National University of Computer & Emerging Sciences, Karachi
Spring-2023 CS-Department
Final Examination
May 30, 2023, 08:30 a.m – 11:30 a.m

Course Code: CS3001	Course Name: Theory of Automata
Instructor Names: Mr. Shahzad,Shaharbano,Bakhtawar, Faisal, Minhal	
Student Roll No:	Section No:

Time: 60 minutes. **Max Marks: ??** points

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are 3 **questions on 1 page 2 sides**.
- In case of any ambiguity, you may make assumptions. But your assumption should not contradict any statement in the question paper.
- Start each question in a new page.

Question-1 Kleene's Theorem: **[CLO-4 , 5+5 Points,35 mins]**

a: **15 mins, 5 Points**

Find the Closure of FA1 given in Figure 1.

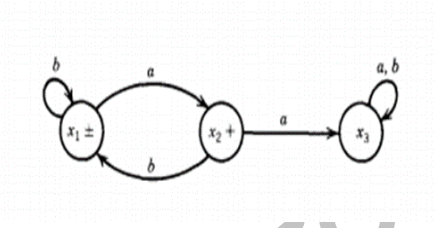
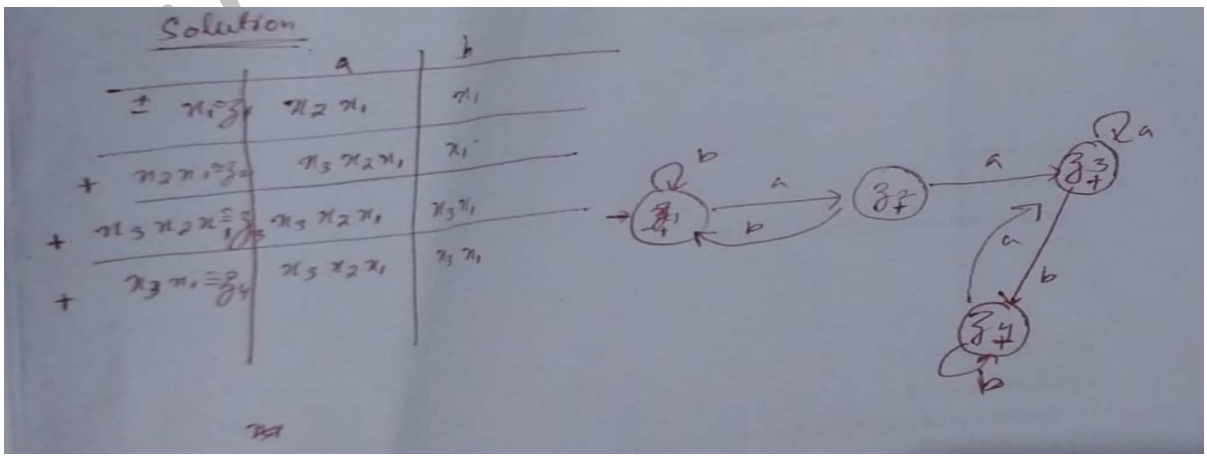


Figure 1

Solution



Question 1b: **20 mins, 5 Points**

Find the union and Intersection of FA2 and FA3 given in Figure 2 and Figure 3.

