1 Correctness

Partial Correctness:
If
1. we start from an initial state (satisfying the precondition “Q”); and
2. the computation terminates
then
we reach a final state (satisfying the postcondition “R”)

Total Correctness = Partial Correctness + Termination

2 Invariants

A predicate $P$ is said to be an invariant if

- if $P$ holds when the loop is started then the validity of $P$ is preserved after each iteration of the loop

The invariant theorem: $P$ is an invariant of

\[
\text{do } B_1 \rightarrow S_1 \\
\quad \_ \rightarrow \_ \\
\quad \_ \rightarrow \_ \\
\text{do } B_n \rightarrow S_n \\
\text{od}
\]

if

$P \land B_i \Rightarrow wp\left(“S_i” , P\right)$
3 Termination and bound function

To prove termination we use a bound function \( t \) with the following properties:

1. “\( t > 0 \)” holds as long as the loop has not terminated (“\( t \leq 0 \)” implies that the loop has terminated)
2. Each iteration of the loop reduces the value of \( t \) with at least one

The two properties imply that the loop is iterated only finitely many times, the loop terminates in a finite amount of time.

4 Proving Loops - The Iterative Command Theorem

To prove the loop:

\[
\begin{align*}
\{Q\} \\
S_{init} \\
\{\text{inv } P\} \\
\{\text{bound } t\} \\
\text{do} & \quad B_1 \rightarrow S_1 \\
& \quad B_2 \rightarrow S_2 \\
& \quad \vdots \\
& \quad B_n \rightarrow S_n \\
\text{od} \\
\{R\}
\end{align*}
\]

prove the following points:

1. \( Q \Rightarrow wp(S_{init}, P) \)
2. \( P \land B_i \Rightarrow wp(S_i, P) \) for all \( i \) such that \( 1 \leq i \leq n \)
3. \( P \land \neg BB \Rightarrow R \)
4. \( P \land BB \Rightarrow t > 0 \)
5. \( P \land B_i \Rightarrow wp("t_1 := t; S_i", t < t_1) \) for all \( i \) such that \( 1 \leq i \leq n \)

5 Example 1

\[
\begin{align*}
\{Q : a \geq 0\} \\
x := a \\
\{\text{inv } P : (x \geq 0)\} \\
\{\text{bound } t : x\} \\
\text{do} & \quad x \geq 1 \rightarrow x := x - 1 \quad \text{od} \\
\{R : x = 0\}
\end{align*}
\]
5.1 Proof

Prove the following points:

1. \((a \geq 0) \Rightarrow wp\("x := a", x \geq 0")\)
2. \(x \geq 0 \land (x \geq 1) \Rightarrow wp\("x := x - 1", x \geq 0")\)
3. \(x \geq 0 \land \neg(x \geq 1) \Rightarrow (x = 0)\)
4. \(x \geq 0 \land (x \geq 1) \Rightarrow t > 0\)
5. \(x \geq 0 \land (x \geq 1) \Rightarrow wp\("t_1 := t; x := x - 1", t < t_1")\)

Proof of 1

Assume \(a \geq 0\)

\[ wp\("x := a", x \geq 0") = \{ \text{Definition of assignment} \} \]
\[ (a \geq 0) = \{ \text{Assumption: "a \geq 0"} \} \]
\[ T \quad \Box \]

Proof of 2

Assume \(x \geq 0\)
Assume \((x \geq 1)\)

\[ wp\("x := x - 1", x \geq 0") = \{ \text{Definition of assignment} \} \]
\[ (x - 1 \geq 0) = \{ \text{Arithmetic} \} \]
\[ (x \geq 1) = \{ \text{Assumption: "x \geq 1"} \} \]
\[ T \quad \Box \]

Proof of 3

Assume \(x \geq 0\)
Assume \(\neg(x \geq 1)\)

\[ x = 0 = \{ \text{Arithmetic} \} \]
\[ x \geq 0 \land x \leq 0 \]
Proof of 4

Assume \( x \geq 0 \)
Assume \( x \geq 1 \)

\[ t > 0 \]
\[ = \text{ } \{ \text{ Definition of } t \} \]
\[ x > 0 \]
\[ = \text{ } \{ \text{ Arithmetic } \} \]
\[ x \geq 1 \]
\[ = \text{ } \{ \text{ Assumption: } "x \geq 1" \} \]
\[ T \]
\[ \square \]

