Exercise 1: Counterexample-guided discovery of predicates
Consider the following program.

\[ \varphi_{\text{init}} \equiv pc = \ell_0 \]
\[ \rho_1 \equiv (\text{move}(\ell_0, \ell_1) \land y \geq z \land \text{skip}(x, y, z)) \]
\[ \rho_2 \equiv (\text{move}(\ell_1, \ell_2) \land x + 1 \leq y \land x' = x + 1 \land \text{skip}(y, z)) \]
\[ \rho_3 \equiv (\text{move}(\ell_1, \ell_2) \land x \geq y \land \text{skip}(x, y, z)) \]
\[ \rho_4 \equiv (\text{move}(\ell_2, \ell_3) \land x \geq z \land \text{skip}(x, y, z)) \]
\[ \rho_5 \equiv (\text{move}(\ell_2, \ell_4) \land x + 1 \leq z \land \text{skip}(x, y, z)) \]

Let \( \text{Preds}_{pc} \) be the set of all predicates on the program counter.
\[ \text{Preds}_{pc} = \{ pc = \ell_1, pc = \ell_2, pc = \ell_3, pc = \ell_4, pc = \ell_5 \} \]

(a) Given the path \( \rho_1 \rho_2 \rho_3 \rho_5 \), provide a set of predicates \( \text{Preds} \) such that \( \text{Preds} \cup \text{Preds}_{pc} \) is sufficient to prove safety of the program, i.e., every abstract state returned by \( \text{ABSTREACH}(\text{Preds} \cup \text{Preds}_{pc}) \) is disjoint from \( \varphi_{\text{err}} \) (the set of error states \( \varphi_{\text{err}} \) is \( pc = \ell_4 \)).

Show that the predicates returned by your algorithm are sufficient to prove safety of the program by providing the abstract reachability graph.

(b) On page 31 of the handbook article you can find the (general) function \( \text{REFINEPATH} \) which is used in the function \( \text{ABSTREFINELOOP} \) and returns a set of predicates \( \text{Preds} \) given a path \( \rho_1, \ldots, \rho_n \).

Let us implement \( \text{REFINEPATH} \) using the following idea:

Let \( \varphi_0 := \varphi_{\text{init}} \) and for the other predicates use the result from the application of \( \text{post} \) (e.g., \( \varphi_1 := \text{post}(\varphi_0, \rho_1) \)).

Observe that the predicates satisfy the required constraints of \( \text{REFINEPATH} \).

Compute \( \text{Preds} \) for the above program and path \( \rho_1 \rho_2 \rho_3 \rho_5 \) using this algorithm. Are the predicates sufficient to prove safety?

(c) Imagine we had a smart implementation for \( \text{REFINEPATH} \) to find predicates sufficient to prove safety of any safe program. What are the implications? What can you conclude about the existence of such an algorithm?