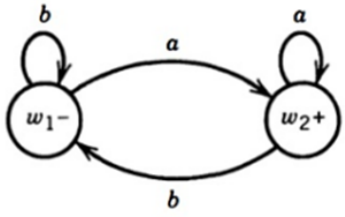


Figure 2

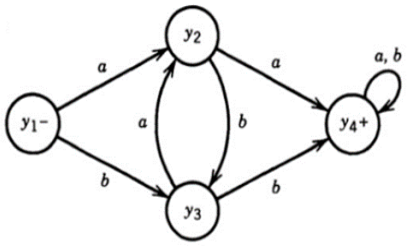
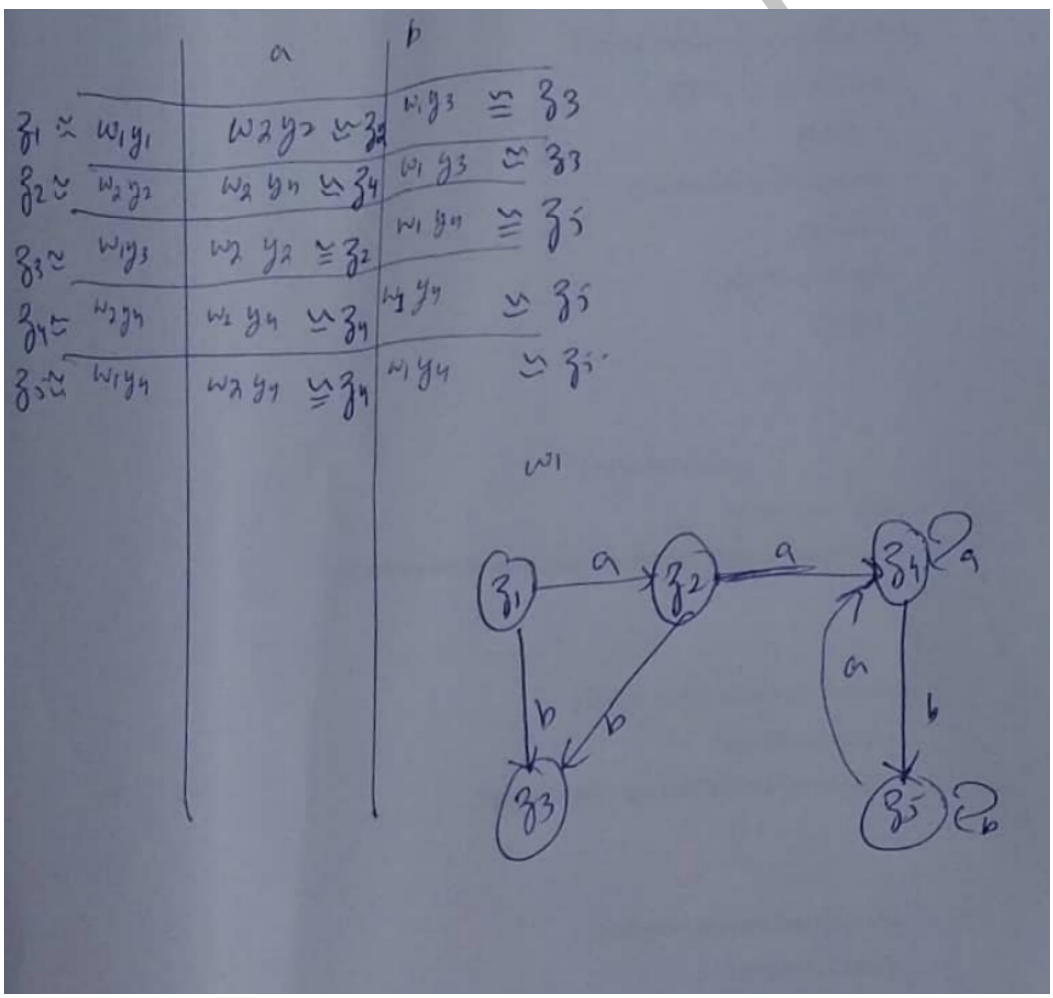


