Tutorial for Program Verification
Exercise Sheet 18

In this exercise sheet we work with strongest postconditions, weakest preconditions; with abstract reachability graphs, and with programs with assert statements.

Submit your solution by uploading it as PDF in ILIAS.

Exercise 1: Strongest Postcondition and Weakest Precondition 3 Points
Let $S$ and $S'$ be sets of states, and let $st$ be a statement. For each of the following set relations, either prove its correctness or give a counterexample.

(a) $S = wp(S', st) \iff sp(S, st) = S'$

(b) $S \subseteq wp(S', st) \iff sp(S, st) \subseteq S'$

(c) $S \supseteq wp(S', st) \iff sp(S, st) \supseteq S'$

Exercise 2: Abstract Reachability Graph 4 Points
Consider the following Boo program $P$, with precondition $i = j$ and postcondition $x = y$.

```
x := i;
y := j;
while (x != 0) {
  x := x - 1;
y := y - 1;
}
```

(a) Draw an abstract reachability graph for $P$ that is precise for the set of formulas $B = \{i = j, i \neq j, x = i, y = j\}$.

(b) Give a set of formulas $B'$ that is suitable to show correctness of the program, i.e., give a set $B'$ and an abstract reachability graph $(AC, T)$ for $P$ that is precise for $B'$, such that $AC$ contains no configuration $(\ell_6, \{\varphi\})$ with $\{\varphi\} \cap \{\neg(x = y)\} \neq \emptyset$.  

1
Exercise 3: Correctness Definitions  
In the lecture, we have seen how we can specify correctness of a program in terms of precondition-postcondition pairs or in terms of `assert` statements. In this exercise we will see how to relate the two concepts.

Given a program \( P = (V, \mu, T) \) that contains an arbitrary number of `assert` statements, give a construction of a program \( P' \) and a precondition-postcondition pair \((\varphi_{\text{pre}}, \varphi_{\text{post}})\) such that the following holds:

\[
P \text{ satisfies all } \text{assert} \text{ statements} \\
\text{iff} \\
P' \text{ satisfies the precondition-postcondition pair } (\varphi_{\text{pre}}, \varphi_{\text{post}}).
\]

*Hint:* Introduce one or more new program variables. Think about the termination of the program \( P \).