CS2349 – Spring 2022 – Homework 2

Manish Yadav

Solution 1:
a)

Implementation of Two-stack PDA for the language $a^n b^n c^n$:

1. Push “a” on stack first whenever we see an “a”.
2. Pop “a” from stack first and push “b” on stack two when you see “b”.
3. Pop b from stack two when you see “c”.
4. Accept if both stacks are empty at the end of this process.

We can define 2-PDA as

$$M = (Q, \Sigma, \Gamma_1, \Gamma_2, \delta, q_0, Z_1, Z_2, F)$$

here,

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

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

$\Gamma_1 = \{a, Z\}$

$\Gamma_2 = \{b, X\}$

$Z_1 = \{Z\}$ **{stack1}**

$Z_2 = \{X\}$ **{stack2}**

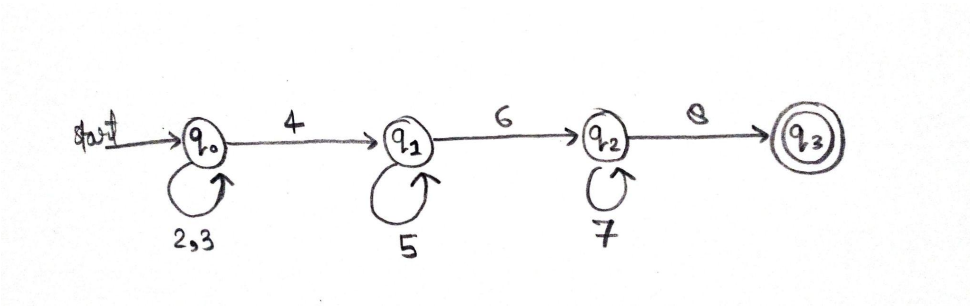
$F = \{q_3\}$

Transition Table:

	State	**Input**	**Transition Function**	**Stack(leftmost symbol is the top symbol of the stack)**	**State after transition**
1	q_0	$aabbcc$		$Z\ ,\ X$	q_0
2	q_0	$\overline{a}abbcc$	$\delta(q_0,\ a, Z\ , X) = (q_0, a,\ aZ,\ X)$	$aZ\ ,\ X$	q_0
3	q_0	$a\overline{a}bbcc$	$\delta(q_0,\ a, a, X) = (q_0,\ aa,\ X)$	$aaZ\ ,\ X$	q_0
4	q_0	$aa\overline{b}bcc$	$\delta(q_0,\ b, a, X) = (q_1,\ \varepsilon,\ bX)$	$aZ\ ,\ bX$	q_1
5	q_1	$aab\overline{b}cc$	$\delta(q_1,\ b, a, b) = (q_1,\ \varepsilon, bb)$	$Z\ ,\ bbX$	q_1
6	q_1	$aabb\overline{c}c$	$\delta(q_1,\ c, Z, b) = (q_2,\ Z, \varepsilon)$	$Z\ ,\ bX$	q_2

7	q_2	$aabbc\bar{c}$	$\delta(q_2, c, Z, b) = (q_2, Z, \varepsilon)$	Z, X	q_2
8	q_2	ϵ	$\delta(q_2, \varepsilon, Z, X) = (q_3, Z, X)$	Z, X	q_3

Automation Diagram:
Considering the above table, numbering below depicts the rows of the table.



This language is recursive and Turing recognizable as there exists a Turing machine that can recognize it. We can simulate the Turing machine by replacing a with another symbol (p) and erasing a matching c from the tape. Further, erase the matching p and b from the tape. The string will be accepted if there is nothing left on the tape.

b)

Implementation of non-deterministic PDA for the language ww^R:

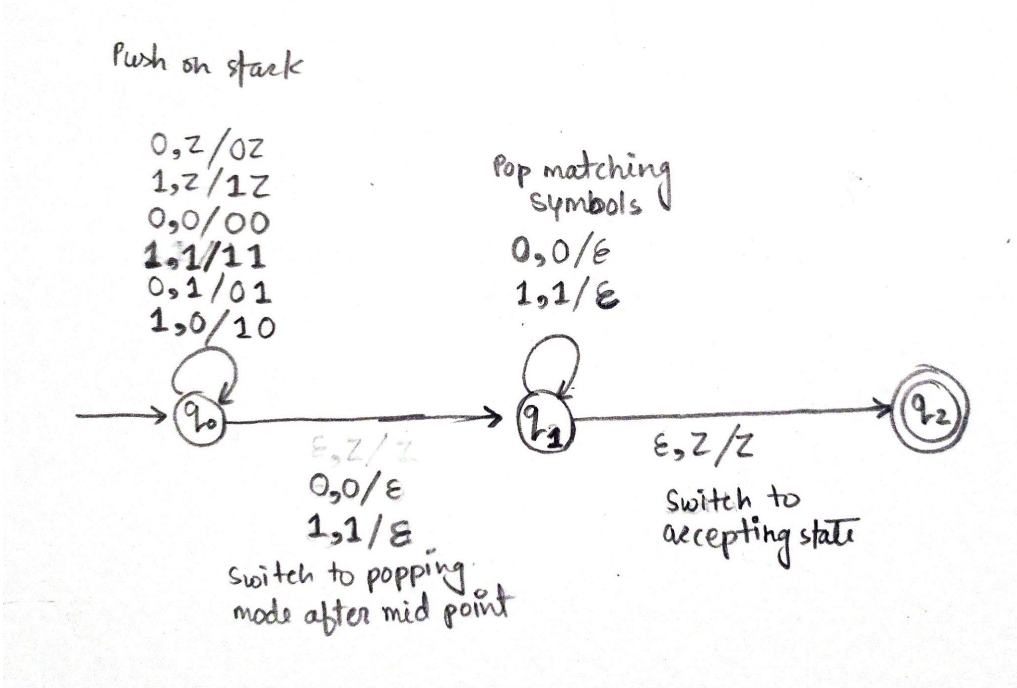
- Read the input in the string and push it on the stack.
- At each character, check if you've reached halfway in the input string.
- Once reaching the midpoint, pop characters from the stack if input characters match with it.
- Accept if every character matches and nothing is left on the stack at the end.

Transition Table:

	State	Transition Function	State after transition
1	q_0	$\delta(q_0, 0, Z) = (q_0, 0Z)$	q_0
2	q_0	$\delta(q_0, 0, 0) = (q_0, 00)$	q_0
3	q_0	$\delta(q_0, 1, Z) = (q_0, 1Z)$	q_0
4	q_0	$\delta(q_0, 1, 1) = (q_0, 11)$	q_0
5	q_0	$\delta(q_0, 1, 0) = (q_0, 10)$	q_0
6	q_0	$\delta(q_0, 0, 1) = (q_0, 01)$	q_0

7	q_0	$\delta(q_0, 0 , 0) = (q_1, \varepsilon)$	q_1
8	q_0	$\delta(q_0, 1 , 1) = (q_1, \varepsilon)$	q_1
9	q_1	$\delta(q_1, 0 , 0) = (q_1, \varepsilon)$	q_1
10	q_1	$\delta(q_1, 1 , 1) = (q_1, \varepsilon)$	q_1
11	q_1	$\delta(q_1, \varepsilon , Z) = (q_2, \varepsilon)$	q_2

Automation Diagram:



This language is Turing recognizable as string is accepted only if it reaches into the accepting state. We can define the Turing machine as well which recognize this language.