4. **Problem:** Prove that the class of context-free languages is not closed under intersections and complements. (*Hint:* Represent the language \( \{ a^kb^kc^k \mid k \geq 0 \} \) as the intersection of two context-free languages.)

**Solution:** Let \( L = \{ a^kb^kc^k \mid k \geq 0 \} \). This language has been proven to be not context-free. We can prove that context-free languages are not closed under intersection by finding two context-free languages \( L_1 \) and \( L_2 \) such that \( L = L_1 \cap L_2 \). Languages \( L_1 = \{ a^ib^ic^i \mid i, k \geq 0 \} \) and \( L_2 = \{ a^kb^kc^k \mid i, k \geq 0 \} \) fulfill this condition.

A direct corollary is that the class of context-free languages cannot be closed under complementation, either, since they are closed under union and \( L_1 \cap L_2 = \overline{L_1} \cup \overline{L_2} \).

Finally, we prove that \( L_1 \) and \( L_2 \) are context-free by presenting context-free grammars that generate them. The language \( L_1 \) is generated by \( G_1 = (\{ S, A, B, a, b, c \}, \{ a, b, c \}, P_1, S) \), where \( P_1 = \{ S \to AB, A \to Aa \mid e, B \to bBc \mid e \} \). Similarly, \( L_2 \) is generated by \( G_2 = (\{ S, A, B, a, b, c \}, \{ a, b, c \}, P_2, S) \), where \( P_2 = \{ S \to AB, A \to aAb \mid e, B \to cB \mid e \} \).

5. **Problem:** Show that pushdown automata with two stacks (rather than just one as permitted by the standard definition) would be capable of recognizing exactly the same languages as Turing machines.

**Solution:** We first show that a two-stack pushdown automaton can simulate a Turing machine. The only difficulty is to find a way to simulate the Turing machine tape using two stacks. This can be done using a construction that is similar to the one presented in the first problem: the first stack holds the tape of part that is left to the read/write head (in reversed order), and the second stack holds the symbols that are right to the head.

![Two-stack pushdown automaton simulation](image)

The computation of the automaton can be divided into two parts:

(a) **Initialization,** when the automaton copies the input to stack \( S_1 \) one symbol at a time, and then moves it, again one-by-one, to stack \( S_2 \). (With the exception of the first symbol).

(b) **Simulation,** where the automaton decides its next transition by examining the top symbol of \( S_1 \). If the machine moves its head to left, the top element of \( S_1 \) is moved into \( S_2 \). If it moves to the other direction, the top element of \( S_2 \) is moved to \( S_1 \).

A two-stack pushdown automaton that is formed using these principles simulates a given Turing machine. The formal details are presented in an appendix.

Next we show that we can simulate a two-stack pushdown automaton using a Turing machine. This can be done trivially using a two tape nondeterministic Turing machine where both stacks are stored on their own tapes.
Appendix: the formalisation of solution 5

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ be a Turing machine. We construct a two-stack push-down automaton $M' = (Q', \Sigma', \Gamma', \delta', p_0, q_{acc}, q_{rej})$ as follows:

$Q' = Q \cup \{p_0, p_1, p_2\}$
$\Sigma' = \Sigma \cup \{<\}$
$\Gamma' = \Gamma \cup \{>, <\}$

$\delta' = \{(p_0, \varepsilon, \varepsilon, \varepsilon), (p_1, >, \varepsilon), (p_2, \varepsilon, <)\}$
$\cup \{((p_1, x, \varepsilon, \varepsilon), (p_1, x, \varepsilon)) | x \in \Sigma\}$
$\cup \{((p_2, \varepsilon, x, \varepsilon), (p_2, \varepsilon, x)) | x \in \Sigma\}$
$\cup \{((q_1, \varepsilon, a, \varepsilon), (q_2, \varepsilon, b)) | (q_1, a, q_2, b, L) \in \delta\}$
$\cup \{((q_1, \varepsilon, a, x), (q_2, x, b, \varepsilon)) | (q_1, a, q_2, b, R) \in \delta, x \in \Gamma'\}$