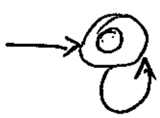
Problem 1: [True of False, with justification] (30 points)
For each of the following questions, state TRUE or FALSE. Justify your answer in brief, indicating only the "proof idea" or counterexample, drawing a diagram if needed.

a) For a regular language L, let $\phi(L)$ be the smallest positive integer for which there exists a DFA with $\phi(L)$ states that recognizes L. If L_1 and L_2 are two regular languages such that $L_1 \subseteq L_2$, then $\phi(L_1) \leq \phi(L_2)$.

FALSE. $L_2 = \Sigma^*$

$L_1 = \Sigma^* \setminus \{\varepsilon\}$



σ for all $\sigma \in \Sigma$

Needs two STATES SINCE INITIAL STATE CANNOT BE A FINAL STATE.

b) Consider the context-free grammar $G = (\{S\}, \{0,1\}, R, S)$ where the rules are given by

$$S \to 0S \mid S1 \mid 0 \mid 1$$

Then $L(G)$ is a regular language.

TRUE. $L(G)$ is THE LANGUAGE $0^* 1^* \setminus \{\varepsilon\}$.

c) For any language L over the alphabet Σ, let $\text{pref}(L)$ be the set of strings that are a prefix of some string in L; more formally

$$\text{pref}(L) = \{w \mid \text{there exists } w' \in \Sigma^\star \text{ such that } ww' \in L\}.$$

If L is a regular language, then so is $\text{pref}(L)$.

TRUE. TAKE DFA FOR L AND EACH

STATE THAT CAN REACH A FINAL STATE

TO THE SET OF FINAL STATES

Problem 2: (15 points)
Let $L = \{0^i 1^j \mid i \text{ and } j \text{ are coprime}\}$. Prove that L is not a regular language.
Recall that two integers a and b are coprime if their greatest common divisor is 1; in other words, there exists no integer greater than 1 that divides both a and b.

Let p be pumping length.
$w = 0^a 1^b$ where $a > p$ is a prime number.

$$b = \frac{(2a-1)!}{a}$$

Note that $\gcd(a,b) = 1$ since a is prime.

Write $w = xyz$ with $|y| > 0$ and $|xy| \leq p$. So $y = 0^l$ for some l

then $xy^i z = 0^{a+(i-1)l} 1^b$

take $i = 2$

$$xy^2z = 0^{a+l} 1^b$$

By pumping lemma, $xy^2z \in L$.

$$\text{So } \gcd(a+l,\ b) = 1.$$

But $a + l > a$

and $a + l \leq a + p < 2a$

But $b = \dfrac{(2a-1)!}{a}$ is a multiple of all integers in $(a, 2a)$,

So L cannot be regular!

Problem 3: (15 points)

a) Let L_1 be a context-free language and L_2 be a regular language. Show that $L_1 \cap L_2$ is a context-free language.

$M_1 =$ PDA FOR L_1 $= (Q_1, \Sigma_1, \Gamma_1, \delta_1, q_0^{(1)}, F_1)$

$M_2 =$ DFA FOR L_2 $= (Q_2, \Sigma_2, \delta_2, q_0^{(2)}, F_2)$

CREATE PDA THAT SIMULATES M_1 AND M_2 SIMULTANEOUSLY.

MORE FORMALLY, CREATE PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$

$Q = Q_1 \times Q_2$, $q_0 = (q_0^{(1)}, q_0^{(2)})$, $F = \{(q', q'') \mid q' \in F_1 \text{ and } q'' \in F_2\}$

$\Sigma = \Sigma_1 \cap \Sigma_2$

$(q', q'', c) \in \delta((\tilde{q}', \tilde{q}''), a, b)$

if $\delta_1(\tilde{q}', a) \ni (q', c)$ and $\delta_2(\tilde{q}'', a) = q''$.

b) Use part (a) to show that the language $L \subseteq \{0, 1, 2\}^*$ given by the set of strings containing the same numbers of 0's, 1's and 2's is not context-free.
(In order to prove that L is not context-free, you may use, without proof, the languages that were proved not to be context-free in class, in the homeworks or in the discussion sections.)

Let $L' = 0^* 1^* 2^*$ which is regular

then if L were CFL, then $L \cap 0^* 1^* 2^* = \{0^i 1^i 2^i \mid i \geq 0\}$ would be CFL. But we saw in class that $\{0^i 1^i 2^i \mid i \geq 0\}$ is not CFL. So L is NOT CFL.

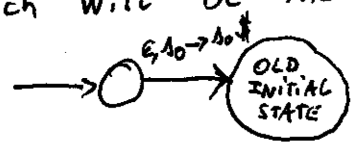
Problem 4: (30 points)
Recall that a pushdown automaton is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$. We then define a *deterministic PDA* as a 7-tuple $(Q, \Sigma, \Gamma, \delta, s_0, q_0, F)$, where $s_0 \in \Gamma$ is the initial symbol in the stack. Moreover, the transition function $\delta\colon Q \times \Sigma_\epsilon \times \Gamma \to Q \times \Gamma^*$ must satisfy the following two criteria:

1. For all $q \in Q$, $a \in \Sigma_\epsilon$ and $s \in \Gamma$, $\delta(q, a, s)$ is a set with at most one element.
2. For all $q \in Q$ and $s \in \Gamma$, if $\delta(q, \epsilon, s) \neq \emptyset$, then $\delta(q, a, s) = \emptyset$ for all $a \in \Sigma$; in other words, if the automaton is at state q with s at the top of the stack, it cannot choose between reading the next input symbol or doing an ϵ-transition.