Figure 3

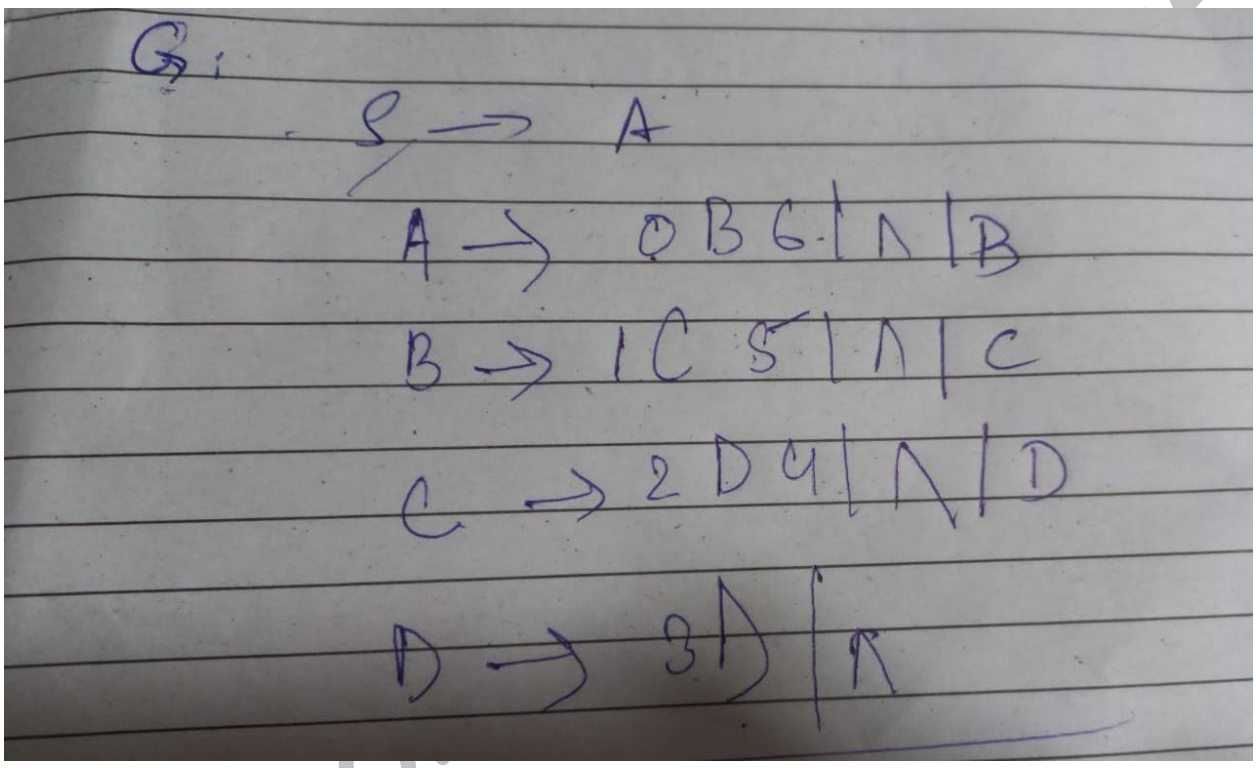
Solutin



Context Free Grammars: **[CLO-3,+ Points,35 mins]**

Question-2a **15 mins, Points**

1. L1 = {$a^n b^m c^k g^q d^p e^r f^s$ | n = s, m = r, k = p}



2. L2 = {w_1 c $a^n b^m a^i b^j w_2$ |w_1 w_2 ε {a,b}* ; lw_1|=|w_2|,j=2i,n ≤ m}

Solution

5. $L_5 = \{ w_1 c a^n b^m a^i b^j w_2 \mid w_1, w_2 \in \{a, b\}^*, \text{length}(w_1)=\text{length}(w_2), j = 2i, n \leq m \}$

Production Rules

$S \rightarrow aSb \mid aSa \mid bSa \mid bSb \mid S_1$
$S_1 \rightarrow cS_2S_3$
$S_2 \rightarrow aS_2b \mid S_2b \mid \epsilon$
$S_3 \rightarrow aS_3bb \mid \epsilon$

Question-2b **CFG to CNF** **10 mins, Points**

Simplify and convert the following productions to CNF

S⟶ TU|V

T⟶aTb| Ɛ

U⟶ cU|Ɛ

V⟶aVc| W

W⟶bW I Ɛ

$$S \to TU \mid V$$
$$T \to aTb \mid \Lambda$$
$$U \to cU \mid \Lambda$$
$$V \to aVc \mid W$$
$$W \to bW \mid \Lambda$$

which can be seen to generate the language $\{a^i b^j c^k \mid i = j \text{ or } i = k\}$.

1. (Identifying nullable variables) The variables T, U, and W are nullable because they are involved in Λ-productions; V is nullable because of the production $V \to W$; and S is also, either because of the production $S \to TU$ or because of $S \to V$. So all the variables are!
2. (Eliminating Λ-productions) Before the Λ-productions are eliminated, the following productions are added:

$$S \to T \quad S \to U \quad T \to ab \quad U \to c \quad V \to ac \quad W \to b$$

After eliminating Λ-productions, we are left with

$$S \to TU \mid T \mid U \mid V \quad T \to aTb \mid ab \quad U \to cU \mid c$$
$$V \to aVc \mid ac \mid W \quad W \to bW \mid b$$

