MVSI – Modelling and Verification of Computer Systems

Annotation, contracts, modelling and verification

Tutorial – with solutions

Dominique Méry – 20th November 2026

✎ Exercise 1

`appex3_1.tla`

Consider the following contract:

variables $int\ X, int\ Y, int\ Z$
requires $P(x_0,y_0,z_0)$
ensures $Q(x_0,y_0,z_0,x_f,y_f,z_f)$
begin
// 0 : $R_0(x_0,y_0,z_0,x,y,z)$
$\mathtt{X} = \mathtt{g}(\mathtt{X},\mathtt{Y},\mathtt{Z})$
// f : $R_f(x_0,y_0,z_0,x,y,z)$
end

- g is an arithmetic function defined on the type of the integers `int`; it leads to a typing predicate $x,y,z \in \mathbb{Z}$ (or $\mathbb{Z}(x,y,z)$).
- $P(x_0,y_0,z_0)$ defines the precondition, that is, the conditions that the initial values of the variables `X,Y,Z` must satisfy.
- $Q(x_0,y_0,z_0,x_f,y_f,z_f)$ defines the postcondition, that is, the relation that the initial values and the final values of the variables `X,Y,Z` must satisfy.
- $R_0(x_0,y_0,z_0,x,y,z)$ and $R_f(x_0,y_0,z_0,x,y,z)$ define the conditions, or assertions, satisfied by the current values of the variables `X,Y,Z`.

Question 1.1 We define

- $P(x_0,y_0,z_0) \stackrel{\text{def}}{=} x_0,y_0,z_0 \in \mathbb{Z}$
- $g \stackrel{\text{def}}{=} \lambda u,v,w.u+v+w$
- $Q(x_0,y_0,z_0,x,y,z) \stackrel{\text{def}}{=} x_0,y_0,z_0,x,y,z \in \mathbb{Z} \wedge x = x_0+y_0+z_0 \wedge y = y_0 \wedge z = z_0$
- $R_0(x_0,y_0,z_0,x,y,z) \stackrel{\text{def}}{=} x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge \mathbb{Z}(x_0,y_0,z_0,x,y,z)$
- $R_f(x_0,y_0,z_0,x,y,z) \stackrel{\text{def}}{=} \mathbb{Z}(x_0,y_0,z_0,x,y,z) \wedge x = x_0+y_0+z_0 \wedge y = y_0 \wedge z = z_0$

Write the verification conditions of this contract and check that they hold.

Question 1.2 We define

- $P(x_0,y_0,z_0) \stackrel{\text{def}}{=} x_0,y_0,z_0 \in \mathbb{Z}$
- $g \stackrel{\text{def}}{=} \lambda u,v,w.max(u,v,w)$
- $Q(x_0,y_0,z_0,x,y,z) \stackrel{\text{def}}{=} x_0,y_0,z_0 \in \mathbb{Z} \wedge x,y,z \in \mathbb{Z} \wedge x = max(x_0,y_0,z_0) \wedge y = y_0 \wedge z = z_0$
- $R_0(x_0,y_0,z_0,x,y,z) \stackrel{\text{def}}{=} \mathbb{Z}(x_0,y_0,z_0,x,y,z)$

Write the verification conditions of this contract and check that they hold. In particular, define a relation $R_f(x_0,y_0,z_0,x,y,z)$ that establishes correctness according to the rules.