Proof of 5

Assume \( x \geq 0 \)
Assume \( x \geq 1 \)

\[ wp(\text{"}t_1 := t; x := x - 1\text{","} t < t_1) \]
\[ = \text{ } \{ \text{ Definition of } t \} \]
\[ wp(\text{"}t_1 := x; x := x - 1\text{","} x < t_1) \]
\[ = \text{ } \{ \text{ Definition of sequential composition } \} \]
\[ wp(\text{"}t_1 := x\text{","} wp(\text{"}x := x - 1\text{","} x < t_1)) \]
\[ = \text{ } \{ \text{ Definition of assignment } \} \]
\[ wp(\text{"}t_1 := x\text{","} x - 1 < t_1) \]
\[ = \text{ } \{ \text{ Definition of assignment } \} \]
\[ x - 1 < x \]
\[ = \text{ } \{ \text{ Arithmetic } \} \]
\[ T \]
\[ \square \]
6 Example 2

\{ Q : a \geq 0 \land b \geq 0 \}

\{ \text{inv } P : (x \geq 0) \land (z + x \cdot y = a \cdot b) \}

\{ \text{bound } t : x \}

\text{do } x \geq 1 \rightarrow \text{if } \text{odd}(x) \rightarrow z := z + y

\quad \text{fi;}

\quad x, y := x \div 2, 2 \cdot y

\text{od}

\{ R : z = a \cdot b \}

6.1 Proof

Prove the following points:

1. \((a \geq 0) \land (b \geq 0) \rightarrow wp\("z, x, y := 0, a, b", P")

2. \(P \land (x \geq 1) \rightarrow wp\("\text{IF; } x, y := x \div 2, 2 \cdot y", P")

3. \(P \land \neg(x \geq 1) \rightarrow (z = a \cdot b)

4. \(P \land (x \geq 1) \rightarrow t > 0

5. \(P \land (x \geq 1) \rightarrow wp\("t_1 := t; \text{IF; } x, y := x \div 2, 2 \cdot y", t < t_1")

Proof of 1

Assume \(a \geq 0\)

Assume \(b \geq 0\)

\(wp\("z, x, y := 0, a, b", (x \geq 0) \land (z + x \cdot y = a \cdot b)")

\(= \{ \text{Definition of multiple assignment } \}

\quad (a \geq 0) \land (0 + a \cdot b = a \cdot b)

\(= \{ \text{Arithmetic and and-simplification } \}

\quad a \geq 0

\(= \{ \text{Assumption: "a \geq 0" } \}

\quad T \quad \square

Proof of 2

To prove \("P \land (x \geq 1) \rightarrow wp\("\text{IF; } x, y := x \div 2, 2 \cdot y", P")\)
let \("P' = wp\("x, y := x \div 2, 2 \cdot y", P\)\) and prove the following points:

1. \(P \land (x \geq 1) \rightarrow \text{odd}(x) \lor \text{even}(x)

2. \(P \land (x \geq 1) \land \text{odd}(x) \rightarrow wp\("z := z + y", P'\)

3. \(P \land (x \geq 1) \land \text{even}(x) \rightarrow wp\("\text{skip", P'}\)
Proof of 2.1

Assume $x \geq 0$
Assume $z + x \ast y = a \ast b$
Assume $x \geq 1$

$add(x) \lor even(x)$

\[
= \{ \text{Arithmetic} \}
\]

$T \quad \square$

---

Proof of 2.2

Assume $x \geq 0$
Assume $z + x \ast y = a \ast b$
Assume $x \geq 1$
Assume $odd(x)$

$wp\left(\z := z + y\right)$

\[
= \{ \text{Definition of } P' \}
\]

$wp\left(\z := z + y\right)$

$= \{ \text{Definition of } P \}$

$wp\left(\z := z + y, \quad wp\left(\x, y := x \div 2, 2 \ast y\right), \quad P\right)$

\[
= \{ \text{Definition of multiple assignment } \}
\]

$wp\left(\z := z + y, \quad \left(x \div 2 \geq 0 \land z + (x \div 2) \ast 2 \ast y = a \ast b\right)\right)$

\[
= \{ \text{Definition of multiple assignment } \}
\]

$(x \div 2 \geq 0) \land (z + y + (x \div 2) \ast 2 \ast y = a \ast b)$

\[
= \{ \text{Conditional Substitution: } \text{“}odd(x)\text{”} \text{ and arithmetic: } \text{“}odd(x) \Rightarrow (x \div 2) = (x - 1)/2\text{”} \}
\]

$(x - 1)/2 \geq 0) \land (z + y + ((x - 1)/2) \ast 2 \ast y = a \ast b)$

\[
= \{ \text{Arithmetic: } \text{“}x/2 \geq 0 \Rightarrow (x \geq 0)\text{”} \}
\]

