CS375 Midterm Exam Solution Set (100 points)

Closed book & Closed Notes Oct. 17, 2024

Name _____Sample_____

1. (4 points)

(i) For the following regular expression find a language (i.e., a set of strings) over A = {a, b, c} that can be represented/described by that expression. Put your answer in the following blank. (2 points)

$b^*ac + bc^*$

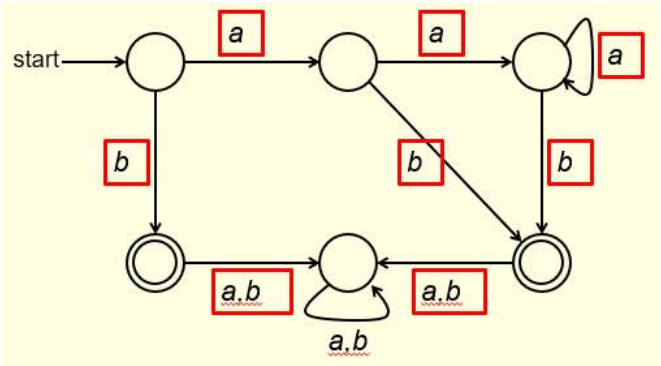
$\{b^m ac \mid m \in N\} \cup \{bc^n \mid n \in N\}$

(ii) If a regular expression for the language over the alphabet {a, b} with no string containing the substring bb is (a+ba)*(Λ+b), then what is the regular expression for the language over the same alphabet with no string containing the substring bbb? Put your answer in the following blank. (2 points)

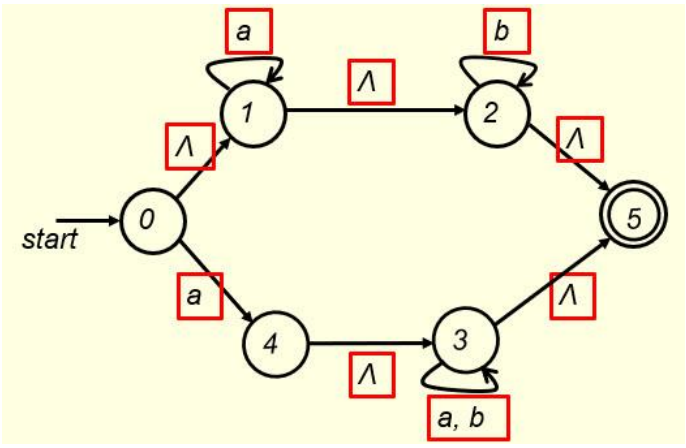
(a + ba + bba)*(Λ + b + bb).

2. (8.5 points)

(i) Fill out the blanks in the following figure to make it a DFA that recognizes the expression b + aa*b. (4 points)

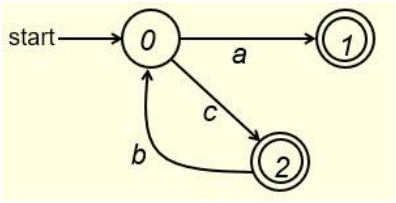


(ii) Fill out the blanks in the following figure to make it an NFA for the expression a*b*+a(a+b)* (4.5 points)

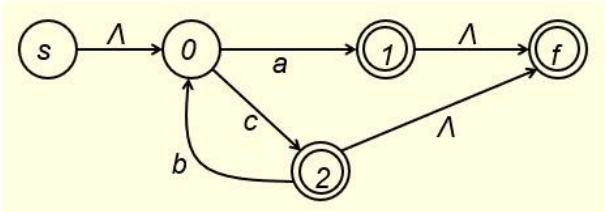


3. (12 points)

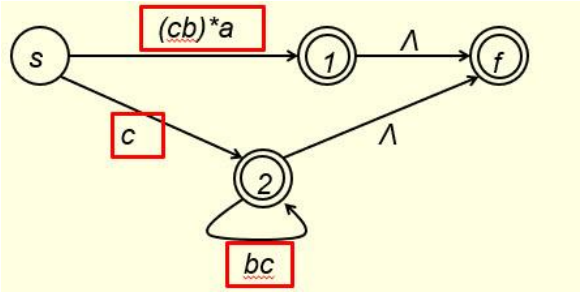
In the process of transforming the following NFA to a regular expression,



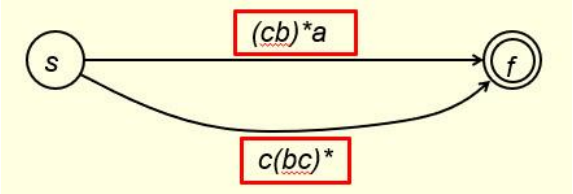
we first connect a new start state s to the start state of the given NFA and connect each final state of the given NFA to a new final state f as shown below.



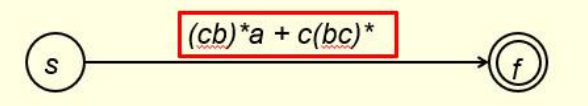
If we eliminate state 0 first, the modified NFA becomes of the following form. Fill out the three blanks in the following figure. (6 points)



If we eliminate state 1 and state 2 then, we get the following NFA. Fill out the two blanks below. (4 points)

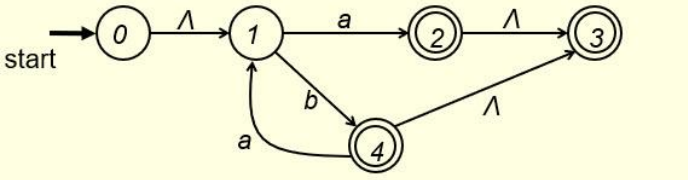


By combining these two edges we get the following NFA. Fill out the blank below (2 points). This is the regular expression of the given NFA.



4. (19 points)

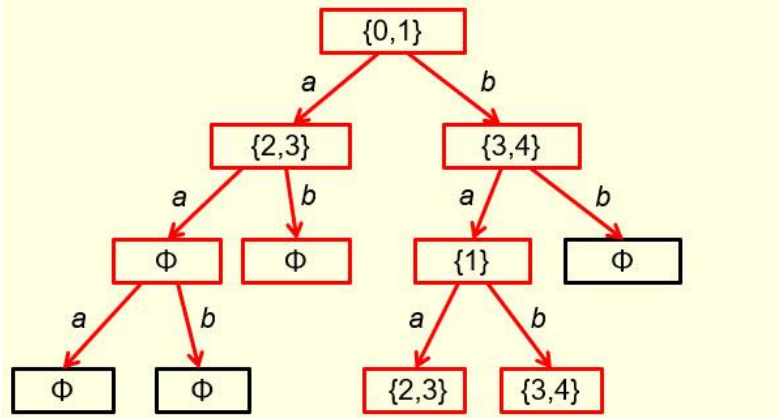
Given the following NFA over the alphabet {a, b},



to transform it to a DFA, first construct Λ-closures of the states of the NFA. Fill out the blanks for state 0, state 1, state 4 and Φ below. (2 points)

Λ(0) = {0, 1}	Λ(1) = {1}
Λ(2) = {2, 3}	Λ(3) = {3}
Λ(4) = {3, 4}	Λ(Φ) = Φ

Next，to get the states of the DFA, we construct a tree as follows. Fill out the blanks in the following tree. (4 points)



The distinct nodes in the above binary tree are listed below (for each level, nodes are taken from the tree from left to right). Fill out the blanks. (2 points)

{0,1}	{2,3}	{3,4}	{1}	Φ

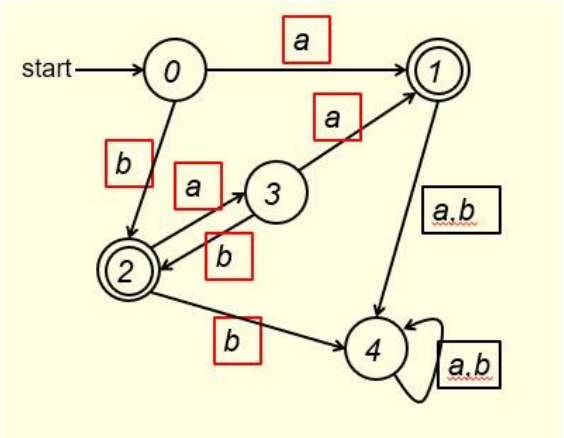
These are the states of the DFA to be constructed and we have the following transition table for the DFA. Fill out the blanks in the table. (5 points)

	T_D	a	b
S	{0,1}	{2,3}	{3,4}
F	{2,3}	Φ	Φ
F	{3,4}	{1}	Φ
	{1}	{2, 3}	{3,4}
	Φ	Φ	Φ

Now replace the five states from top down with 0, 1, 2, 3 and 4, respectively, the transition table of the DFA is of the following form. Fill out the blanks in the table. (3 points)

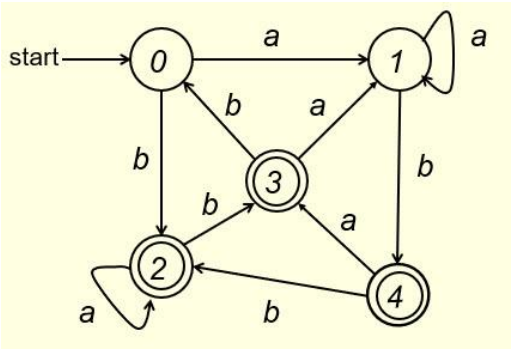
	T_D	a	b
S	0	1	2
F	1	4	4
F	2	3	4
	3	1	2
	4	4	4

Hence, a DFA can be constructed as follows. Fill out the blanks in the DFA. (3 points)

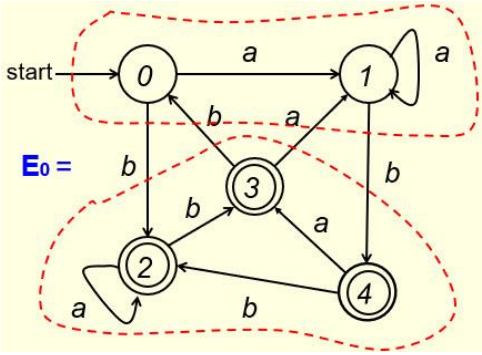


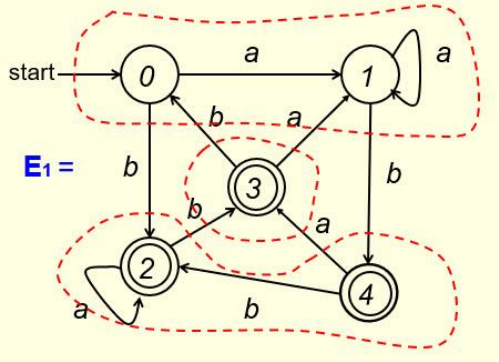
5. (4 points)

(i) Given the following DFA, to find a minimum-state DFA, (2 points)



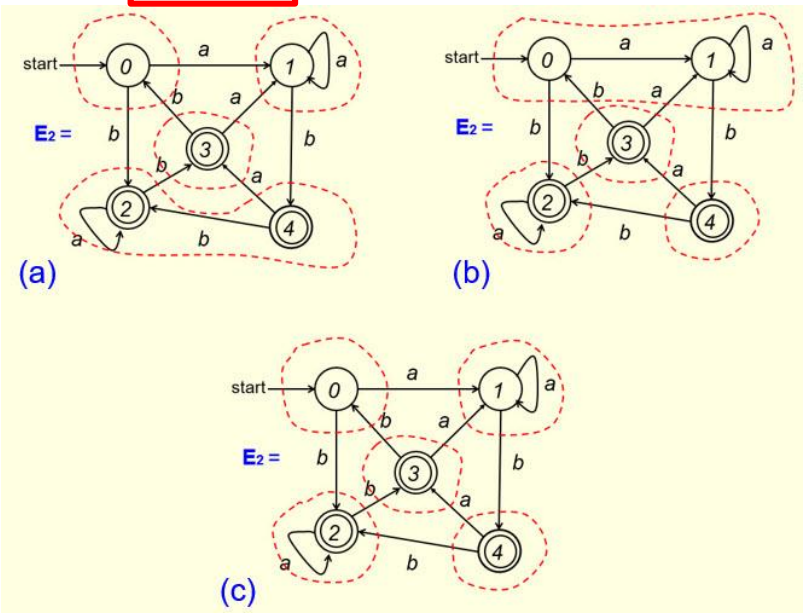
we first construct E_0 as follows and then E_1





Since E_1 is not the same as E_0, so we need to construct E_2. Out of the following three cases (a), (b) and (c), which one represents the correct E_2? Put your answer in the following blank.

The correct E_2 is : (b)



(ii) Let the set of states for a DFA be S = {0, 1, 2, 3, 4, 5}, where the start state is 0 and the final states are 1, 3 and 5. Let the equivalence relation on S for a minimum-state DFA be generated by the following set of equivalent pairs of states:

{ (0, 2), (1, 3) }

The states of the minimum-state DFA are: (2 points)

{0,2}	{1, 3}	{4}	{5}

(set notations)

or

[0]	[1]	[4]	[5]

(equivalence class notations)

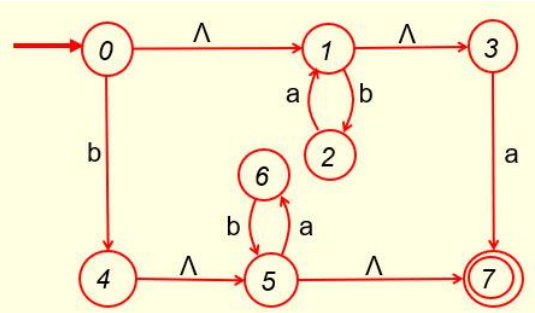
You only need to answer one of the above two cases.

6. (16 points)

Given the following regular expression over the alphabet {a, b},

(ba)*a + b(ab)*

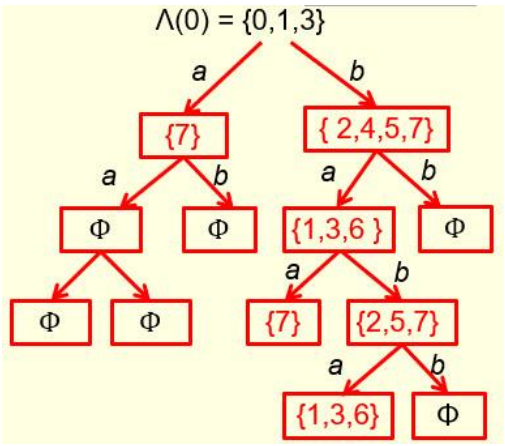
to transform it to a regular grammar, we first transform it to an NFA as the one shown below.



Instead of converting this NFA to a regular grammar directly, we convert it a DFA first and then convert the DFA to a regular grammar (why?). So we construct Λ-closures of the above NFA,

Λ(0) = {0,1,3} Λ(1) = {1,3} Λ(2) = {2} Λ(3) = {3} Λ(4) = {4,5,7}
Λ(5) = {5,7} Λ(6) = {6} Λ(7) = {7} Λ(Φ) = Φ

and build the following tree, (3 points)



so that the following nodes of this tree can be used to build a DFA. Fill out the following blanks and the blanks in the above tree.

{0,1,3}	{7}	{2,4,5,7}	{1,3,6}	{2,5,7}	Φ

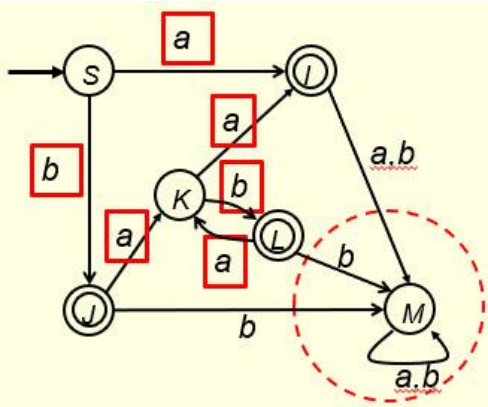
(2 points)

With the selected nodes from the above tree, we build a DFA by constructing the following transition table (the table on the left), and renaming the states as S (start state), I, J, K, L and M: (3.5 points)

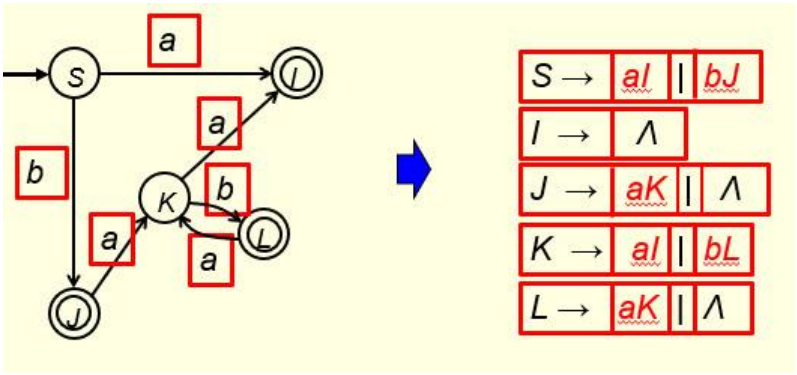
	T	a	b
S	{0,1,3}	{7}	{2,4,5,7}
F	{7}	Φ	Φ
F	{2,4,5,7}	{1,3,6}	Φ
	{1,3,6}	{7}	{2,5,7}
F	{2,5,7}	{1,3,6}	Φ
	Φ	Φ	Φ

	T	a	b
S	S	I	J
F	I	M	M
F	J	K	M
	K	I	L
F	L	K	M
	M	M	M

Then convert the table representation to a digraph representation as follows. Fill out the blanks in the following digraph. (3 points)

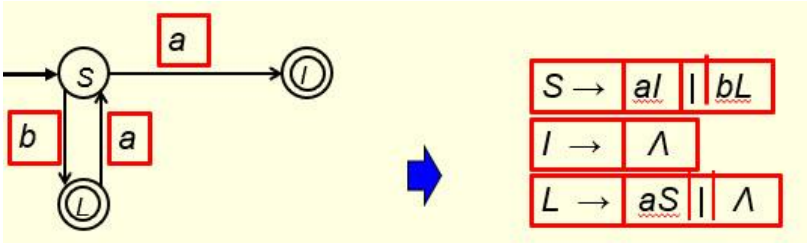


However, we don't need the portion circled in the above digraph representation of the DFA (why?). By removing that portion, the DFA is like the one shown on the left side of the following figure.



By constructing production rules from this FA, we get the production set of the regular grammar on the right side of the above figure. Fill out the blanks in the above figure. (3 points)

The above FA can be further simplified into a form like the one shown below (left hand side of the following figure). (1.5 points)

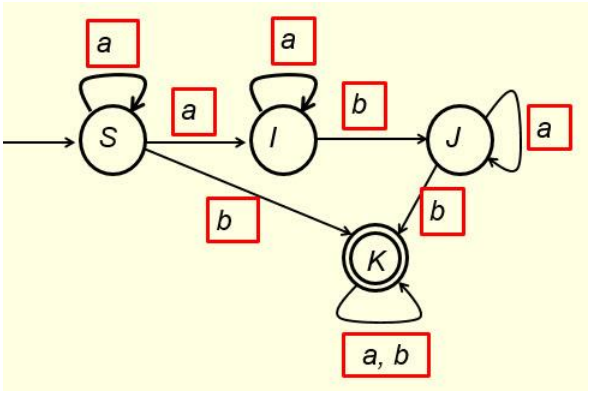


By constructing production rules from this FA, we get the production set on the right side of the above figure (1.5 points). This is the production set of the regular grammar for the regular expression (ba)*a+b(ab)* = (ba)*(a+b).

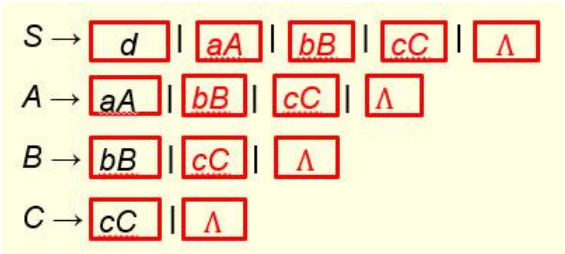
7. Given the following regular grammar:

S → aS | aI | bK, I → bJ | aI, J → bK | aJ, K → aK| bK | Λ

one can use the algorithm given in slide 21 of the notes "Regular Languages and Finite Automata- IV" to construct an NFA to recognize the language of the given grammar. Fill out blanks in the following figure so that the resulting NDA would fulfill such a task, i.e., would recognize the language of the given grammar. (4 points)



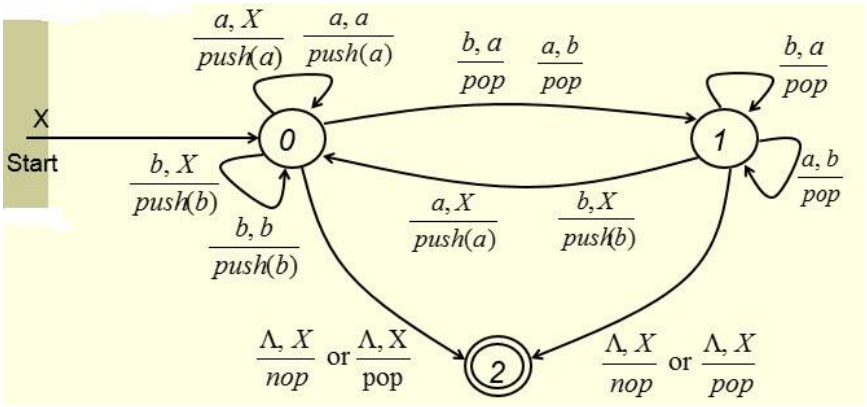
8. Fill out the following blanks to make it a regular grammar for the given regular expression with S being the start symbol: a*b*c* + d (5 points)

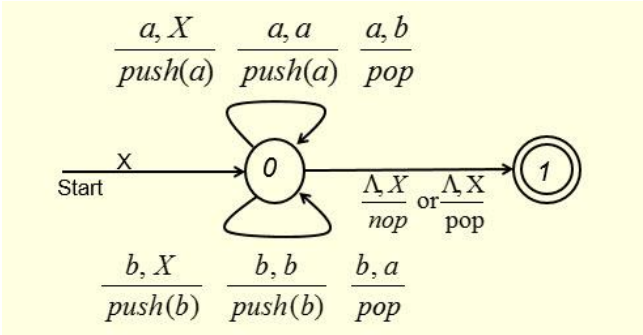


9. Fill out the following blanks to make it a context-free grammar for the given language over the alphabet {a, b}: $\{ a^{2n+3}b^{3n} \mid n >= 0\}$ (2 points)

S → aaSbbb | aaa

10. For the PDA's shown below, which one/ones or no one would accept the language $L = \{x \in \{a,b\}^* \mid N_a(x) = 1 + N_b(x)\}$ by empty stack? Use the execution of the string abaab to justify your answer. (6 points)





Sol.

For the PDA in Figure 1, we have

(0, abaab, X)
→ (0, baab, aX)
→ (1, aab, X)
→ (0, ab, aX)
→ (0, b, aaX)
→ (1, Λ, aX)

We got stuck here. The stack is not empty, so the PDA in Fig.1 does not accept L.

For the PDA in Figure 2, we have

(0, abaab, X)
→ (0, baab, aX)
→ (0, aab, X)
→ (0, ab, aX)
→ (0, b, aaX)
→ (0, Λ, aX)

We got stuck here. The stack is not empty, so the PDA in Fig.2 does not accept L either.

11. (9 points)

The language generated by the following C-F grammar

$$S \rightarrow ABC \qquad A \rightarrow aA \mid \Lambda \qquad B \rightarrow bB \mid \Lambda \qquad C \rightarrow cC \mid \Lambda$$

is $\{a^m b^n c^p \mid m, n, p \in N\}$. To transform the above C-F grammar into a one-state empty-stack PDA so that the accepted language of the PDA is the same as the language generated by this C-F grammar, we need to construct a set of specific instructions for this PDA. Three instructions have already been constructed below. Construct the remaining instructions for this PDA in the following blanks.

(0, a, a, pop, 0)
(0, b, b, pop, 0)
(0, c, c, pop, 0)

(0, d, d, pop, 0)

(0, Λ, S, <pop, push(d)>, 0)

(0, Λ, S, <pop, push(C), push(B), push(A)>, 0)

(0, Λ, A, <pop, push(A), push(a)>, 0)

(0, Λ, A, pop, 0)

(0, Λ, B, <pop, push(B), push(b)>, 0)

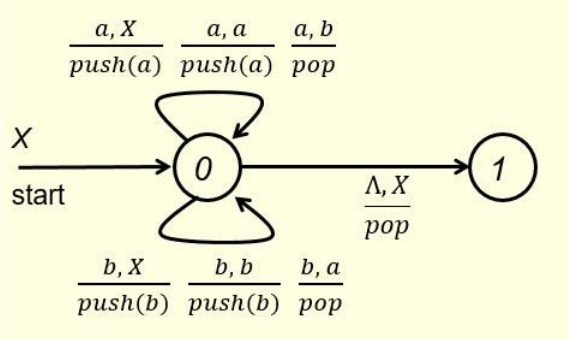
(0, Λ, B, pop, 0)

(0, Λ, C, <pop, push(C), push(c)>, 0)

(0, Λ, C, pop, 0)

12. (11.5 points; 8 points for the first 8 blanks)

Given the following empty-stack PDA,



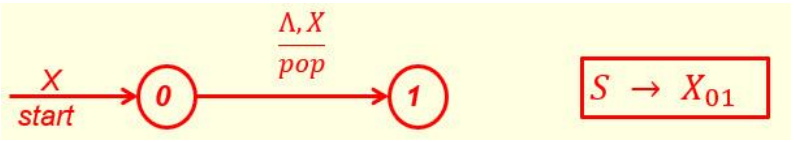
the language *L* accepted by the empty-stack PDA is the set of strings over the alphabet {a, b} such that each string has the same number of a's and b's.

To transform the PDA into a C-F grammar, for each type of the PDA instructions, we need to construct the corresponding grammar productions.

In the following, I list the PDA instructions on the left and you put the corresponding grammar productions in the blanks on the right. We start with Type 4.

Type 4:

The start state and the PDA instruction Λ,X/pop as shown below gives:



Type 1:

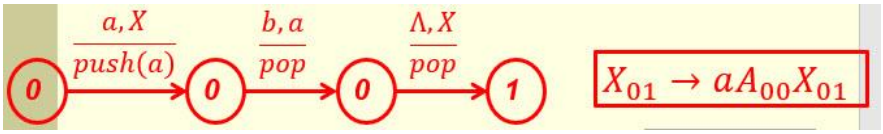
The PDA instruction Λ,X/pop by itself as shown below gives:

The PDA instruction a,b/pop by itself as shown below gives:

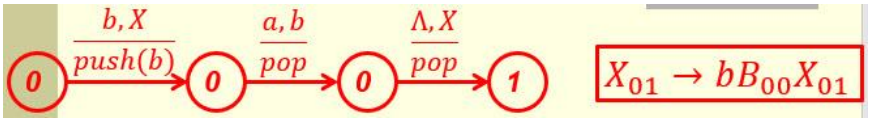
The PDA instruction b,a/pop by itself as shown below gives:

Type 3:

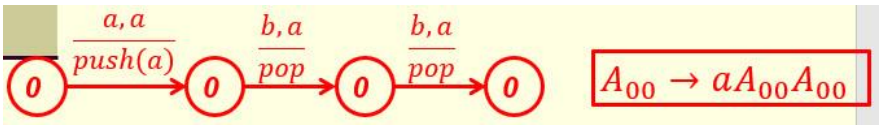
The PDA instruction a,X/push(a) together with the instructions b,a/pop and Λ,X/pop as shown below gives:



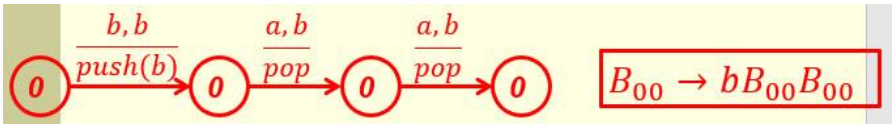
The PDA instruction b,X/push(b) together with the instructions a,b/pop and Λ,X/pop as shown below gives:



The PDA instruction a,a/push(a) together with the instruction b,a/pop (used twice) as shown below gives:



The PDA instruction b,b/push(b) together with the instruction a,b/pop (used twice) as
shown below gives:



There are no Type 2 productions because there are no PDA instructions that perform nop stack operation.

Collecting all the productions, we get a context-free grammar whose production set is as follows:

$S \rightarrow$	X_{01}		
$X_{01} \rightarrow$	Λ	$aA_{00}X_{01}$	$bB_{00}X_{01}$
$A_{00} \rightarrow$	b	$aA_{00}A_{00}$	
$B_{00} \rightarrow$	a	$bB_{00}B_{00}$	

(0 points)

After simplification, we get

$S \rightarrow$	Λ	aAS	bBS
$A \rightarrow$	b	aAA	
$B \rightarrow$	a	bBB	

(3.5 points)

and it is easy to see that this is a C-F grammar for L.