Tutorial for Program Verification
Exercise Sheet 8

In this exercise sheet, we work with the Hoare proof system for our programming language Boostan.

Submit your solution by uploading it as PDF in ILIAS.

Exercise 1: Program Semantics
3 Points
In the lecture we defined the semantics of Boostan programs by assigning a relation to each statement. Compute this relation for the Boostan program $P_{\text{pow}} = (V, \mu, \mathcal{T})$ with $V = \{e, x, y, z\}$, $\mu(e) = \mu(x) = \mu(y) = \mu(z) = \mathbb{Z}$, and $\mathcal{T}$ a derivation tree for the program code shown below. List all intermediate steps, i.e., state the relation for each sub-statement.

```plaintext
\begin{verbatim}
e := 1;
z := 0;
while (z < y) {
    e := e * x;
z := z + 1;
}
\end{verbatim}
```

Exercise 2: Precondition - Postcondition
4 Points
Consider the following precondition-postcondition pairs. Which of them are satisfied by all program statements $st$ and all formulas $\varphi$?

(a) $\{true\} \ st \ \{\varphi\}$

(b) $\{false\} \ st \ \{\varphi\}$

(c) $\{\varphi\} \ st \ \{true\}$

(d) $\{\varphi\} \ st \ \{false\}$

If a precondition-postcondition is satisfied by all program statements $st$ and all formulas $\varphi$, then explain why. If a precondition-postcondition is not satisfied by some program statement $st$ and some formulas $\varphi$, then give a counterexample.
Exercise 3: Assignment Axiom

Find some program \( P \) whose code is a single assignment statement of the form \( x := \text{expr} \); and some formula \( \varphi \) such that \( P \) does not satisfy the precondition-postcondition pair \( (\{ \varphi \}, \{ \varphi \land x = \text{expr} \}) \).

The motivation of this exercise is the following. In the lecture we have seen the assignment axiom of the Hoare proof system.

\[
(\text{assig}) \quad \{ \varphi[x \mapsto \text{expr}] \} \ x := \text{expr}; \ {\varphi} 
\]

This rule is not very intuitive because the precondition is obtained as a modification of the postcondition. One may wonder if the following proof rule could be an alternative.

\[
(\text{BadAss}) \quad \{ \varphi \} \ x := \text{expr}; \ {\varphi \land x = \text{expr}} 
\]

The result of this exercise should hint that the (BadAss) proof rule cannot be used as an axiom in a proof system whose goal is the derivation of valid Hoare triples.

Exercise 4: Hoare Proof System

Is there a program that can swap the values of two variables without using a temporary variable? In this exercise we will consider such a program and prove that the program indeed has this property.

Consider the Boostan program \( P_{\text{swap}} = (V, \mu, \mathcal{T}) \) with \( V = \{a, b, x, y\} \), \( \mu(a) = \mu(b) = \mu(x) = \mu(y) = \mathbb{Z} \), and \( \mathcal{T} \) a derivation tree for the program code shown below.

\[
\begin{align*}
x &:= x + y; \\
y &:= x - y; \\
x &:= x - y;
\end{align*}
\]

Use the Hoare proof system to show that \( P \) satisfies the precondition-postcondition pair \( (\{ x = a \land y = b \}, \{ x = b \land y = a \}) \).

Exercise 5: Programming in Boogie

Using the Boogie\(^1\) language, implement a procedure with signature

\[
\text{procedure square}(x : \text{int}) \text{ returns } (z : \text{int})
\]

that takes a (mathematical) integer \( x \) and, if it is greater or equal 0, computes and returns the square \( z = x^2 \). The algorithm may only make use of addition and subtraction, but not use multiplication, division or modulo.

You can use the Boogie interpreter Boogaloo\(^2\) to test your program. A user manual is available online\(^3\).

---


\(^2\)http://comcom.csail.mit.edu/comcom/#Boogaloo