3. (Identifying A-derivable variables, for each A) The S-derivable variables obviously include T, U, and V, and they also include W because of the production $V \to W$. The V-derivable variable is W.
4. (Eliminating unit productions) We add the productions

$$S \to aTb \mid ab \mid cU \mid c \mid aVc \mid ac \mid bW \mid b \qquad V \to bW \mid b$$

before eliminating unit productions. At this stage, we have

$$\begin{aligned}
S &\to TU \mid aTb \mid ab \mid cU \mid c \mid aVc \mid ac \mid bW \mid b \\
T &\to aTb \mid ab \\
U &\to cU \mid c \\
V &\to aVc \mid ac \mid bW \mid b \\
W &\to bW \mid b
\end{aligned}$$

5. (Converting to Chomsky normal form) We replace a, b, and c by X_a, X_b, and X_c, respectively, in productions whose right sides are not single terminals, obtaining

$$\begin{aligned}
S &\to TU \mid X_aTX_b \mid X_aX_b \mid X_cU \mid c \mid X_aVX_c \mid X_aX_c \mid X_bW \mid b \\
T &\to X_aTX_b \mid X_aX_b \\
U &\to X_cU \mid c \\
V &\to X_aVX_c \mid X_aX_c \mid X_bW \mid b \\
W &\to X_bW \mid b
\end{aligned}$$

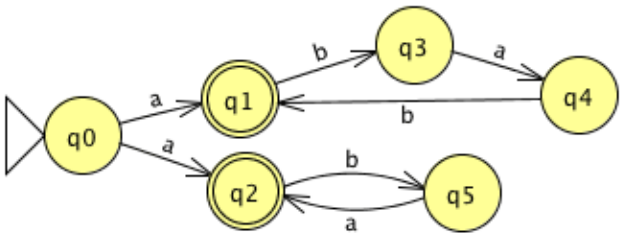
This grammar fails to be in Chomsky normal form only because of the productions $S \to X_aTX_b$, $S \to X_aVX_c$, $T \to X_aTX_b$, and $V \to X_aVX_c$. When we take care of these as described above, we obtain the final CFG G_1 with productions

$$\begin{aligned}
S &\to TU \mid X_aY_1 \mid X_aX_b \mid X_cU \mid c \mid X_aY_2 \mid X_aX_c \mid X_bW \mid b \\
Y_1 &\to TX_b \\
Y_2 &\to VX_c \\
T &\to X_aY_3 \mid X_aX_b \\
Y_3 &\to TX_b \\
U &\to X_cU \mid c \\
V &\to X_aY_4 \mid X_aX_c \mid X_bW \mid b \\
Y_4 &\to VX_c \\
W &\to X_bW \mid b
\end{aligned}$$

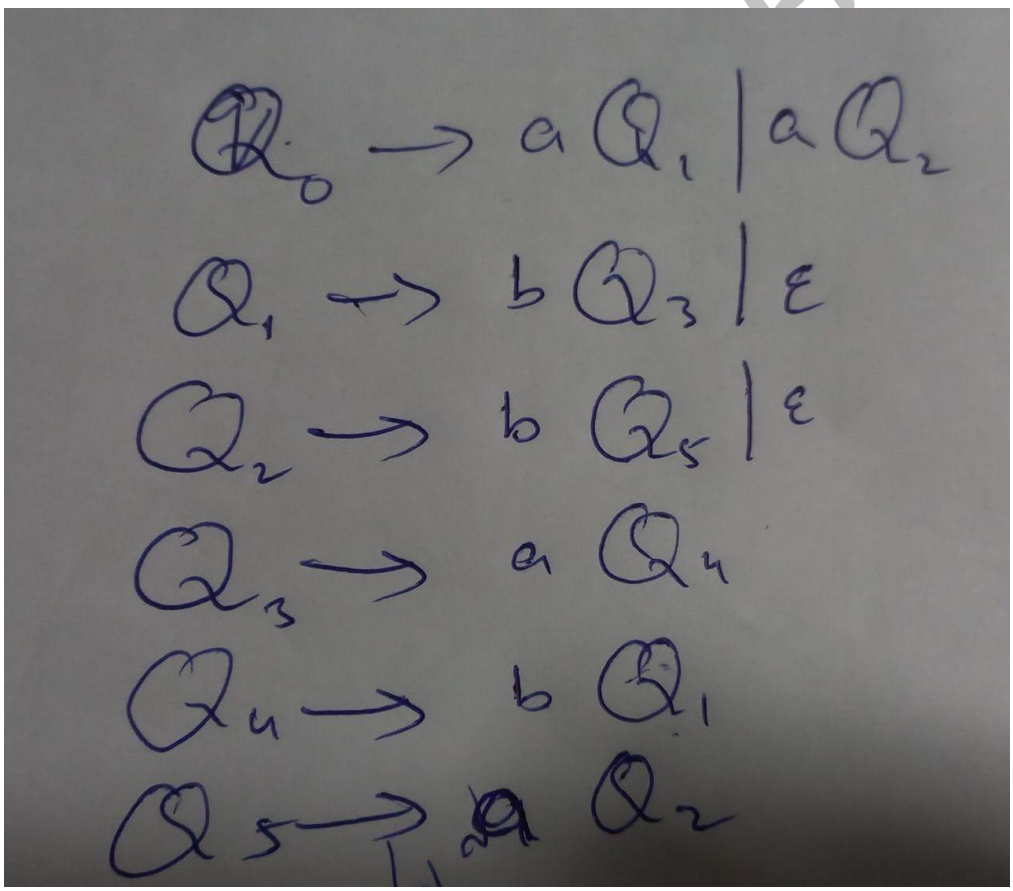
(We obviously don't need both Y_1 and Y_3, and we don't need both Y_2 and Y_4, so we could simplify G_1 slightly.)

