1. Design a Turing Machine M for $\Sigma = \{0, 1\}$ that accepts the language $L = \{0^n 1^n | n \geq 1\}$

   Solution : The heart of a TM are the transition functions and if you can figure out δ transitions then it is easy to solve the TM problems. In this, the idea is accept strings with n 0's followed by n 1's.

   - Start at the leftmost 0, replace it with another symbol, X.
   - The read-write head then travels right to find the leftmost 1, which is replaced by another symbol, Y.
   - We now need to go back left to the leftmost 0, replace it with an X, and again move to the leftmost 1, replace it with Y and so on.
   - Thus, we try to match each 0 with a corresponding 1.
   - Accept the language when no more 0's or 1's remain.
   - Thus the transitions can be written as

     $$\delta(q_0, 0) = (q_1, X, R)$$

     $$\delta(q_1, 0) = (q_1, 0, R)$$
     $$\delta(q_1, Y) = (q_1, Y, R)$$
     $$\delta(q_1, 1) = (q_2, Y, L)$$

     Now to move back to the first 0 we can add the following transitions,

     $$\delta(q_2, Y) = (q_2, Y, L)$$

     $$\delta(q_2, 0) = (q_2, 0, L)$$
     $$\delta(q_2, X) = (q_0, X, R)$$

     To terminate, one must perform a final check to ensure all 0's and 1's have been replaced

     $$\delta(q_0, Y) = (q_3, Y, R)$$

     $$\delta(q_3, Y) = (q_3, Y, R)$$
     $$\delta(q_3, B) = (q_f, B, R)$$

     Thus the TM M can be defined as

     $$Q = \{q_0, q_1, q_2, q_3, q_4\}$$

     $$F = \{q_f\}$$
     $$\Sigma = \{a, b\}$$
     $$\Gamma = \{a, b, X, Y, B\}$$

     You can next draw the transition diagram from the transitions and definition.

2. Design a TM to accept the language $L = \{a^n b^n c^n | \ n > 0\}$

Solution : Going by the above explanation we see that we can extend the same idea over three symbols.

- Mark the left most a with X.
- Scan right to reach the leftmost unmarked b. In no such b's exist then crash.
- Mark the leftmost b with Y.
- Scan right to reach the leftmost unmarked c. If no such c's exist then crash.
- Mark the leftmost c with Z.
- Check to see that there are no unmarked a's or b's and c's and then accept.
- The transitions can then be defined as

  To replace a,b,c with X,Y,Z respectively

$$\delta(q_0, a) = (q_1, X, R)$$
$$\delta(q_1, a) = (q_1, a, R)$$
$$\delta(q_1, Y) = (q_1, Y, R)$$
$$\delta(q_1, b) = (q_2, Y, R)$$
$$\delta(q_2, b) = (q_2, b, R)$$
$$\delta(q_2, Z) = (q_2, Z, R)$$
$$\delta(q_2, c) = (q_3, Z, L)$$

  To reverse the direction and move back to the initial state we can define the following transitions. When looking for the leftmost 'a' the read-write head travels left with the machine in state q_3, when an X is encountered, the direction is reversed to get the a. (Idea is to position the heat on leftmost a and return control to the initial state.)

$$\delta(q_3, Z) = (q_3, Z, L)$$
$$\delta(q_3, Y) = (q_3, Y, L)$$
$$\delta(q_3, b) = (q_3, b, L)$$
$$\delta(q_3, c) = (q_3, c, L)$$
$$\delta(q_3, X) = (q_0, X, R)$$

Finally to accept we need to check if all a's,b's and c's are replaced or not. Instead of finding a now, it will find a 'Y'. to terminate, a final check is made to see it all a's , b's and c's have been replaced.

$$\delta(q_0, Y) = (q_4, Y, R)$$

$$\delta(q_4, Y) = (q_4, Y, R)$$

$$\delta(q_4, Z) = (q_4, Z, R)$$

$$\delta(q_4, B) = (q_f, B, R)$$

Hence,

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_f\}$$

$$\Sigma = \{a, b, c\}$$

$$\Gamma = \{a, b, c, X, Y, Z, B\}$$

$$F = \{q_f\}$$

Now that you have an idea about how to write transition functions we now ignore the functions for the following questions.

3. Construct a Turing Machine M for $L = \{ww | w \in \{0, 1\}^*\}$

   Solution : So it accepts a even length string which is basically a concatenation of a string with its copy. (Kleinberg notes)

   The machine M works as follows.

   - On input x, scan out to the first blank symbol $\sqcup$, counting the number of symbols mod 2 to make sure x is of even length and rejecting immediately if not.
   - Put $\dashv$ then repeatedly scan back and forth over the input.
   - In each pass from right to left, mark the first unmarked a or b it sees with $\acute{a}$ or $\acute{b}$.
   - In each pass from left to right, mark the first unmarked a or b it sees with $\grave{a}$ or $\grave{b}$.
   - Continue until all symbols are marked.

   For example the input string is *aabbaaabba*

$$\vdash a\ a\ b\ b\ a\ a\ a\ b\ b\ a \dashv \sqcup \sqcup \sqcup ...$$

$$\vdash a\ a\ b\ b\ a\ a\ a\ b\ b\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

$$\vdash \grave{a}\ a\ b\ b\ a\ a\ a\ b\ b\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

$$\vdash \grave{a}\ a\ b\ b\ a\ a\ a\ b\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

$$\vdash \grave{a}\ \grave{a}\ b\ b\ a\ a\ a\ b\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

...

$$\vdash \grave{a}\ \grave{a}\ \grave{b}\ \grave{b}\ \grave{a}\ \acute{a}\ \acute{a}\ \acute{b}\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

The machine then repeatedly scans left to right over the input. In each pass it replaces the first $\grave{a}$ or $\grave{b}$ with $\sqcup$ but remembers that symbol in its finite control.
It then scans forward until it sees the first $\acute{a}$ or $\acute{b}$, checks if it is the same, and replaces it with $\sqcup$. If the two symbols are not the same, it rejects. Otherwise, when it has erased all the symbols, it accepts. In our example, the following would be the tape contents after each pass.

$$\grave{a}\ \grave{a}\ \grave{b}\ \grave{b}\ \grave{a}\ \acute{a}\ \acute{a}\ \acute{b}\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

$$\sqcup\ \grave{a}\ \grave{b}\ \grave{b}\ \grave{a}\ \sqcup\ \acute{a}\ \acute{b}\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

$$\sqcup\ \sqcup\ \grave{b}\ \grave{b}\ \grave{a}\ \sqcup\ \sqcup\ \acute{b}\ \acute{b}\ \acute{a} \dashv \sqcup \sqcup \sqcup ...$$

...

$$\sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \sqcup\ \dashv \sqcup \sqcup \sqcup ...$$

Thus

$$\Gamma = \{a, b, \vdash, \dashv, \sqcup, \grave{a}, \grave{b}, \acute{a}, \acute{b}\}$$