from that state to an accept state in \( M \). Therefore \( wx \in A \) and thus \( w \in \text{PREFIX}(A) \).
4. Convert the following regular expression into an NFA:

\((0 \cup 11)^*01(01 \cup 1)^*\)

Solution Sketch. We can directly use the transformation from a regular expression to an NFA. First construct NFAs recognizing \(0\), \(11\), \(01\), \(1\) and then use the constructions for concatenation, star and union to get the final NFA.

5. Let \(\Sigma = \{0, 1\}\). Give an equivalent regular expression for each of the languages below.

(a) \(L_1 = \{w | w\text{ begins with a }1\text{ and ends with a }0\}\)
(b) \(L_2 = \{w | w\text{ contains at least three }1s\}\)
(c) \(L_3 = \{w | w\text{ contains the substring }0101\}\)
(d) \(L_4 = \{w | w\text{ has length at least three and its third symbol is }0\}\)
(e) \(L_5 = \{w | w\text{ starts with a }0\text{ and has odd length, or starts with a }1\text{ and has even length}\}\)
(f) \(L_6 = \{w | w\text{ does not contain the substring }011\}\)

Solution Sketch. (a) \((1 \cup 0)^*0\)
(b) \((1 \cup 0)^*1(1 \cup 0)^*1(1 \cup 0)^*\)
(c) \((1 \cup 0)^*0101(1 \cup 0)^*\)
(d) \((1 \cup 0)(1 \cup 0)(1 \cup 0)^*\)
(e) \(0((1 \cup 0)(1 \cup 0))^*) \cup (1(1 \cup 0)((1 \cup 0)(1 \cup 0))^*)\)
(f) First construct a DFA that recognize \(\overline{L_6}\), i.e., the complementation of \(L_6\). Then construct another DFA that recognizes \(L_6\) by changing the non-accept states into accept states, and vice versa. Finally use the transformation from a DFA to a regular expression (via GNFA) to get the regular expression.

\(\square\)