1 Exercise 1

Take $M_1$, a diagram for a deterministic FSA.

Give the formal description of $M_1$, where $M_1 = \langle Q, \Sigma, \delta, q_0, F \rangle$, where

1. $Q = \{q_1, q_2, q_3, q_4\}$
2. $\Sigma = \{a, b\}$
3. $\delta$ is defined as

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_1$</td>
<td>$q_1$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
<td>$q_4$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_2$</td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>$q_2$</td>
<td>$q_4$</td>
</tr>
</tbody>
</table>

4. The start state is $q_1$
5. $F = \{q_1, q_4\}$

Which of the following strings does $M_1$ accept?

a. aabbb yes  
   b. baaba no  
   c. bbba no  
   d. abab yes  
   e. abaa no  
   f. a yes  
   g. bab yes  
   h. $\epsilon$ yes  
   i. aaaaa yes  
   j. aaabaaa no
2 Exercise 2

Draw a diagram for the formal description of $M_2$: $M_2 = \langle Q, \Sigma, \delta, q_0, F \rangle$, where

1. $Q = \{q_1, q_2, q_3, q_4, q_5\}$
2. $\Sigma = \{u, d\}$
3. $\delta$ is defined as

<table>
<thead>
<tr>
<th>$u$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_1$</td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>$q_3$</td>
</tr>
<tr>
<td>$q_5$</td>
<td>$q_4$</td>
</tr>
</tbody>
</table>

4. The start state is $q_3$.
5. $F = \{q_3\}$.

Which of the following strings does $M_2$ accept?

a. $d$ NO
b. $du$ YES
c. $udu$ NO
d. $dudu$ YES
e. $uuudu$ NO
f. $uuuddddddduu$ YES
g. $duuuddu$ NO
h. $\epsilon$ YES
i. $duddddduudu$ YES
j. $uudddu$ NO

3 Exercise 3

Exercise 1 in Partee et al., page 480.

i NO
ii YES
iii YES
iv YES

The FSA accepts strings $w$, s.t. $w$ contains $n$-many “1”s, not necessarily consecutive, where $n = 2 + 3m$ ($m$ is 0 or natural #), and any number of “0”s.
4 Exercise 4

Show that the following languages are regular. That is, construct state diagrams for (deterministic) FSA that recognize the following languages. For each language, assume that \( \Sigma = \{1,0\} \) and use as few states as possible.

i) \( \{w: \text{the length of } w \text{ is at most } 5\} \)

ii) \( \{\epsilon, 10, 01\} \)

iii) \( \{w: w \text{ contains an even number of } 0s, \text{ not necessarily consecutive (and any number of } 1s \text{ in any order)}\} \)

iv) \( \{w: w \text{ contains no } 0 \text{ and contains a total of } n \text{ } 1s, \text{ where } n = 4m + 1, \text{ for } m = 0 \text{ or } m \text{ is a natural number}\} \)

v) \( \{w: w \text{ contains the sub-string } 0101\} \)

vi) \( \{w: w \text{ is any string except } 11 \text{ and } 111 \} \)

i) Use different states to mark your progress:

![State Diagram 1](image1)

We do not distinguish between cases where we get 0 or 1, so every arrow is labeled with both.

ii) Use different states to distinguish different cases (00 vs. 01, 10 vs. 11) and to mark your progress. Never have more than one trash state. Try to reduce the number of accept states as much as possible.

![State Diagram 2](image2)

Keeping track of whether we had 0 or 1, to later distinguish 10 and 00.

iii) Even = any number that gives an integer when divided by 2.

So, 0 / 2 = 0 (integer), so 0 is even.

Note: when we already have an even # of "0"s, we're as happy as if we hadn't done anything - one more 0 makes us unhappy, while two more make us happy. So, every two "0"s we come back to the start state.
iv) Note that after we've had four "1"s, we're back at square one: one more "1" makes us happy. So, every four "1"s we come back to the start state:

Again, we're using different states to mark our progress.

v) Use different states to mark how far along "0101" you are:

vi) This is actually easy: build FSA that accepts only 11 and 111, then switch to accept and non-accept states!

Again, different states are used to distinguish different cases:

vi) This is actually easy: build FSA that accepts only 11 and 111, then switch to accept and non-accept states!

vi) This is actually easy: build FSA that accepts only 11 and 111, then switch to accept and non-accept states!