✔ Solution of exercise 1

Listing 1: TLA$^+$ module appex3_1.tla

```
---------------------- MODULE appex3_1 -------------------------------
EXTENDS Naturals, Integers, TLC
CONSTANTS x0,y0,z0,max,undef
min == -max
Z == min..max
P(x,y,z) == x \in Z /\ y \in Z /\ z \in Z
Q(x,y,z,xx,yy,zz) ==
                  /\ x \in Z /\ y \in Z /\ z \in Z
                  /\ xx \in Z /\ yy \in Z /\ zz \in Z
                  /\ xx = x +y+z /\ yy=y /\ zz=z

----------------------------------------
(* precondition *)
ASSUME x0 \in Z /\ y0 \in Z /\ z0 \in Z
----------------------------------------
(*
--algorithm appex3_1 {
  variables x=x0,
            y = y0,
            z=z0;

{
l0: assert x = x0  /\ y=y0 /\ z=z0 ;
    x := x+y+z;
l1: assert x = x0+y0+z0 /\ y=y0 /\ z=z0 ;
}
}
*)
\* BEGIN TRANSLATION (chksum(pcal) = "346fb05e" /\ chksum(tla) = "d5fed099")
VARIABLES x, y, z, pc

vars == << x, y, z, pc >>

Init == (* Global variables *)
        /\ x = x0
        /\ y = y0
        /\ z = z0
        /\ pc = "l0"

l0 == /\ pc = "l0"
      /\ Assert(x = x0  /\ y=y0 /\ z=z0,
                "Failure␣of␣assertion␣at␣line␣23,␣column␣5.")
      /\ x' = x+y+z
      /\ pc' = "l1"
      /\ UNCHANGED << y, z >>

l1 == /\ pc = "l1"
      /\ Assert(x = x0+y0+z0 /\ y=y0 /\ z=z0,
                "Failure␣of␣assertion␣at␣line␣25,␣column␣5.")
      /\ pc' = "Done"
      /\ UNCHANGED << x, y, z >>

(* Allow infinite stuttering to prevent deadlock on termination. *)
Terminating == pc = "Done" /\ UNCHANGED vars

Next == l0 \/ l1
           \/ Terminating

Spec == Init /\ [][Next]_vars

Termination == <>(pc = "Done")

\* END TRANSLATION

-------------------------------------------------
ISDEF(X,Y) == X # undef => X \in Y
DD(X) == X # undef => X \in min..max
-------------------------------------------------
i ==
```

```
    /\ pc \in {"l0","l1","Done"}
    /\ ISDEF(x,Int)  /\ ISDEF(y,Int) /\ ISDEF(z,Int)
    /\ pc = "l0" => x = x0  /\ y=y0 /\ z=z0
    /\ pc = "l1" => x = x0+y0+z0 /\ y=y0 /\ z=z0
    /\ pc="Done" => Q(x0,y0,z0,x,y,z)

safetyrte ==DD(x) /\ DD(y) /\ DD(z)
safetypc == pc="Done" => Q(x0,y0,z0,x,y,z)
check == i /\ safetypc /\ safetyrte
=============================================================================
```

✎ Exercise 2 appex3_21.tla

Show that each annotation is correct or incorrect, according to the verification conditions stated as follows:
$\forall v, v'.P_\ell(v) \wedge cond_{\ell,\ell'}(v) \wedge v' = f_{\ell,\ell'}(v) \Rightarrow P_{\ell'}(v')$
This condition is first written
$\forall v, v', pc, pc'.pc = \ell \wedge P_\ell(v) \wedge pc = \ell \wedge cond_{\ell,\ell'}(v) \wedge v' = f_{\ell,\ell'}(v) \wedge pc' = \ell' \Rightarrow pc' = \ell' \wedge P_{\ell'}(v')$
but it can be reduced by forgetting the variable pc. For each question, you will also write a TLA module that checks whether the annotation is correct.

- (appex3_21.tla)

$$\ell_1 : x = 10 \wedge y = z{+}x \wedge z = 2{\cdot}x$$
$$y := z{+}x$$
$$\ell_2 : x = 10 \wedge y = x{+}2{\cdot}10$$

- $$\ell_1 : x = 1 \wedge y = 12$$
$$x := 2{\cdot}y$$
$$\ell_2 : x = 1 \wedge y = 24$$

- We assume that p is a prime number:

$$\ell_1 : x = 2^p \wedge y = 2^{p+1} \wedge x{\cdot}y = 2^{2\cdot p+1}$$
$$x := y{+}x{+}2^x$$
$$\ell_2 : x = 5{\cdot}2^p \wedge y = 2^{p+1}$$

- $$\ell_1 : x = 11 \wedge y = 13$$
$$z := x; x := y; y := z;$$
$$\ell_2 : x = 26/2 \wedge y = 33/3$$

✔ Solution of exercise 2

The module for the first annotation:

Listing 2: TLA$^+$ module appex3_21.tla

```
---------------------- MODULE appex3_21 -------------------------------
EXTENDS Naturals, Integers, TLC
CONSTANTS x0,y0,z0,max,undef
min == -max
Z == min..max
P(x,y,z) == x= 10 /\  y = z+x /\ z=2*x
Q(x,y,z,xx,yy,zz) ==
                  /\ x= 10 /\  y = z+x /\ z=2*x
                  /\ xx= 10 /\ yy = xx+2*10
-----------------------------------------
(* precondition *)
ASSUME P(x0,y0,z0)
-----------------------------------------
(*
--algorithm appex3_21 {
  variables x=x0,
            y = y0,
            z=z0;

{
l0: assert x= 10 /\  y = z+x /\ z=2*x;
    y := z+x;
l1: assert x= 10 /\ y = x+2*10 ;
}
}
*)
\* BEGIN TRANSLATION (chksum(pcal) = "218092e6" /\ chksum(tla) = "a768c50f")
```

```
VARIABLES x, y, z, pc

vars == << x, y, z, pc >>

Init == (* Global variables *)
        /\ x = x0
        /\ y = y0
        /\ z = z0
        /\ pc = "l0"

l0 == /\ pc = "l0"
      /\ Assert(x= 10 /\  y = z+x /\ z=2*x,
                "Failure of assertion at line 21, column 5.")
      /\ y' = z+x
      /\ pc' = "l1"
      /\ UNCHANGED << x, z >>

l1 == /\ pc = "l1"
      /\ Assert(x= 10 /\ y = x+2*10,
                "Failure of assertion at line 23, column 5.")
      /\ pc' = "Done"
      /\ UNCHANGED << x, y, z >>

(* Allow infinite stuttering to prevent deadlock on termination. *)
Terminating == pc = "Done" /\ UNCHANGED vars

Next == l0 \/ l1
           \/ Terminating

Spec == Init /\ [][Next]_vars

Termination == <>(pc = "Done")

\* END TRANSLATION

-----------------------------------------------
ISDEF(X,Y) == X # undef => X \in Y
DD(X) == X # undef => X \in min..max
-----------------------------------------------
i ==
    /\ pc \in {"l0","l1","Done"}
    /\ ISDEF(x,Int)  /\ ISDEF(y,Int) /\ ISDEF(z,Int)
    /\ pc = "l0" => x= 10 /\  y = z+x /\ z=2*x
    /\ pc = "l1" =>  x= 10 /\ y = x+2*10

safetyrte ==DD(x) /\ DD(y) /\ DD(z)
safetypc == pc="Done" => Q(x0,y0,z0,x,y,z)
check == i /\ safetypc /\ safetyrte
=============================================================
```

✎ Exercise 3

Show that each annotation is correct or incorrect, according to the verification conditions stated as follows:
$\forall x,y,x',y'.P_\ell(x,y) \wedge cond_{\ell,\ell'}(x,y) \wedge (x',y') = f_{\ell,\ell'}(x,y) \Rightarrow P_{\ell'}(x',y')$

- (1)

$\ell_1 : x = 9 \wedge y = z+x$
$y := x+9$
$\ell_2 : x = 9 \wedge y = x+9$

- (2)

$\ell_1 : x = 1 \wedge y = 3 \wedge x+y = 12$
$x := y+x$
$\ell_2 : x = 567 \wedge y = 34$

- (3)

$\ell_1 : x = 3 \wedge y = 3$
$x := y+x$
$\ell_2 : x = 6 \wedge y = 3$

- (4)

$\ell_1 : x = 1 \wedge y = 3$
$z := x; x := y; y := z;$
$\ell_2 : x = 3 \wedge y = 1$

✔ Solution of exercise 3

c denotes the control variable.

