Solutions for Homework Assignment #1

Answer to Question 1. The idea is that we will keep subtracting 1 from $\alpha$, and marking one symbol off of $\beta$, until $\alpha$ represents the integer 0.

Before starting the main loop (beginning in state $q_4$), we check that the input is of the form $\alpha\#\beta$ followed by blanks ($\sim$ is the blank symbol) where $\epsilon \neq \alpha \in \{0, 1\}^*$ and $\beta \in \{0, 1\}^*$.

After we have done the main loop $i$ times, the tape will consist of $\alpha_i\#\beta_i\sim\sim\ldots$ where:

- $\alpha_i$ is the result of subtracting $i$ from $\alpha$, with the leftmost symbol marked (0' instead of 0, or 1' instead of 1).
- $\beta_i$ is the result of marking the leftmost $i$ symbols of $\beta$.
- The head is on the square to the left of $#$ and the state is $q_4$.

The tape alphabet will be $\{0, 1, 0', 1', \#, \sim\}$.
The state set will be $\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{\text{accept}}, q_{\text{reject}}\}$ where $q_0$ is the initial state.
The machine transitions are as follows. We have explicitly included the rejecting transitions, but we have omitted those transitions that cannot occur.

The initialization transitions check the form of the input; if the form is correct, we end up in state $q_4$ with $i = 0$ and the head just to the left of $#$, and with the leftmost bit marked; the transitions are as follows:

$$
\begin{align*}
\delta(q_0, 0) &= (q_1, 0', R) & \delta(q_0, 1) &= (q_1, 1', R) & \delta(q_0, \#) &= \delta(q_0, \sim) &= (q_{\text{reject}}, [\text{anything}], [\text{anything}]) \\
\delta(q_1, 0) &= (q_1, 0, R) & \delta(q_1, 1) &= (q_1, 1, R) & \delta(q_1, \#) &= (q_2, \#, R) \\
\delta(q_1, \sim) &= (q_{\text{reject}}, [\text{anything}], [\text{anything}]) \\
\delta(q_2, 0) &= (q_2, 0, R) & \delta(q_2, 1) &= (q_2, 1, R) & \delta(q_2, \sim) &= (q_3, \sim, L) \\
\delta(q_2, \#) &= (q_{\text{reject}}, [\text{anything}], [\text{anything}]) \\
\delta(q_3, 0) &= (q_3, 0, L) & \delta(q_3, 1) &= (q_3, 1, L) & \delta(q_3, \#) &= (q_4, \#, L)
\end{align*}
$$

Note that if $\alpha$ is empty or is not followed by $\#$, the machine rejects. After seeing $\#$, the machine moves to the right past the 0’s and 1’s to make sure that they are followed by $\sim$ (and not $\#$). It then moves left to the square just to the left of $#$, entering state $q_4$.

The next set of transitions subtracts 1 from $\alpha$. We start in state $q_4$ with the head just to the left of $\#$. We go left, changing 0’s to 1’s until we either find a 1 or reach the end of the tape; if we find a 1, we change it to 0 (thereby completing the subtraction) and enter state $q_5$, and then go to the right just past the $\#$ ending up in state $q_6$; if we reach the end of the tape without finding 1 (because $\alpha$ consisted of all 0’s) we will enter state $q_7$ and then go to the right just past the $\#$ ending up in state $q_8$.

$$
\begin{align*}
\delta(q_4, 0) &= (q_4, 1, L) & \delta(q_4, 0') &= (q_7, 1', R) \\
\delta(q_4, 1) &= (q_5, 0, R) & \delta(q_4, 1') &= (q_5, 0', R) \\
\delta(q_5, 1) &= (q_5, 1, R) & \delta(q_5, \#) &= (q_6, \#, R) \\
\delta(q_7, 1) &= (q_7, 1, R) & \delta(q_7, \#) &= (q_8, \#, R)
\end{align*}
$$

The next set of transitions deals with the case where we have successfully subtracted 1 from $\alpha$. We are in state $q_6$ and the head is just to the right of $\#$. We want to mark the next bit of $\beta$, by...
and then wind up back in state $q_4$ with the head just to the left of #; if there is no next bit of $\beta$ to mark, then we want to reject.

