CS 481 Homework #8 Problem 1

PDA for \{a, b, c\}^* - \{a^nb^nc^n | n \geq 0\}

Initial guess

$$\delta(q_1, a, Z_0) = \{(q_1, Z_0), (q_3, Z_0), (q_6, Z_0), \text{special cases}\}$$

Case where \{a^ib^jc^k | i \neq j \& k \geq 0\}. Also handles strings of the form \{a^ic^k | i > 0, k \geq 0\} and \{b^jc^k | j > 0, k \geq 0\}

$$\delta(q_1, a, Z_0) = \{(q_1, AZ_0)\} \quad \text{// count number of a’s}$$
$$\delta(q_1, a, A) = \{(q_1, AA)\} \quad \text{// continue counting a’s}$$
$$\delta(q_1, A, A) = \{(q_{f3}, A)\} \quad \text{// anytime there’s an A on stack, it means i > j, so accept}$$
$$\delta(q_1, b, Z_0) = \{(q_{f1}, Z_0)\} \quad \text{// see a b with no prior a’s, i.e. i < j. accept}$$
$$\delta(q_1, c, A) = \{(q_{f2}, \varepsilon)\} \quad \text{// see c after a’s, i.e. i > j (j = 0). accept}$$
$$\delta(q_1, b, A) = \{(q_2, \varepsilon)\} \quad \text{// start seeing b’s. start popping A’s}$$
$$\delta(q_2, b, A) = \{(q_2, \varepsilon)\} \quad \text{// keep popping A’s as we see b’s}$$
$$\delta(q_2, c, A) = \{(q_{f2}, \varepsilon)\} \quad \text{// see c before we’ve popped all A’s, i.e. i > j. accept}$$
$$\delta(q_{f1}, b, Z_0) = \{(q_{f1}, Z_0)\} \quad \text{// keep accepting any b’s}$$
$$\delta(q_{f1}, c, Z_0) = \{(q_{f1}, Z_0)\} \quad \text{// keep accepting any c’s}$$
$$\delta(q_{f2}, c, Z_0) = \{(q_{f2}, Z_0)\} \quad \text{// keep accepting any c’s}$$
$$\delta(q_{f2}, c, A) = \{(q_{f2}, \varepsilon)\} \quad \text{// keep accepting any c’s}$$

Case where \{a^ib^jc^k | j \neq k \& i \geq 0\}. Also handles strings of the form \{a^ic^k | i > 0, k \geq 0\} and \{a^ib^j | i \geq 0, j > 0\}

$$\delta(q_3, a, Z_0) = \{(q_3, Z_0)\} \quad \text{// disregard all a’s}$$
$$\delta(q_3, c, Z_0) = \{(q_{f2}, Z_0)\} \quad \text{// see c’s with no prior b’s, i.e. j < k. accept.}$$
$$\delta(q_3, b, Z_0) = \{(q_{f4}, BZ_0), (q_{f4}, Z_0)\} \quad \text{// count number of b’s and guess that string only contains b’s}$$
$$\delta(q_4, b, B) = \{(q_{f4}, BB)\} \quad \text{// continue counting b’s}$$
$$\delta(q_4, c, B) = \{(q_{f5}, \varepsilon)\} \quad \text{// see a c so start popping B’s}$$
$$\delta(q_4, c, B) = \{(q_{f4}, B)\} \quad \text{// anytime there’s a B on stack, it means j > k, so accept}$$
$$\delta(q_5, c, B) = \{(q_{f5}, \varepsilon)\} \quad \text{// keep popping B’s}$$
$$\delta(q_5, c, B) = \{(q_{f4}, B)\} \quad \text{// saw c’s but there’re still B’s on stack, so j > k. accept}$$
$$\delta(q_5, c, Z_0) = \{(q_{f2}, Z_0)\} \quad \text{// ran out of B’s with more c’s, i.e. j < k. accept}$$

