Exercise 5F-2. VCGen Do-While

We are given:

- A loop of the form `doInv c while b`
- An invariant Inv that holds before each evaluation of the loop guard b
- A postcondition P that must hold after the loop terminates

The backward verification condition is:

$$VC(\texttt{doInv c while b},\ P) = VC(c,\ (\neg b \Rightarrow P) \wedge (b \Rightarrow Inv))$$

Explanation

- The loop body c must ensure that:
  - If b becomes false after executing c, then P must hold.
  - If b remains true after executing c, then the invariant Inv must hold for the next iteration.
- This formula captures both loop continuation and termination conditions.

Question assigned to the following page: 3

Exercise 5F-3. VCGen Mistakes

We are given three alternate while Hoare rules, named **lannister**, **stark**, and **targaryen**. All are sound but incomplete. We analyze two of them below: **lannister** and **targaryen**.

Lannister Rule

$$\frac{\vdash \{X\}\ c\ \{b \Rightarrow X \wedge \neg b \Rightarrow Y\}}{\vdash \{b \Rightarrow X \wedge \neg b \Rightarrow Y\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{Y\}}$$

Targaryen Rule

$$\frac{\vdash \{X\}\ c\ \{X\}}{\vdash \{X\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{X \wedge \neg b\}}$$

We demonstrate that these rules are incomplete by constructing valid Hoare triples that cannot be derived using the respective rules.

Rule	Precondition (A)	Postcondition (B)	Command (c)	Loop
Lannister	`true`	$z = 0$	$z := z - 1$	`while (z > 0) do z := z - 1`
Targaryen	$z \geq 0$	$z = 0$	$z := z - 1$	`while (z > 0) do z := z - 1`

Example 1: Lannister

- Initial state: $\sigma = \{z \mapsto 1\}$
- Final state: $\sigma' = \{z \mapsto 0\}$
- Execution: $\langle \texttt{while}\ z > 0\ \texttt{do}\ z := z - 1, \sigma\rangle \Downarrow \sigma'$
- $\sigma \models A = \texttt{true}, \quad \sigma' \models B = (z = 0)$
- The triple $\{A\}\ c\ \{B\}$ is **valid**, but cannot be proven using the Lannister rule.
- Reason: There is no fixed X that can serve as an invariant such that c satisfies

  $$\{X\}\ c\ \{b \Rightarrow X \wedge \neg b \Rightarrow Y\}$$

  and such that the conclusion leads to $Y = (z = 0)$. The rule lacks support for inductive reasoning needed to express and maintain an appropriate invariant.

Example 2: Targaryen

- Initial state: $\sigma = \{z \mapsto 1\}$
- Final state: $\sigma' = \{z \mapsto 0\}$
- Execution: $\langle \texttt{while}\ z > 0\ \texttt{do}\ z := z - 1, \sigma\rangle \Downarrow \sigma'$
- $\sigma \models A = (z \geq 0), \quad \sigma' \models B = (z = 0)$
- The triple $\{z \geq 0\}\ c\ \{z = 0\}$ is **valid**, but cannot be derived using the Targaryen rule.
- Reason: The Targaryen rule requires that $\{X\}\ c\ \{X\}$ hold. But:

  $$\{z \geq 0\}\ z := z - 1\ \{z \geq 0\}$$

  is **not valid**, because if $z = 0$, then after the assignment $z = -1$ which violates $z \geq 0$. Thus, the premise of the rule fails and it cannot be applied, despite the program being correct.

These examples demonstrate that both the **Lannister** and **Targaryen** rules are incomplete: they cannot prove valid Hoare triples for simple while loops because they either lack invariant preservation (Lannister) or make overly strong assumptions about command preservation (Targaryen).