$(x - 1) \geq 0) \land (z + y + ((x - 1)/2) \ast 2 \ast y = a \ast b)$

\[
= \{ \text{Arithmetic } \}
\]

$(x \geq 1) \land (z + y + (x - 1) \ast y = a \ast b)$

\[
= \{ \text{Arithmetic } \}
\]

$(x \geq 1) \land (z + x \ast y = a \ast b)$

\[
= \{ \text{Assumptions: } \text{“}x \geq 1\text{”} \text{ and } “z + x \ast y = a \ast b” \}
\]

$T \land T$

\[
= \{ \text{and-simplification } \}
\]

$T \quad \square$
Proof of 2.3

Assume \( x \geq 0 \)
Assume \( z + x \ast y = a \ast b \)
Assume \( x \geq 1 \)
Assume \( \text{even}(x) \)

\[ wp("\text{skip}\), P') \]
\[ = \{ \text{Definition of } P' \} \]
\[ wp("\text{skip}\), wp("x, y := x \div 2, 2 \ast y", P)) \]
\[ = \{ \text{Definition of } P \} \]
\[ wp("\text{skip}\), wp("x, y := x \div 2, 2 \ast y", x \geq 0 \land z + x \ast y = a \ast b)) \]
\[ = \{ \text{Definition of multiple assignment} \} \]
\[ wp("\text{skip}\), z + (x \div 2) \ast 2 \ast y = a \ast b) \]
\[ = \{ \text{Definition of skip} \} \]
\[ x \div 2 \geq 0 \land z + (x \div 2) \ast 2 \ast y = a \ast b \]
\[ = \{ \text{Conditional Substitution: } \text{"even}(x)" \text{ and } \text{"even}(x) \Rightarrow (x \div 2 = x/2)" \} \]
\[ x/2 \geq 0 \land z + (x/2) \ast 2 \ast y = a \ast b \]
\[ = \{ \text{Arithmetic: } 2 \ast (x/2) = x" \} \]
\[ x/2 \geq 0 \land z + x \ast y = a \ast b \]
\[ = \{ \text{Arithmetic: } "(x/2 \geq 0) = (x \geq 0)" \} \]
\[ x \geq 0 \land z + x \ast y = a \ast b \]
\[ = \{ \text{Assumptions: } "x \geq 0" \text{ and } "z + x \ast y = a \ast b" \}
\[ \text{and-simplification} \}
\[ T \quad \square \]

Proof of 3

Assume \( x \geq 0 \)
Assume \( z + x \ast y = a \ast b \)
Assume \( \neg(x \geq 1) \)

\[ z = a \ast b \]
\[ = \{ \text{Assumption: } "z + x \ast y = a \ast b" \} \]
\[ z = z + x \ast y \]
\[ = \{ \text{and-simplification and assumption: } "x \geq 0" \} \]
\[ x \geq 0 \land z = z + x \ast y \]
\[ = \{ \text{Arithmetic} \} \]
\[ (x > 0 \lor x = 0) \land z = z + x \ast y \]

7
\[ (x \geq 1 \lor x = 0) \land z = z + x \ast y \]
\[ = \{ \text{and-simplification and assumption: } \neg(x \geq 1) \} \]
\[ (x \geq 1 \land \neg(x \geq 1)) \lor x = 0 \land z = z + x \ast y \]
\[ = \{ \text{Contradiction and or-simplification} \} \]
\[ x = 0 \land z = z + x \ast y \]
\[ = \{ \text{Substitution} \} \]
\[ x = 0 \land z = z + 0 \ast y \]
\[ = \{ \text{or-simplification and Contradiction} \} \]
\[ (x \geq 1 \land \neg(x \geq 1)) \lor x = 0 \land z = z + 0 \ast y \]
\[ = \{ \text{Assumption: } \neg(x \geq 1) \text{ and and-simplification} \} \]
\[ (x \geq 1 \lor x = 0) \land z = z + 0 \ast y \]
\[ = \{ \text{Arithmetic} \} \]
\[ x > 0 \lor x = 0 \land z = z + 0 \ast y \]
\[ = \{ \text{arithmetric} \} \]
\[ x > 0 \land z = z + 0 \ast y \]
\[ = \{ \text{Assumption: } x \geq 0 \text{ and and-simplification} \} \]
\[ z = z + 0 \ast y \]
\[ = \{ \text{Arithmetic} \} \]
\[ z = z \]
\[ = \{ \text{Identity} \} \]
\[ T \quad \Box \]