Question-2c **10 mins, Points**

2. NFA to CFG



Find a CFG producing the language accepted by the given NFA



Pumping Lemma for CFL and CYK Algorithm **[CLO-5, 10+5 Points, 30 mins]**

Question-3a **10 mins, 5 Points**

Let A ={w ε a,b,c,d}* | $n_a(w) = n_b(w) = n_c(w) = n_d(w)$ }. (i.e. w has the same number of a's and w has . he same number of c's as d's).

Suppose you are trying to prove that A is not context-free using the pumping lemma for context-free languages. Your proof starts (correctly) like this: Suppose for a contradiction that A is context-free. Let p be the pumping length given by the pumping lemma for context-free languages.

Now you have to choose string s. For each choice of s below, state whether or not this choice of s can be used to finish the proof that A is not context-free. If you answer that s cannot be used, you should also briefly and clearly explain why it cannot be used. If you answer that s can be used, complete the proof.

I. $s = a^p b^p c^p d^p$. Can this s be used? yes/no

II. $s = a^p c^{p\,bp} d^p$. Can this s be used? yes/no.

III. $s = d^p c^p b^p a^p$. Can this s be used? yes/no.

IV. $a^p b^p$. Can this s be used? yes/no.

Solution :

If a language L is context-free, then there exists some integer $p > 0$ (called a "pumping length"[1]) such that every string s in L that has a length of p or more symbols can be written as

$s = uvwxy$

with substrings u, v, w, x and y, such that

1. $|vwx| \leq p$,
2. $|vx| \geq 1$, and
3. uv^nwx^ny is in L for all $n \geq 0$.

a.
For $s = a^p b^p c^p d^p$
let p be 2, therefore s= aabbccdd
let
u = aa
v = b
w = NIL
x = b
y = ccdd
Therefore we can easily prove that $uv^n wx^n y$ *is not in L*

*We cannot better fit v,w,x since we have a constrain $|vwx| \leq p$

So a is not Context-free

Same applies for b and c

d. this cannot be used

For $s = a^p b^p$
let p be 2, therefore s= aabb
let
u = a
v = a
w = NIL
x = b
y = b
Therefore we clearly say that $uv^n wx^n y$ *is in L*

Therefore $a^p b^p$ might be context free.

I hope you like the explanation provided by me. I left b and c for you, since they are very similar to problem explained in a and I hope that you can solve them yourself with ease.

Question-3b **20 mins, 10 Points**

Check whether a string "cbba" belongs to given Grammar using CYK Algorithm.

