1. For the following FSM, $M$, document its parts by listing each of the following:
   a) alphabet b) states c) starting state d) accepting states e) transitions.
   List three strings (from $\Sigma^*$) that $M$ accepts.
   List three strings (from $\Sigma^*$) that $M$ does not accept.

**Solution:**

a) alphabet $\Sigma = \{a, b\}$

b) states $Q = \{0, 1, 2, 3, 4\}$

c) starting state $q_0 = 0$

d) accepting states $F = \{1, 3\}$

e) transitions: (hint: be systematic in how you make your list either order by states or by inputs

$$f(0, a) = 1, f(1, a) = 1, f(2, a) = 1, f(3, a) = 4, f(4, a) = 4$$
2. Design an FSM to recognise each of the following languages:

a) Words using only the letters \{a, b, c\} that contain no c.
b) Words using only the letters \{a, b\} such that the last two symbols are the same.
c) Words using only the letters \{a, b\} of the form \((ab)^n\) for \(n > 0\) (that is \(ab\) repeated \(n\) times. 
d) Words using only the letters \{a, b\} that contain exactly 2 bs.

**Solution:**
3. Define a divisible by 3 checker as a machine that accepts a string from the alphabet 0, 1 if and only if the sum of its digits is divisible by 3. Design an FSA to implement a divisible by 3 checker.

**Solution:** This definition implies the string contains exactly 0, 3, 6, 9, ... 1s and any number of 0s.

![Diagram of an FSA for divisibility by 3](image)

4. Design a nondeterministic FSM (NFSM) to recognise any binary string that ends in 01. Hint: you only need 3 states.

**Solution:**

![Diagram of an NFSM for the language ending in 01](image)

5. Convert your NFSM from the previous question into a deterministic FSM for the same language.

**Solution:**

Note that many of these states can not be reached from the initial state. I have left them in to remind you of the construction. See Sipser for details of the construction algorithm.
6. Devise an FSM to solve the farmer, wolf, goat and cabbage problem and find the solutions. This problem involves a farmer, a wolf, a goat and a cabbage all on one side of a river. There is a boat but the farmer can carry only one passenger at a time. The farmer wants to get them all to the other side of the river. However, left alone, the wolf will eat the goat, and the goat will eat the cabbage. How do they all get to the other side?
7. **Source: Sipser 1.12**

Let \( D = \{ w \mid w \text{ contains an even number of as and} \)

an odd number of bs and

does not contain the substring \( ab \} \)

Give a FSM with 5 states that recognises \( D \).

Hint: describe \( D \) more simply.

**Solution:**

The reasoning is really more important than the answer here:

First, think about the machines for an even number of as (2 states)
and an odd number of bs (3 states).

Remember that 0 is even, but not odd. So you must have 1 or more
bs, but can have 0 as.

Next, notice that since \( D \) can not ever contain the substring \( ab \) that
all the bs have to occur before the as. So we can concatenate (that
is, join in sequence) our machines for the odd bs and even as.

The only addition that is needed is that you need an extra state to
move into the even a counting part, so that the machine will not
accept any more bs.
Here is a machine that does it, but not showing the non-accepting transitions.

Here is the same machine but with the transitions to the non-accept state added.