$$
\delta(q_6, 0') = (q_6, 0', R) \quad \delta(q_6, 1') = (q_6, 1', R) \quad \delta(q_6, 0) = (q_9, 0', L) \quad \delta(q_6, 1) = (q_9, 1', L)
$$

$$
\delta(q_6, \sim) = (q_{\text{reject}}, [\text{anything}], [\text{anything}])
$$

$$
\delta(q_9, 0') = (q_9, 0', L) \quad \delta(q_9, 1') = (q_9, 1', L) \quad \delta(q_9, #) = (q_4, #, L)
$$

The next set of transitions deals with the case where we could not subtract 1 from $\alpha$ because it consisted of all 0’s. We want to accept if at least one bit of $\beta$ is marked, and the rightmost marked bit of $\beta$ is 1; otherwise we want to reject. State $q_8$ will remember that the last seen marked bit of $\beta$ was 0, or that no such bits have been seen yet; state $q_{10}$ will remember that the last seen marked bit of $\beta$ was 1.

$$
\delta(q_8, 0') = (q_8, 0', R) \quad \delta(q_8, 1') = (q_{10}, 1', R)
$$

$$
\delta(q_{10}, 0') = (q_8, 0', R) \quad \delta(q_{10}, 1') = (q_{10}, 1', R)
$$

$$
\delta(q_{10}, 0) = \delta(q_{10}, 1) = \delta(q_{10}, \sim) = (q_{\text{accept}}, [\text{anything}], [\text{anything}])
$$

$$
\delta(q_8, 0) = \delta(q_8, 1) = \delta(q_8, \sim) = (q_{\text{reject}}, [\text{anything}], [\text{anything}])
$$

**Answer to Question 2.** $M'$ will work as follows. Whenever $M$ wants to enter state $p$ without moving the head, $M'$ will enter state $\hat{p}$ and move right (where $\hat{p}$ is a new state), and then enter state $p$ and move left. So we have $M' = (Q', \Sigma, \Gamma, \delta', q_0, q_{\text{accept}}, q_{\text{reject}})$ where $Q' = Q \cup \{\hat{p} | p \in Q\}$, and $\delta'$ as follows.

First, consider $q \in Q - \{q_{\text{accept}}, q_{\text{reject}}\}$ and $a \in \Gamma$. If $\delta(q, a) = (p, b, d)$ where $d \in \{R, L\}$, then $\delta'(q, a) = (p, b, d)$. If $\delta(q, a) = (p, b, S)$, then $\delta'(q, a) = (\hat{p}, b, R)$.

Now consider $\hat{p}$ where $p \in Q$. Then for all $a \in \Gamma$, $\delta'(\hat{p}, a) = (p, a, L)$.

**Answer to Question 3.**

a. The following algorithm accepts $\overline{L}$.

Given a string, compute if it is of the form $\langle M \rangle$ where $M$ is a Turing machine over $\{0, 1\}$, and if not accept. So assume the input is $\langle M \rangle$ where $M$ is a Turing machine over $\{0, 1\}$. Run (or simulate) $M$ on $\langle M \rangle^R$; if and when the simulation accepts, accept; if and when the simulation rejects, reject (or go into an infinite loop).

b. Let $L' \subseteq \{0, 1\}^*$ be the language $L' = \{x^R | x \in L\}$. We will first show that $L'$ is not recognizable. So let $\langle M \rangle$ be an arbitrary Turing machine over $\{0, 1\}$. We want to show that $L' \neq \mathcal{L}(M)$. This follows immediately from the fact that $\langle M \rangle^R \in L' \iff \langle M \rangle \in L \iff \langle M \rangle^R \notin \mathcal{L}(M)$.

If $L$ were decidable, then $L'$ would also be decidable, and it is not.

**Answer to Question 4.** Let $S \subseteq \Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ be the set of digits that occur infinitely often in $\alpha$. We can write $\alpha = \alpha_0\alpha_1$ where $\alpha_0 \in \Sigma^*$ and where $\alpha_1$ is an infinite string consisting only of members of $S$, and where every member of $S$ occurs infinitely often. Define the finite language $L_0 \subseteq \Sigma^*$ by

$$
L_0 = \{\beta \in \Sigma^* | \beta \text{ is a subsequence of } \alpha_0\}.
$$

The following algorithm decides $L$. $S$ and $L_0$ will be built into the algorithm.

Given a string $w \in \Sigma^*$: If $w \in S^*$, then accept. So assume $w \notin S^*$. Then we can compute $w = w_0w_1$ where $w_0$ ends with a digit not in $S$ and where $w_1 \in S^*$. Then accept if $w_0 \in L_0$, otherwise reject.

(Do you see why this works?)