S -> AB

A -> CC | a |c

B -> BC |b

C -> CB | BA |c

SOLUTION

{S, A, C}			
{C}	{B}		
{S, C}	∅	{C}	
{A, C}	{B}	{B}	{A}
c	b	b	a

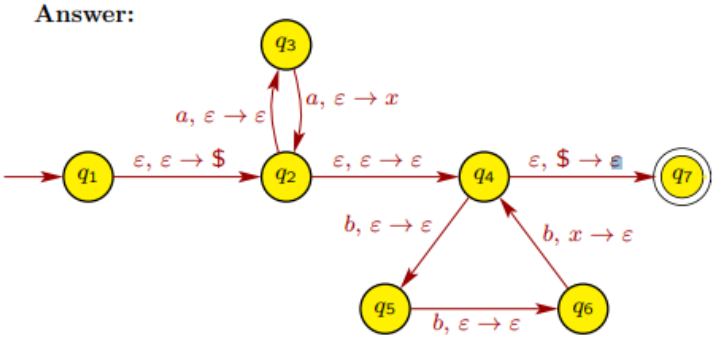
Push Down Automata: Mr. Faisal **[5+5+5] Points, 30 mins]**

Question-4a **25 mins, 5 Points**

Give pushdown automata that recognize the following languages & also show complete computation of any one word (of your choice) of the below languages.You may either use PDA Instantaneous description notation or by drawing stack repeatedly.

F = { $a^{2n} b^{3n}$ | n ≥ 0 }

Answer:



Question-4b **5 mins, Points**

CFG to PDA

Consider the following CFG G = (V, Σ, R, E), where V = {E, T, F}, Σ = {a, b, +, x, (,)}, the start variable is E, and the rules R are

E → E + T|T
T → T × F|F
F → (E)|a

Convert G to an equivalent PDA

Answer:

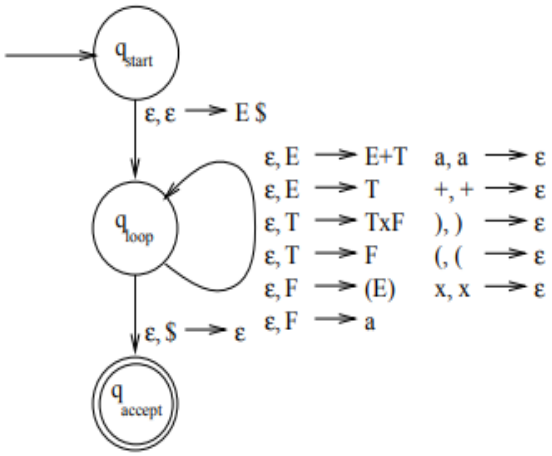
Problem 8 Convert the CFG G_4 given below to an equivalent PDA.

The CFG G_4 is:

$$\begin{aligned} E &\rightarrow E+T|T \\ T &\rightarrow T \times F|F \\ F &\rightarrow (E)|a \end{aligned}$$

Assuming that a shorthand notation allows us to write an entire string to the stack in one PDA step, this task simply reduces to forming transition rules that implement the productions in the grammar.

Here is the PDA:

