NOTE: I stick with the tabular form in my answers, just because transition diagrams are time-consuming to typeset. Please don’t interpret what I do here as any discouragement from your giving transition diagrams in your answers.

1. Consider the following DFA:

![DFA Diagram]

(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.

<table>
<thead>
<tr>
<th>String</th>
<th>State</th>
<th>Acceptance</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaa bb bbb abab bbbbbbbbbbbaaa ε aabbbbababaaabbaabababbababbb</td>
<td>1</td>
<td>acc</td>
</tr>
</tbody>
</table>

(b) Give two different strings of length 4 that each make the DFA go from state 0 to state 1.

**Answer:**

(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.

<table>
<thead>
<tr>
<th>String</th>
<th>State</th>
<th>Acceptance</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaa bb bbb abab bbbbbbbbbbbaaa ε aabbbbababaaabbaabababbababbb</td>
<td>1</td>
<td>acc</td>
</tr>
</tbody>
</table>

(b) abba and abaa (also bbba)
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.

**Answer:** Here is the tabular form:

\[
\begin{array}{c|cc}
 & 0 & 1 \\
\hline
\rightarrow E & O & O \\
\ast O & E & E
\end{array}
\]

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

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

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

![Diagram of DFA A and B](image)

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

**Answer:** Here is the tabular form (note the consistent way I label and order the states):

\[
\begin{array}{c|ccc}
 & 0 & 1 \\
\hline
\rightarrow A_0 & A_0 & B_1 \\
A_1 & A_2 & B_0 \\
A_2 & A_1 & B_2 \\
*B_0 & A_0 & B_1 \\
B_1 & A_2 & B_0 \\
B_2 & A_1 & B_2
\end{array}
\]

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

**Answer:**

\[
\begin{array}{c|ccc}
 & a & b & c \\
\hline
\rightarrow 0 & A & B & C \\
A & A & A' & A' \\
*A' & A & A' & A' \\
B & B' & B & B' \\
*B' & B' & B & B' \\
C & C' & C' & C \\
*C' & C' & C' & C
\end{array}
\]
5. Consider the following two languages over the alphabet \( \{a, b\} \):

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

(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).

Answer:

(a)

\[
\begin{array}{c|cc}
 & a & b \\
\hline
\rightarrow & \ast X & Y \\
Y & Y & X
\end{array}
\]

(b)

\[
\begin{array}{c|ccc}
 & a & b & c \\
\hline
\rightarrow & X & Y & X \\
X & 0 & Y & X \\
\ast X & Y & 2 & X \\
Y & 0 & Y & 1X \\
Y & 1 & Y & 2Y & 1X \\
\ast Y & 2 & Y & 2X \\
Y & 2 & Y & 2X \\
\end{array}
\]

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?

Answer: Here are DFAs for the former and latter languages, respectively:

\[
\begin{array}{c|ccc}
 & a & b & c \\
\hline
\rightarrow & X & Y & X \\
Y & X & Y & Z \\
*Z & Z & Z & Z \\
\end{array}
\]

Using the product construction, we build the unoptimized (12-state) product DFA for the
States $X_1, X_2, X_3, Y_0, Y_2, Y_3$ are unreachable from $X_0$ and can be eliminated to obtain the following 6-state sane DFA:

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_0$</td>
<td>$Y_1$</td>
<td>$X_0$</td>
<td>$X_0$</td>
</tr>
<tr>
<td>$Y_1$</td>
<td>$Y_1$</td>
<td>$X_2$</td>
<td>$X_0$</td>
</tr>
<tr>
<td>*$Z_0$</td>
<td>$Z_1$</td>
<td>$Z_0$</td>
<td>$Z_0$</td>
</tr>
<tr>
<td>*$Z_1$</td>
<td>$Z_1$</td>
<td>$Z_2$</td>
<td>$Z_0$</td>
</tr>
<tr>
<td>*$Z_2$</td>
<td>$Z_1$</td>
<td>$Z_3$</td>
<td>$Z_0$</td>
</tr>
<tr>
<td>$Z_3$</td>
<td>$Z_3$</td>
<td>$Z_3$</td>
<td>$Z_3$</td>
</tr>
</tbody>
</table>

This DFA is minimal (all pairs of states are distinguishable) and alone suffices for an answer.