Case where \{a^ib^jc^k | i \neq k \& j \geq 0\}. Also handles strings of the form \{a^ib^j | i > 0, j \geq 0\} and \{b^jc^k | j \geq 0, k \geq 0\}

$$\delta(q_6, a, Z_0) = \{(q_6, AZ_0)\} \quad \text{// count number of a’s}$$
$$\delta(q_6, a, A) = \{(q_6, AA)\} \quad \text{// keep counting a’s}$$
$$\delta(q_6, A, A) = \{(q_{f5}, A)\} \quad \text{// anytime there’s an A on stack, it means i > k, so accept}$$
$$\delta(q_6, b, Z_0) = \{(q_6, Z_0)\} \quad \text{// no a’s, disregard b’s}$$
$$\delta(q_6, b, A) = \{(q_6, A)\} \quad \text{// saw some a’s, disregarding any b’s}$$
$$\delta(q_6, c, Z_0) = \{(q_{f2}, Z_0)\} \quad \text{// see a c without any prior a’s, i.e. i < k. accept}$$
$$\delta(q_6, c, A) = \{(q_7, \varepsilon)\} \quad \text{// see a c, so start popping A’s}$$
\[ \delta(q_7, c, A) = \{(q_7, \varepsilon)\} \quad \text{// continue popping A's} \]
\[ \delta(q_7, c, Z_0) = \{(q_{i2}, Z_0)\} \quad \text{// ran out of A's with more c's, i.e. i < k. accept} \]

Remaining special cases to handle:

Strings of the form \(a^+b^+a^+(a + b + c)^*\) and \(a^+b^+c^+(a + b + c)^+\) (this takes care of strings of the form \(a^n b^n c^n (a + b + c)^* \mid n \geq 0\))
Problem 2

(a) $S \rightarrow 0S1 \mid 1S0 \mid \varepsilon$.

(b) The following is a PDA for $L$. Z is the bottom of stack symbol, e is $\varepsilon$. 

\[ 
\begin{align*}
q_0 & \rightarrow 0, Z / 0 Z \\
& \rightarrow 1, Z / 1 Z \\
& \rightarrow 0, 0 / 0 0 \\
& \rightarrow 0, 1 / 0 1 \\
& \rightarrow 1, 0 / 1 0 \\
& \rightarrow 1, 1 / 1 1 \\
q_1 & \rightarrow 0, 1 / e \\
& \rightarrow 1, 0 / e \\
& \rightarrow e, Z / Z \\
& \rightarrow e, 0 / 0 \\
& \rightarrow e, 1 / 1 \\
q_2 & \rightarrow e, Z / Z
\end{align*} \]
Problem 3a. It is easy to show that this language is not regular. Consider a homomorphism $h : \Sigma \cup Q \cup \{\$\} \rightarrow \Sigma \cup \{\varepsilon\}$ that maps $h(a) = a$ for all $a \in \Sigma$ and $h(\$) = h(p) = \varepsilon$ for all $p \in Q$. Then the image of the given language is the language $\{ww^R \mid w \in \Sigma^+\}$ which we know is not regular.

Problem 3b. In order to show it is regular, we construct a PDA. The errors in the input $v$ can be of the following types. The PDA can simultaneously check for both type of errors.

- $v$ might not be of the form $(w_1, q, aw_2)\$(w_1', p, a'w_2')$, where each $w_i, w_i' \in \Sigma^*$, $a, a' \in \Sigma$ and $p, q \in Q$, but this is just testing for a regular language.

- If $v$ is of the form $v = (w_1, q, aw_2)\$(w_1', p, a'w_2')$, the automata first pushes $(w_1, q, aw_2)$ onto the stack and remembers the state $q$ in its finite state. Then, while reading the portion $w_1'$ from $(w_1', p, a'w_2')$ the machine pops $w_2$ off the stack to check whether $w_2^R = w_1'$. The last symbol $a$ from $w_2$ is stored in finite state and the machine reads $p$ and $a'$ and checks whether $a = a'$ and if $\delta(q, a) = p$. Finally, the last part to check is whether $w_2' = w_1^R$.

The correctness of the automata is obvious, since any string not in the language can only be due to one of the errors discussed above. For a induction proof, a machine construction and induction on strings can be done.