EXERCISE 1

Decide which of the following statements is true and which is false. Give a brief explanation of your answer.

a) For all languages $L_1$ and $L_2$, it holds that $(L_1^* L_2^*)^* = (L_1^* L_2^*)^*$.

b) If $L_1$ and $L_2$ are both not regular then $L_1 L_2$ could be regular.

c) For all languages $L_1$ and $L_2$, if $L_1 \leq L_2$ then $L_1^* \leq L_2^*$.

EXERCISE 2

Show that the following languages are not regular.

a) $\{0^n 1^m 0^{n+m} | m, n > 0 \}$

b) $\{ w \in \{0,1\}^* | w \text{ is a palindrome} \}$

EXERCISE 3

Give algorithms to tell whether:

a) a regular language $L$ is universal (i.e., $L = \Sigma^*$);

b) two regular languages have at least one string in common.

EXERCISE 4

Show that if $L$ and $M$ are regular languages then so is $L \cap M$ (without using the De Morgan law $L \cap M = \overline{\overline{L} \cup \overline{M}}$). Apply the construction to the following automata:

$A_L^* = \begin{array}{c}
\begin{array}{ccc}
q_0 & \xrightarrow{0} & q_1 \\
& & \xrightarrow{1} q_2
\end{array}
\end{array}$

$A_M^* = \begin{array}{c}
\begin{array}{ccc}
q_0 & \xrightarrow{1} & q_1 \\
& & \xrightarrow{1} q_2
\end{array}
\end{array}$
1) a) False. Consider the languages \( L_1 = \{a\} \) and \( L_2 = \{b\} \). Then \( b \notin (L_1^* L_2^*)^* \) but \( b \notin (L_1^* L_2^*)^* \).

1) b) True. Assume that \( L_1 = \overline{L}_2 \), i.e. \( L_2 = \overline{L}_1 \). If \( L_1 \) is not regular then \( \overline{L}_1 = \overline{L}_2 \) is also not regular (because, if \( L_2 \) were regular then, by the closure properties of regular languages, \( L_1 \) would be regular too, thus leading to a contradiction). Since \( L_2 \cup L_2 = \{\}^* \), we have that the union of two non-regular languages can be regular.

1) c) True. Given that, for all \( w \in L_1 \), we also have that \( \overline{w} \in L_2 \), the argument goes as follows. If \( w' \in L_1^* \), then \( w' = w_1 \ldots w_n \) for some \( n \in \mathbb{N} \) and \( w_i \in L_1 \) (\( 1 \leq i \leq n \)). But then each \( w_i \) is also in \( L_1 \) and therefore \( w' \in L_1^* \).

2) a) Assume that the language is regular. Then, by the pumping lemma, we would have that:

- There exists \( n \) such that
- For all \( w \in L \) such that \( |w| > n \)
  - There are three strings \( x, y, z \) such that \( w = x y z \) and \( |y| \leq n \), \( |y| > 1 \), and for all \( k \geq 0 \), \( x y^k z \in L \).

Now, given some \( n \), let \( w = 0^n 1^n 2^n 0^n \). Since \( |w| = 4n \) we have that \( |w| > n \). In order to apply the pumping lemma we need to find strings \( x \) and \( y \) such that \( |xy| \leq n \). The only possible choices are \( x = 0^n \) and \( y = 0^b \) where \( b \geq 1 \). But then we have that \( x z = 0^n 1^n 2^n 0^{n+b} \) and thus \( n+b+n \neq 2n \). Therefore, for \( b = 0 \), \( x y^2 z \notin L \). Since we assumed that the language is regular this is a contradiction. Hence the language cannot be regular.
2) b) Again, we use the pumping lemma. Given some \( n \), let \( w = 0^n 1^n \).
If we consider \( x, y, z \) such that
\[
a) \ w = xy^2z, \quad b) \ |xy| \leq n, \quad c) \ |y| \geq 1
\]
then \( y \) can only be a non-empty string of 0's. Thus, for each \( k \geq 1 \), the string \( xy^kz \) has more 0's on the left-hand side of 1 than on the right-hand side. We can conclude that, for \( k \geq 1 \), \( xy^kz \notin L \). Therefore we have that the language is not regular.

3) a) Note that if \( L \) is universal then \( \Sigma^* - L = \emptyset \). Therefore we only need to check whether \( L \) is empty.

3) b) We can check whether the intersection \( L \) of the two languages that we denote with \( L_1 \) and \( L_2 \) is non-empty, i.e., we can check whether \( L = L_1 \cap L_2 \) is non-empty. Note that \( L \) is regular because of the closure properties of regular languages.

4) Let \( L \) and \( M \) be the regular languages accepted by the automata \( A_L = (Q_L, \Sigma_L, \delta_L, q_L, F_L) \) and \( A_M = (Q_M, \Sigma_M, \delta_M, q_M, F_M) \). We assume:
\[
a) \ \Sigma_L = \Sigma_M = \Sigma, \quad b) \ A_L \text{ and } A_M \text{ are deterministic.}
\]
We construct an automaton \( A \) that simulates \( A_L \) and \( A_M \). The states of \( A \) are pairs of states \((p, q)\) where \( p \in Q_L \) and \( q \in Q_M \). If \( a \) is an input symbol and \( A \) is in state \((p, q)\) then \( A \) goes in state \((p', q')\) where \( p' = \delta_L(p, a) \) and \( q' = \delta_M(q, a) \). The start state of \( A \) is \((q_L, q_M)\) and the accepting states of \( A \) are those pairs \((p, q)\) where both \( p \in F_L \) and \( q \in F_M \).
To sum up, we have that

\[ A = (Q_L \times Q_R, \Sigma, S, (q_i, q_n), F_L \times F_R) \]

where \( S((q, q'), a) = (S_L(q, a), S_R(q', a)) \).

Note that \( A \) is constructed in such a way that \( w \) is accepted by \( A \) (i.e. \( w \in L(A) \)) if and only if \( w \) is accepted by \( A_L \) and \( A_R \) (i.e. \( w \in L(A_L) \) and \( w \in L(A_R) \)), i.e. if \( w \in L(A_L) \cap L(A_R) \) or \( w \in L \cap M \).

By applying this construction to the automata \( A_L^* \) and \( A_R^* \) we get:

Note that \( q_{i, j} \) is shorthand for \( (q_i, q_j) \).