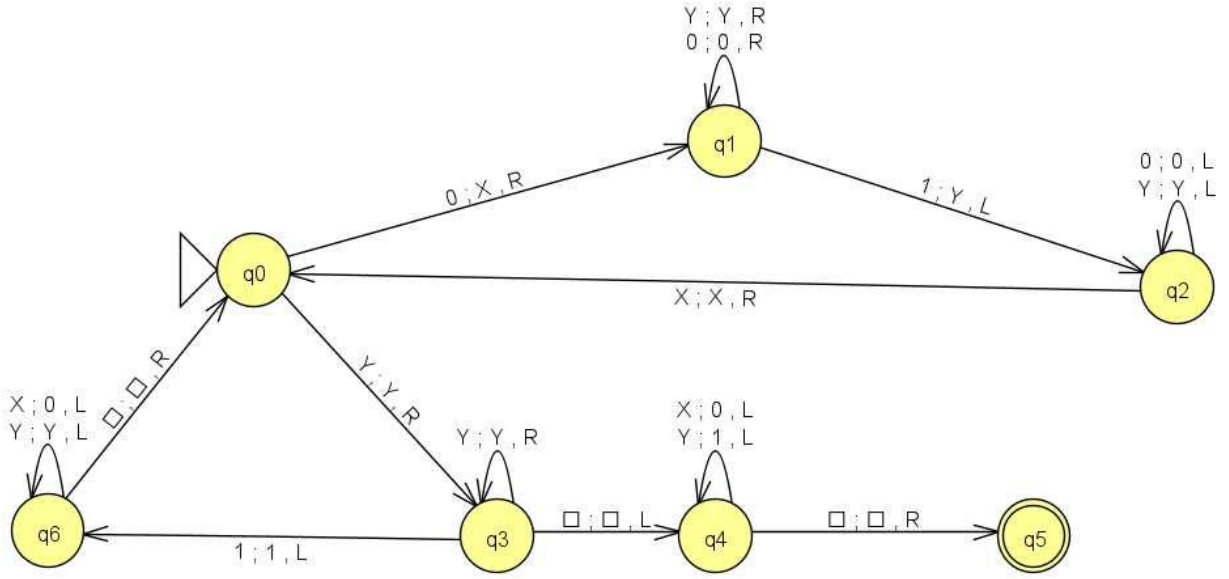


The transitions for the rules of the grammar allow us to nondeterministically replace grammar non-terminals on the stack with their corresponding right-hand-sides; the transitions for the terminals of the grammar $(+, \times,), (, a)$ allow matching of input symbols to grammar terminals. There will be an accepting path through the PDA on string w if and only if w can be generated by the grammar G_4.

Turing Machines:

Question-5a **25 mins, 10+10 Points**

0^n1^m ,m is divisible by m ; n.m>=0



Input	Result
0011	Accept
001111	Accept
0011111111	Accept
00111111111	Reject
000111	Accept
0001111	Reject
000111111	Accept
00011111	Reject
000111111111	Accept
0001111111111111	Accept
00011111111111111	Reject

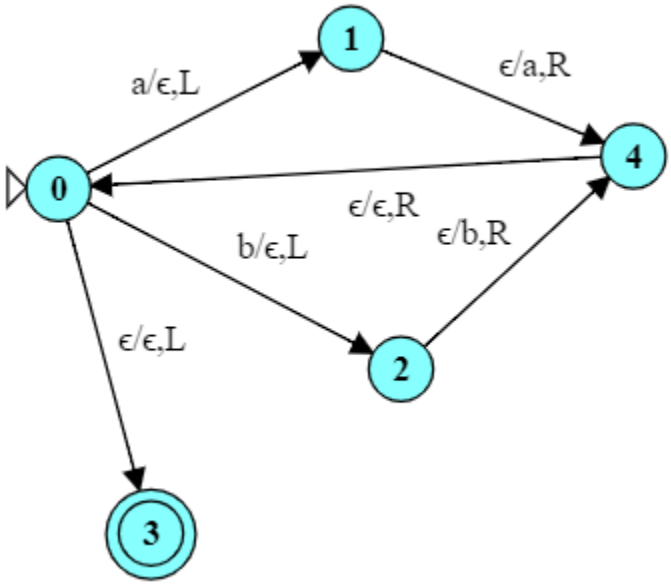
Question-5b **30 mins, Points**

Design a TM that generates **palindromes** over the alphabet {a,b,c}. The machine should take an input string consisting of {a,b,c} characters and it copies the reverse of that string onto its own end. For example, to process ***abc,*** it will produce an output ***abccba***

Solution:

1. Moves to the right end of the string.
2. Changes the final 'c' to 'C': *abC*
3. Moves to the right until it hits an empty cell and writes a 'C' there: *abCC*.
4. Moves to the left until it hits a lower-case letter ('b') and changes that to 'B': *aBCC*
5. Moves to the right until it hits an empty cell and writes a 'B' there: *aBCCB*.
6. Moves to the left until it hits a lower-case letter ('a') and changes that to 'A': *ABCC*
7. Moves to the right until it hits an empty cell and writes a 'B' there: *ABCCBA*.
8. Moves to the left looking for a lower-case letter, but hits the left end of the string instead.
9. Enters state 5, which scans down the string reducing each upper case character to its lower-case equivalent: *abccba*

b2 What is the function computed by following TM?



