Exercise 1: Weakest precondition for sequential composition

In the lecture we discussed that the weakest precondition of the sequential composition is independent of the way we add parentheses, i.e.,

\[ \text{wp}((C_1; C_2); C_3, \phi) \equiv \text{wp}(C_1; (C_2; C_3), \phi) \]

Use the following program and postcondition to exemplarily show this fact, i.e., compute wp for both interpretations step by step and compare the results.

\[
C_1 : \text{if } x > 0 \text{ then } x := 1 \text{ else } x := 2 \\
C_2 : y := 1 \\
C_3 : x := x + y
\]

\[ \phi : x = 3 \]

Exercise 2: Recursive equation for loop invariants

In this exercise we derive a recursive equation for the loop invariant of a while loop. This equation might be useful to guess inductive loop invariants.

Consider the following equivalence of commands.

\[ \text{while } b \text{ do } C_0 \equiv \text{if } b \text{ then } C_0 ; \text{while } b \text{ do } C_0 \text{ else skip} \]

(a) Use the operational semantics of commands ("\( \sim \)") to show that the preceding equivalence holds, i.e., show that the following equation is valid.

\[ [\text{while } b \text{ do } C_0] = [\text{if } b \text{ then } C_0 ; \text{while } b \text{ do } C_0 \text{ else skip}] \]

(b) Use the weakest precondition wp(\( \cdot, \cdot \)) to state a recursive equation for a loop invariant \( \theta \) of a while loop while \( b \) do \( C_0 \).

Hint: Start computing wp for both sides. Finally, the right-hand side of the equation should be a first-order logic formula that contains \( b, \theta \), and wp(\( C_0, \phi \)) for some suitable first-order logic formula \( \phi \).
Exercise 3: Hoare logic derivation – Multiplication 2 Points

(a) Write down a partial correctness specification (i.e., precondition and postcondition) for a program $C$ that multiplies two integers $m$ and $n$, where $m$ is nonnegative, and stores the result in $r$.

(b) Write down a program $C$ as specified above that only uses addition (but not multiplication). Use the command language introduced in the lecture.

   \textit{Hint:} Using an auxiliary variable may be helpful for the next part of the exercise.

(c) Annotate the while loop of your program with a suitable loop invariant and construct a Hoare logic derivation that proves that your program $C$ fulfills your correctness specification.

Exercise 4: Hoare logic derivation – Factorial function 2 Points

Consider the annotated program $\text{Fact}$ that was presented in the lecture.

\[
\begin{align*}
\{ n \geq 0 \} \\
f &:= 1; \\
i &:= 1; \\
\text{while } i \leq n \do \{ \theta \} \\
&\{ f := f \cdot i; \}
\}
\quad \{ f = \text{fact}(n) \}
\end{align*}
\]

Recall that $\text{fact}(n)$ denotes the factorial function of $n$.

In Figure 1 you find a derivation of the given partial correctness specification in the Hoare calculus and the following loop invariant.

\[
\theta := f = \text{fact}(i - 1) \land 1 \leq i \land i \leq n + 1
\]

Collect all side conditions from the strengthening/weakening rule applications (marked with “s/w”) and show that they are valid (you can skip trivial proofs). Note that one of the proofs requires a case split.
\[
\begin{align*}
\{1 = 1 \land n \geq 0\} f &:= 1 \{f = 1 \land n \geq 0\} \quad \text{asgn} \\
\{n \geq 0\} f &:= 1 \{f = 1 \land n \geq 0\} \quad \text{s/w}
\end{align*}
\]

\[
\begin{align*}
\{f = 1 \land 1 = 1 \land n \geq 0\} i &:= 1 \{f = 1 \land i = 1 \land n \geq 0\} \quad \text{asgn} \\
\{f = 1 \land n \geq 0\} i &:= 1 \{f = 1 \land i = 1 \land n \geq 0\} \quad \text{s/w}
\end{align*}
\]

\[
\begin{align*}
\{f = 1 \land i = 1 \land n \geq 0\} \quad \text{seq} \\
\{n \geq 0\} f &:= 1 \{f = 1 \land i = 1 \land n \geq 0\} \\
\{n \geq 0\} \textbf{Fact} \{f = \text{fact}(n)\} \quad \text{seq}
\end{align*}
\]

Proof tree for (1):

\[
\begin{align*}
\{f = \text{fact}(i + 1 - 1) \land 1 \leq i + 1 \land i + 1 \leq n + 1\} i &:= i + 1 \{\theta\} \quad \text{asgn} \\
\{f = \text{fact}(i) \land 1 \leq i \land i \leq n\} i &:= i + 1 \{\theta\} \quad \text{s/w}
\end{align*}
\]

\[
\begin{align*}
\{\theta \land i \leq n\} f &:= f \cdot i ; i := i + 1 ; \{\theta\} \quad \text{seq} \\
\{f = 1 \land i = 1 \land n \geq 0\} \textbf{while} i \leq n \{\theta\} \{f := f \cdot i ; i := i + 1\} \{\theta \land \neg(i \leq n)\} \quad \text{whl} \\
\{f = 1 \land i = 1 \land n \geq 0\} \textbf{while} i \leq n \{\theta\} \{f := f \cdot i ; i := i + 1\} \{f = \text{fact}(n)\} \quad \text{s/w}
\end{align*}
\]

Proof tree for (2):

\[
\begin{align*}
\{f \cdot i = \text{fact}(i - 1) \cdot i \land 1 \leq i \land i \leq n\} f &:= f \cdot i \{f = \text{fact}(i - 1) \cdot i \land 1 \leq i \land i \leq n\} \quad \text{asgn} \\
\{\theta \land i \leq n\} f &:= f \cdot i \{f = \text{fact}(i) \land 1 \leq i \land i \leq n\} \quad \text{s/w}
\end{align*}
\]

Figure 1: Hoare derivation for \textbf{Fact} function and \(\theta \equiv f = \text{fact}(i - 1) \land 1 \leq i \land i \leq n + 1\). Due to space constraints the proof tree is split into three subtrees and we have not substituted \(\theta\). On the web page you can find a full picture of the proof tree.