1. $c=\ell_1 \wedge x=9 \wedge y=z+x \wedge$ **TRUE** $\wedge (x',y',c')=(x,x+9,\ell_2) \Rightarrow c'=\ell_2 \wedge x'=9 \wedge y'=x'+9$:
   (a) $c=\ell_1 \wedge x=9 \wedge y=z+x \wedge c'=\ell_2 \Rightarrow c'=\ell_2 \wedge x=9 \wedge x+9=x+9$
   (b) $c=\ell_1 \wedge x=9 \wedge y=z+x \wedge c'=\ell_2 \Rightarrow x=9 \wedge x+9=x+9$
   (c) CORRECT.
2. $c=\ell_1 \wedge x=1 \wedge y=3 \wedge x+y=12 \wedge$ **TRUE** $\wedge (x',y',c')=(y+x,y,\ell_2) \Rightarrow c'=\ell_2 \wedge x'=567 \wedge y'=34$:
   (a) $c=\ell_1 \wedge x=1 \wedge y=3 \wedge x+y=12 \wedge (x',y',c')=(y+x,y,\ell_2) \Rightarrow c'=\ell_2 \wedge y+x=567 \wedge y=34$
   (b) $c=\ell_1 \wedge x=1 \wedge y=3 \wedge x+y=12 \wedge c'=\ell_2 \Rightarrow c'=\ell_2 \wedge y+x=567 \wedge y=34$
   (c) $c=\ell_1 \wedge x=1 \wedge y=3 \wedge x+y=12 \wedge c'=\ell_2 \Rightarrow x+y=4 \wedge x+y=12$
   (d) $c=\ell_1 \wedge x=1 \wedge y=3 \wedge x+y=12 \wedge c'=\ell_2 \Rightarrow$ **FALSE**
   (e) **FALSE** $\Rightarrow c'=\ell_2 \wedge y+x=567 \wedge y=34$
   (f) CORRECT: the precondition $x=1 \wedge y=3 \wedge x+y=12$ is false, so the condition holds trivially.
3. $c=\ell_1 \wedge x=3 \wedge y=3 \wedge$ **TRUE** $\wedge (x',y',c')=(y+x,y,\ell_2) \Rightarrow c'=\ell_2 \wedge x'=6 \wedge y'=3$
   (a) $c=\ell_1 \wedge x=3 \wedge y=3 \wedge c'=\ell_2 \Rightarrow c'=\ell_2 \wedge y+x=6 \wedge y=3$
   (b) $c=\ell_1 \wedge x=3 \wedge y=3 \wedge c'=\ell_2 \Rightarrow c'=\ell_2 \wedge 6=6 \wedge y=3$
   (c) CORRECT.
4. $c=\ell_1 \wedge x=1 \wedge y=3 \wedge$ **TRUE** $\wedge (x',y',z',c')=(y,x,x,\ell_2) \Rightarrow c'=\ell_2 \wedge x'=3 \wedge y'=1$
   (a) $c=\ell_1 \wedge x=1 \wedge y=3 \wedge$ **TRUE** $\wedge (x',y',z',c')=(y,x,x,\ell_2) \Rightarrow c'=\ell_2 \wedge y=3 \wedge x=1$
   (b) CORRECT.

✎ Exercise 4

Show that each annotation is correct or incorrect, according to the verification conditions stated as follows:
$\forall x,y,x',y'.P_\ell(x,y) \wedge cond_{\ell,\ell'}(x,y) \wedge (x',y')=f_{\ell,\ell'}(x,y) \Rightarrow P_{\ell'}(x',y')$

- $\ell_1 : x=3 \wedge y=z+x \wedge z=2\cdot x$
  $y := z+x$
  $\ell_2 : x=3 \wedge y=x+6$
- $\ell_1 : x=2^4 \wedge y=2^{345} \wedge x\cdot y=2^{350}$
  $x := y+x+2^x$
  $\ell_2 : x=2^{56} \wedge y=2^{345}$
- $\ell_1 : x=1 \wedge y=12$
  $x := 2\cdot y+x$
  $\ell_2 : x=1 \wedge y=25$

✎ Exercise 5 `appex3_4.tla`

Consider the following small annotated algorithm:

$l0 : \{v=3\}$
$v := v+2$
$l1 : \{v=5\}$

Write a TLA$^+$ module that makes explicit the transition relation, the initial conditions, the invariant and the safety property for partial correctness.

✔ Solution of exercise 5

In the module below, the invariant i and the safety property $safety$ must agree: at $l1$, $v = 5$ (the module first written had $v = 6$ in i, which TLC would reject).

MODULE an_1

EXTENDS *Integers*, *TLC*

VARIABLES v, pc

$titi \triangleq pc = \text{"l0"} \land v = 3$

$skip \triangleq$ UNCHANGED $\langle pc, v \rangle$
$skip_2 \triangleq pc' = pc \land v' = v$

$trans \triangleq pc = \text{"l0"} \land TRUE \land pc' = \text{"l1"} \land v' = v+2$
$trans_2 \triangleq pc = \text{"l0"} \land pc' = \text{"l1"} \land v' = v+2$
$trans_3 \triangleq$
$\land pc = \text{"l0"} \land TRUE$
$\land pc' = \text{"l1"}$
$\land v' = v+2$

$toto \triangleq skip \lor trans$

$i \triangleq$
$\land pc \in \{\text{"l0"}, \text{"l1"}\}$
$\land pc = \text{"l0"} \Rightarrow v = 3$
$\land pc = \text{"l1"} \Rightarrow v = 5$

$safety \triangleq pc = \text{"l1"} \Rightarrow v = 5$

✎ Exercise 6

Define the verification conditions of partial correctness for the following structures. Define a TLA$^+$ model to check that the annotation is correct.