We say that a language is *deterministic context-free* if it is recognized by a deterministic PDA.
Note that a deterministic PDA must pop the top symbol of the stack at *every* step, and recall that, for non-deterministic pushdown automata, acceptance by *empty stack* is equivalent to acceptance by *final state.*

a) Show that a deterministic PDA that accepts by empty stack can be transformed into a deterministic PDA that accepts by final state.

Add a new STATE which will BE THE FINAL STATE.
Add new initial STATE



, where $ is a new marker symbol added to Γ.
then, from each STATE, Add a Transition "$\epsilon, \$ \to \epsilon$" To the final STATE. this TRANSITION is ONLY USE when the STACK WERE EMPTY IN THE ORIGINAL PDA. Also, IT RESPECTS THE FACT THAT THE NEW PDA IS DETERMINISTIC.

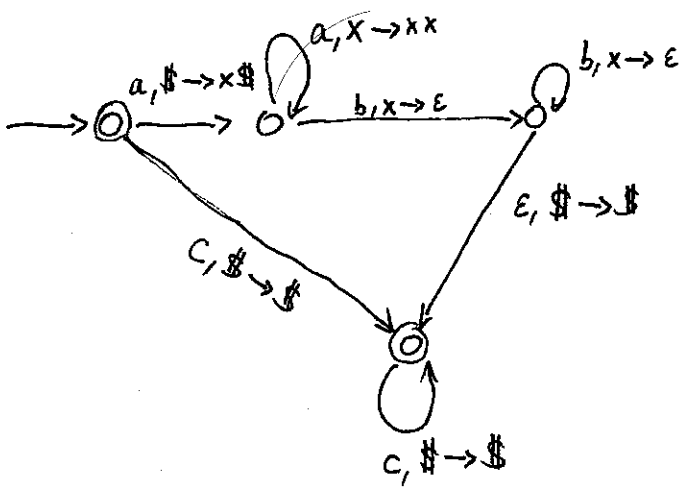
b) Is the language 00*1* accepted by a deterministic PDA via empty stack? Justify your answer.

No. SINCE 0 is ACCEPTED BY PDA, AFTER READING 0, the STACK is empty and THE PDA CANNOT PROCEED. So 00 is NOT ACCEPTED, FOR example.

c) In this question you will show that the class of deterministic context-free languages is not closed under union. Consider the language $L = \{a^i b^j c^k \mid i = j \text{ or } j = k\}$, which is not deterministic context-free. Show that L can be written as $L_1 \cup L_2$, where L_1 and L_2 are two deterministic context-free languages. You must show that your choices of L_1 and L_2 are deterministic context-free.

$$L_1 = \{a^i b^j c^k \mid i = j \text{ and } i,j,k \geq 0\}$$

$$L_2 = \{a^i b^j c^k \mid j = k \text{ and } i,j,k \geq 0\}$$



DPDA for L_1 that Accepts by final STATE.

$ is the first symbol on the STACK

Problem 5: (10 points)

A grammar is called *linear* if all its rules have at most one variable on the right hand side. In other words, if V is the set of variables and Σ is the set of terminals, then all rules of a linear grammar are of the form

$$A \to \alpha B \beta \quad \text{or} \quad A \to \gamma, \quad \text{where } A, B \in V \text{ and } \alpha, \beta, \gamma \in \Sigma^\star.$$

If $\alpha = \epsilon$ in the definition above, that is, all rules have the form

$$A \to B\beta \quad \text{or} \quad A \to \gamma \quad \text{for } A, B \in V \text{ and } \beta, \gamma \in \Sigma^\star,$$

then the grammar is called *left-linear*. Similarly, if $\beta = \epsilon$ in the definition of linear grammars, that is, all rules have the form

$$A \to \alpha B \quad \text{or} \quad A \to \gamma \quad \text{for } A, B \in V \text{ and } \alpha, \gamma \in \Sigma^\star,$$

then the grammar is called *right-linear*.

Give a linear grammar G such that the language $L(G)$ generated by G *cannot* be written as $L(G_1) \cup L(G_2)$ where G_1 is a left-linear grammar and G_2 is a right-linear grammar. Justify your answer.

G:

$S \to 0S1 \mid \varepsilon$

$L(G) = \{0^i 1^i \mid i \geq 0\}$

We saw in class that $L(G)$ is not regular.

But $L(G_1)$ and $L(G_2)$ must be regular, since G_1 is left-linear and G_2 is right-linear. therefore $L(G_1) \cup L(G_2)$ must be ~~[illegible]~~ regular and cannot be equal to $L(G)$.