**Proof of 4**

**Assume** \( x \geq 0 \)
**Assume** \( z + x \ast y = a \ast b \)
**Assume** \( x \geq 1 \)

\( t > 0 \)
\[ = \{ \text{Definition of } t \} \]
\[ x > 0 \]
\[ = \{ \text{Arithmetic} \} \]
\[ x \geq 1 \]
\[ = \{ \text{Assumption: } x \geq 1 \} \]
\[ T \quad \Box \]
Proof of 5

Assume \( x \geq 0 \)
Assume \( z + x * y = a * b \)
Assume \( x \geq 1 \)

\[
wp ("t_1 := t\; ;\; \text{IF}; \; x, y := x \div 2, 2 * y", \; t < t_1) = \]
\[
\{ \text{Definition of } t \} \]
\[
wp ("t_1 := x\; ;\; \text{IF}; \; x, y := x \div 2, 2 * y", \; x < t_1) = \]
\[
\{ \text{Definition of sequential composition } \}
wp ("t_1 := x", \; wp ("\text{IF}" \; wp ("x, y := x \div 2, 2 * y", \; x < t_1))) = \]
\[
\{ \text{Definition of multiple assignment } \}
wp ("t_1 := x", \; wp ("\text{IF}" \; wp ("x, y := x \div 2, 2 * y", \; x \div 2 < t_1))) = \]
\[
\{ \text{Definition of the alternative command } \}
wp ("t_1 := x", \; \left( \begin{array}{c}
odd(x) \lor even(x) \land \\
odd(x) \Rightarrow wp ("z := z + y", \; x \div 2 < t_1) \land \\
even(x) \Rightarrow wp ("\text{skip}" \; wp ("\text{IF}" \; wp ("x, y := x \div 2, 2 * y", \; x \div 2 < t_1))), \; x \div 2 < t_1)
\end{array} \right) = \]
\[
\{ \text{Definition of assignment } \}
odd(x) \lor even(x) \land 
odd(x) \Rightarrow wp ("z := z + y", \; (x \div 2) < x) \land 
even(x) \Rightarrow wp ("\text{skip}" \; wp ("\text{IF}" \; wp ("x, y := x \div 2, 2 * y", \; (x \div 2) < x))), \; x \div 2 < x = \]
\[
\{ \text{Definition of assignment and Definition of } \text{skip} \}
(odd(x) \lor even(x)) \land (odd(x) \Rightarrow x \div 2 < x) \land 
even(x) \Rightarrow x \div 2 < x = \]
\[
\{ \text{Proof by Cases } \}
(odd(x) \lor even(x)) \land (odd(x) \lor even(x) \Rightarrow x \div 2 < x) = \]
\[
\{ \text{Arithmetic } \}
T \land (T \Rightarrow (x \div 2 < x)) = \]
\[
\{ \text{and-simplification and Left identity of } \Rightarrow \}
x \div 2 < x = \]
\[
\{ \text{Arithmetic: } "0 < x = (x \div 2 < x)" \}
0 < x = \]
\[
\{ \text{Arithmetic } \}
x \geq 1 = \]
\[
\{ \text{Assumption: } "x \geq 1" \}
T \; \Box
**Exercise 1** Remember that $P$ is an invariant of a loop iff

$$P \land B_i \Rightarrow wp(S_i, P)$$

. Are the following conjectures true or false?

- $T$ is an invariant of every loop.
- $F$ is an invariant of every loop.
- Every predicate is an invariant of the loop

$$\text{do } T \rightarrow \text{skip } \text{od}$$

- No predicate is an invariant of the loop

$$\text{do } T \rightarrow \text{abort } \text{od}$$

**Exercise 2** An 8x8 chessboard can be covered completely by 2x1 and 1x2 tiles in many different ways. However, if we remove two opposite corners from the chessboard, can we still cover it completely with tiles? Tiles may not overlap or spill over the edges.

*Hint:* testing different tile configurations is unlikely to help you! Instead, try solving the problem by developing an invariant that every correct tile configuration must satisfy, and then show how to satisfy this invariant or why it can’t be satisfied.

**Exercise 3** In the toy language we use in this course, we allow variables to assume arbitrary values from $\mathbb{Z}$. However, this ignores the problem of integer overflow. On a real computer, integer variables only have a fixed number of bits of storage available. For an example, consider the following addition of 8-bit integers: $1100\ 1100 + 1010\ 1010 = 1\ 0111\ 0110$. The resulting integer would require 9 bits of storage, which we don’t have!

How can we handle this problem when using formal methods? In particular, how would you handle it within the toy language and deduction system used so far in this course?