NOTE the following definitions:

- The binary alphabet is the set \( \{0, 1\} \).
- A binary string is any string over the binary alphabet.
- If \( w \) is any string, then \( w^R \) (the reversal of \( w \)) is \( w \) written backwards, that is, comprising the symbols of \( w \) in reverse order.
- A string \( x \) is a prefix of a string \( y \) iff there exists a string \( z \) such that \( y = xz \).
- A string \( x \) is a suffix of a string \( y \) iff there exists a string \( z \) such that \( y = zx \).
- A string \( x \) is a substring of a string \( y \) if there exist strings \( w, z \) such that \( y = wxz \).

1. Consider the following DFA:

   (a) For each of the strings below, say which state the DFA is in after reading the string, and say whether or not the DFA accepts the string.

   \[ \text{aaa bb bbb abab bbbbbbbbbbbbbbaaa } \varepsilon \text{ aabbbbabbbaaabbaababababbb} \]

   (b) Give two different strings of length 4 that each make the DFA go from state 0 to state 1.
2. Draw a DFA with alphabet \( \{0, 1\} \) that accepts a binary string \( x \) iff \( x \) has odd length, i.e., iff \(|x|\) is odd.

3. Let \( A \) be the DFA given by the following tabular form:

\[
\begin{array}{c|cc}
& 0 & 1 \\
\rightarrow & q_0 & q_1 \\
q_0 & q_0 & q_1 \\
q_1 & q_2 & q_0 \\
q_2 & q_1 & q_2 \\
\end{array}
\]

(\( A \) accepts a binary string iff it represents a multiple of 3.) Recall the DFA described in class (here we’ll call it \( B \)) that accepts a binary string iff the string ends with 1:

\[
\begin{array}{c}
\text{A} \\
\text{0} & \rightarrow & \text{B} \\
\text{0} & \rightarrow & \text{1} \\
\text{1} & \rightarrow & \text{0} \\
\end{array}
\]

Recall the product construction from class. Draw the diagram for the product of \( A \) and \( B \) so the resulting DFA recognizes the language \( L(A) \cap L(B) \).

4. Describe a DFA \( B \) that accepts a string over the alphabet \( \{a, b, c\} \) iff its first and last symbols are different.

5. Consider the following two languages over the alphabet \( \{a, b\} \):

- \( L_1 = \{ w \mid w \) is either the empty string or ends with \( b \} \),
- \( L_2 = \{ w \mid \) there is a \( b \) followed by an \( a \) somewhere in \( w \} \).

(a) Draw a 2-state DFA recognizing \( L_1 \) and a 3-state DFA recognizing \( L_2 \).

(b) Using your answer and the product construction, draw a DFA recognizing \( L_1 \cap L_2 \). Do not perform any optimizations (e.g., removing unreachable states or transitions, or merging indistinguishable states).

6. Give the transition diagram for a DFA over the alphabet \( \Sigma = \{a, b, c\} \) that accepts a string \( w \) iff \( w \) contains \( ab \) as a substring but does not contain \( abb \) as a substring. What is the least number of states you need?

7. This exercise is adapted from Exercise 2.2.1 on pages 52–53, which is formulated somewhat vaguely. Consider the marble-rolling toy (redrawn from Figure 2.8):
A marble is dropped at $A$ or $B$. Levers $x_1$, $x_2$, and $x_3$ cause the marble to fall either to the left or to the right. Whenever a marble encounters a lever, it causes the lever to reverse after the marble passes, so the next marble will take the opposite branch.

Model this toy as a finite automaton. An input to the automaton is a string over the alphabet \{A, B\}, which represents a sequence of marbles being dropped into the toy. The toy is initially in the configuration above before any marbles are dropped (so that the first ball will exit at $C$ regardless of where it is dropped). Say that a sequence of marble drops is *accepted* exactly in the case that if one additional marble were to be dropped in, it would go out through $D$ regardless of where it was dropped.