Solution:

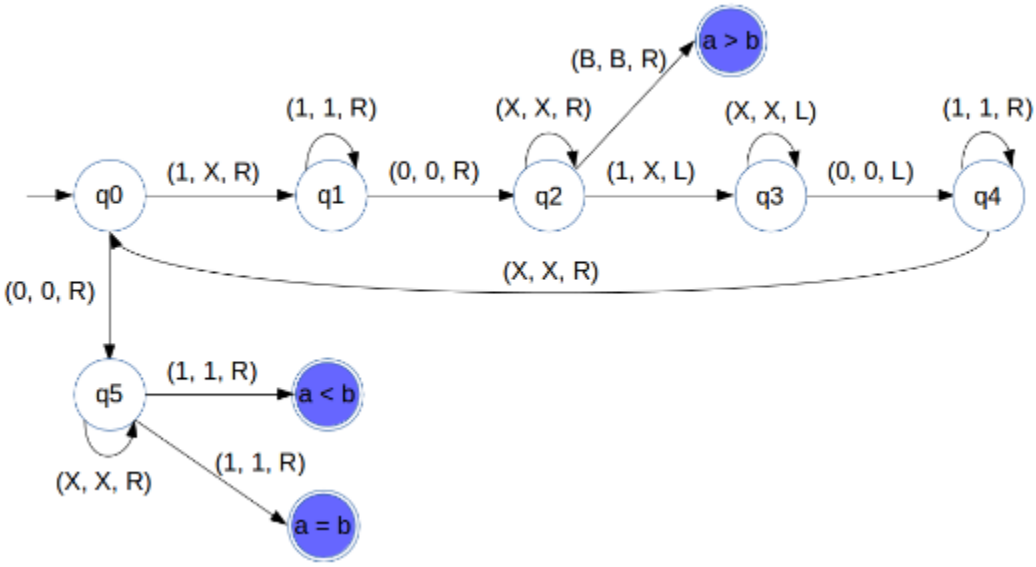
It shifts all of the characters on the right of its starting position one step to the left, overwriting whatever character the head was originally positioned over.

Question-5c **Combining Turing Machine Problem.** **15 mins, Points**

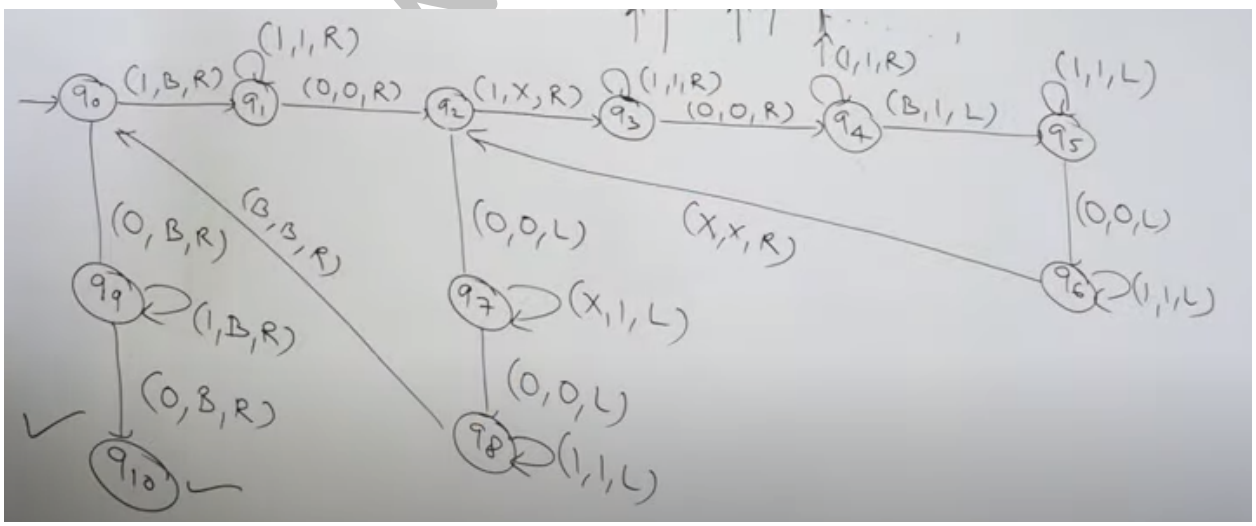
Give pseudocode and its corresponding TM for the following functions:

$$f(a,b) = \begin{cases} a-b, & if\ a > b \\ a \times b, & if\ a \leq b \end{cases}$$

Solution:



Multiplier:



Subtractor: