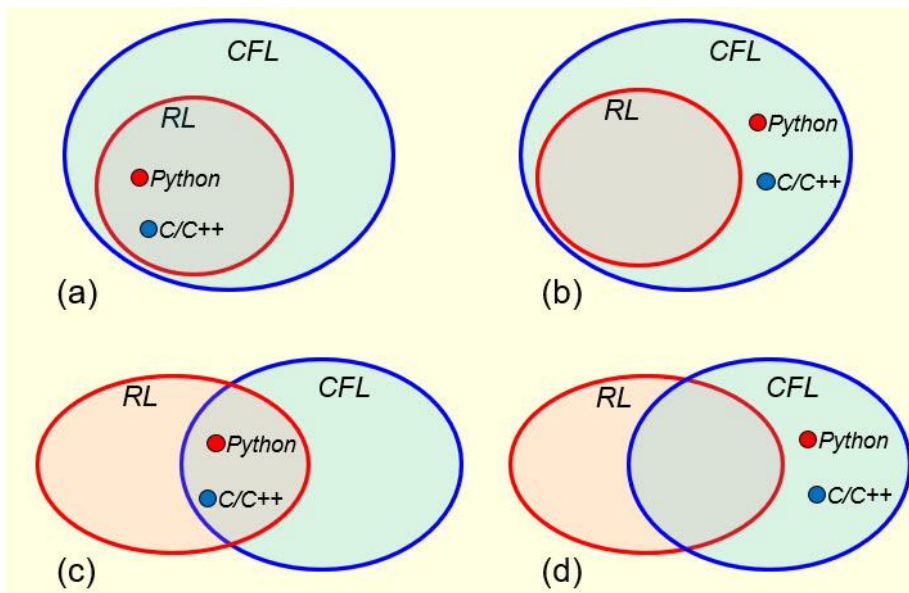


CS375 Homework Assignment 5 Solution Set (40 points)

Due date: 10/16/2025

1. (2 points)

In the following four cases, which one is the correct representation of the relationship between the set of Regular Languages (RL) and the set of Context-Free Languages (CFL), and also the memberships of the Python and C/C++ programming languages with RL and CFL?



Put your answer in the following blank.

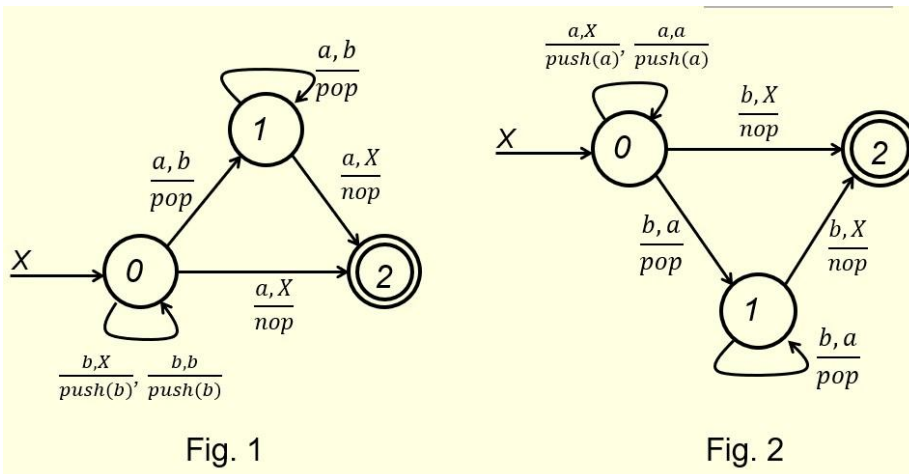
(b)

2. (4 points)

The language accepted by the final-state PDA shown in Fig. 1 below is:

$$\{b^n a^{n+1} \mid n \in \mathbb{N}\}$$

(2 points)



The language accepted by the final-state PDA shown in Fig. 2 is:

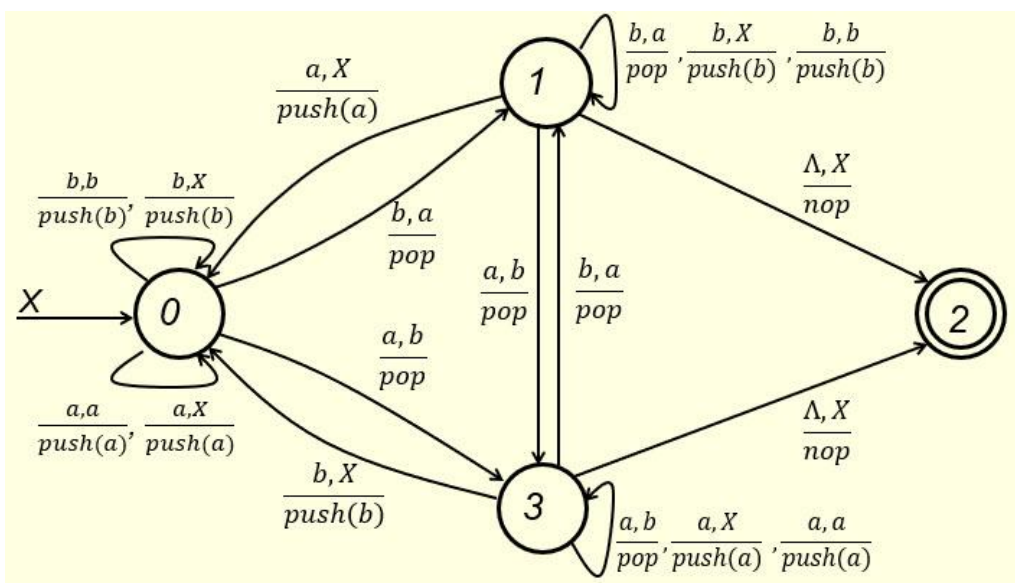
$$\{a^n b^{n+1} \mid n \in N\}$$

(2 points)

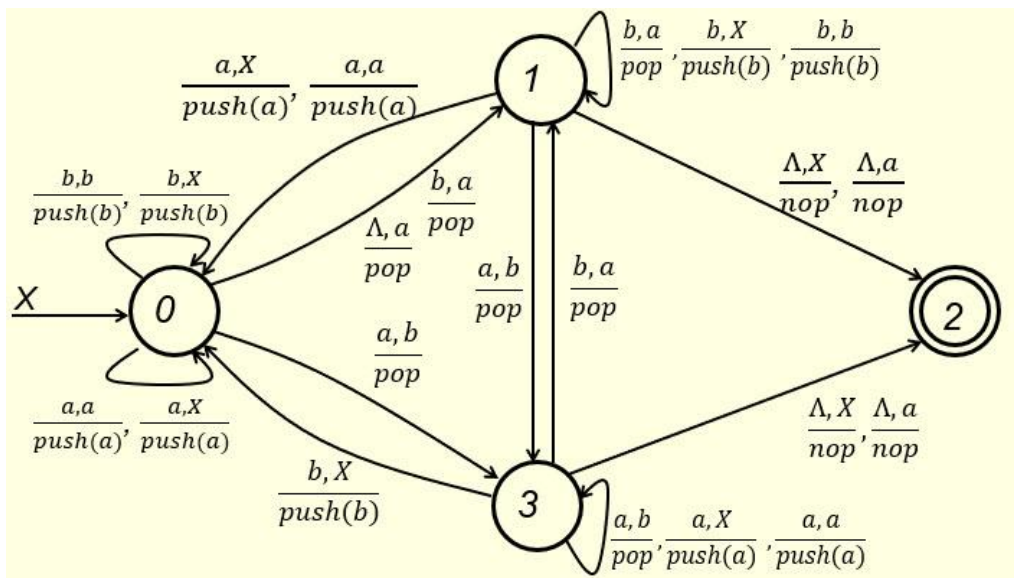
3. (4 points)

If the language accepted by the following final-state PDA is:

$$\{ w \in \{a, b\}^+ \mid N_a(w) = N_b(w) \}$$



then what is the language accepted by the following final-state PDA?

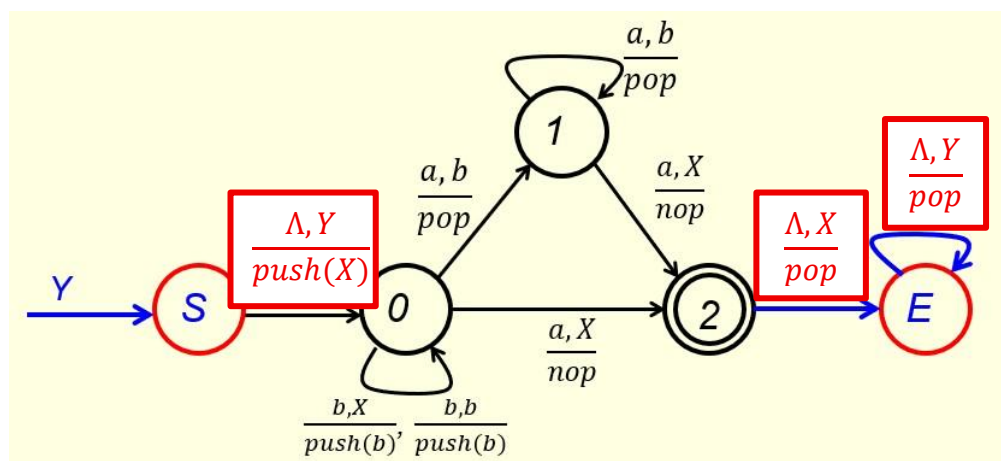


Put your answer in the following text box.

Ignore this question.

4. (6 points)

To convert a final-state PDA to an empty-stack PDA, we put it between a new start state **S** and a new state **E** and define some new edges and new transition instructions for the new edges. For instance, for the final-state PDA shown in Fig. 1 of Question 2, we would get the following empty-stack PDA. Fill out the blanks in the following figure.



5. (8 points)

The language generated by the following C-F grammar

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

is $\{a^k b^m c^n + d \mid k, m, n \in N\}$. To transform the above C-F grammar to 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. Four instructions have already been constructed below. Construct the remaining instructions for this PDA in the following blanks. You might not need all the provided 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(C), push(B), push(A)>, 0)

(0, Λ , C, <pop, push(C), push(c)>, 0)

(0, Λ , C, pop, 0)

(0, Λ , B, <pop, push(B), push(b)>, 0)

(0, Λ , B, pop, 0)

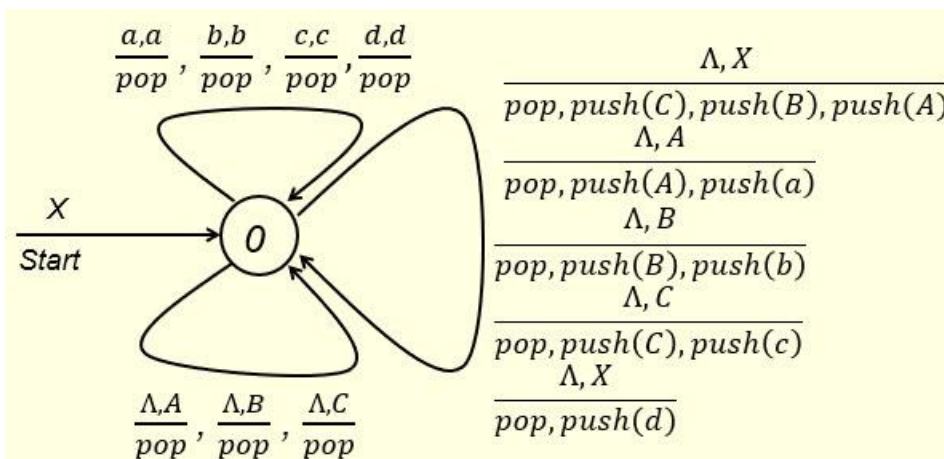
(0, Λ , A, <pop, push(A), push(a)>, 0)

(0, Λ , A, pop, 0)

(0, Λ , S, <pop, push(d)>, 0)

6. (16 points)

Given the following empty-stack PDA,



check if the strings *ba*, *bc*, Λ and *d* would be accepted by this PDA. Show your work in the following three tables, respectively. You need to create/use your own text boxes for your work in the tables.

Note that the key here is, **whenever the top symbol of the stack is a nonterminal**, if the next input symbol to be processed is not a Λ then you put a Λ in front of the un-consumed portion of the input string so you can apply one of the seven instructions of the form “ $\frac{\Lambda, ?}{?}$ ” to replace the top symbol of the stack with a sequence of terminals and/or nonterminals.

(0, <u>ba</u> , X)	(0, a, C)
(0, Λ <u>ba</u> , X)	(0, Λ a, C)
(0, <u>ba</u> , ABC)	(0, a, Φ) or (0, a, <u>cC</u>)
(0, Λ <u>ba</u> , ABC)	<i>Either way, we got stuck here.</i>
(0, <u>ba</u> , BC)	<i>So the string is rejected.</i>
(0, Λ <u>ba</u> , BC)	
(0, <u>ba</u> , <u>bBC</u>)	
(0, a, BC)	
(0, Λ a, BC)	

(0, <u>bc</u> , X)	(0, c, C)
(0, Λ <u>bc</u> , X)	(0, Λ c, C)
(0, <u>bc</u> , ABC)	(0, c, <u>cC</u>)
(0, Λ <u>bc</u> , ABC)	(0, Λ , C)
(0, <u>bc</u> , BC)	(0, Λ , Φ)
(0, Λ <u>bc</u> , BC)	<i>Accepted.</i>
(0, <u>bc</u> , <u>bBC</u>)	
(0, c, BC)	
(0, Λ c, BC)	

$(0, \Lambda, X)$	$(0, d, X)$
$(0, \Lambda, ABC)$	$(0, \Lambda d, X)$
$(0, \Lambda, BC)$	$(0, d, d)$
$(0, \Lambda, C)$	$(0, \Lambda, \Phi)$
$(0, \Lambda, \Phi)$	<i>Accepted</i>
<i>Accepted</i>	