Question 6.1

$\ell 1 : \{P_{\ell 1}(x, y)\}$
$x := x+y+7;$
$\ell 2 : \{P_{\ell 2}(x, y)\}$

✔ Solution of question 6.1

Verification conditions for partial correctness. pc denotes the control variable.

$$pc = \ell 1 \wedge P_{\ell 1}(x,y) \wedge pc' = \ell 2 \wedge x' = x+y+7 \wedge y' = y \Rightarrow pc' = \ell 2 \wedge P_{\ell 2}(x',y')$$

which simplifies to

$$P_{\ell 1}(x,y) \wedge x' = x+y+7 \wedge y' = y \Rightarrow P_{\ell 2}(x',y')$$

then to

$$P_{\ell 1}(x,y) \wedge x' = x+y+7 \Rightarrow P_{\ell 2}(x',y)$$

and finally to

$$P_{\ell 1}(x,y) \Rightarrow P_{\ell 2}(x+y+7,y)$$

TLA$^+$ model to check the annotation. With $x = 3$ and $y = 8$ initially, the annotation at $\ell 2$ is $x = 18 \wedge y = 8$.

MODULE *an2*

EXTENDS *Naturals*
VARIABLES x, y, pc

Define actions from the text of annotated algorithm

$$al0l_1 \triangleq \\ \wedge pc = \text{"l0"} \\ \wedge pc' = \text{"l1"} \\ \wedge x' = x+y+7 \\ \wedge y' = y$$

Define the computation relation

$$next \triangleq al0l1$$

Define the initial conditions

$$init \triangleq pc = \text{"l0"} \wedge x = 3 \wedge y = 8$$

Define the invariant from the annotation

$$i \triangleq \\ \wedge pc = \text{"l0"} \Rightarrow x = 3 \wedge y = 8 \\ \wedge pc = \text{"l1"} \Rightarrow x = 18 \wedge y = 8$$

Define the safety property to check namely the partial correctness

$$safe \triangleq pc = \text{"l1"} \Rightarrow x = 18 \wedge y = 8$$

$$prop \triangleq i \Rightarrow safe$$
$$Init \triangleq init$$
$$Next \triangleq next$$
$$principe \triangleq init \Rightarrow \wedge prop$$

Modification History

Last modified Wed Sep 21 13:28:17 CEST 2016 by mery

Created Wed Sep 09 17:02:47 CEST 2015 by mery

Question 6.2

$$\begin{array}{l}\ell : \{P_\ell(x,y)\}\\ \quad x,y := y,x;\\ \ell' : \{P_{\ell'}(x,y)\}\end{array}$$

✔ **Solution of question 6.2**

Verification conditions for partial correctness. c denotes the control variable.

$$c = \ell 1 \wedge P_{\ell 1}(x,y) \wedge c' = \ell 2 \wedge (x',y') = (y,x) \Rightarrow c' = \ell 2 \wedge P_{\ell 2}(x',y')$$

which simplifies to

$$P_{\ell 1}(x,y) \wedge x' = y \wedge y' = x \Rightarrow P_{\ell 2}(x',y')$$

and then to

$$P_{\ell 1}(x,y) \Rightarrow P_{\ell 2}(y,x)$$

TLA$^+$ model to check the annotation.

MODULE *an3*

EXTENDS *Naturals*
CONSTANTS a, b
VARIABLES x, y, pc

Define actions from the text of annotated algorithm

$$\begin{array}{l} al1l_2 \triangleq \\ \quad \wedge\, pc = \text{"l1"} \\ \quad \wedge\, pc' = \text{"l2"} \\ \quad \wedge\, x' = y \;\wedge\; y' = x \end{array}$$

$$newaction \triangleq pc = \text{"l2"} \wedge pc' = \text{"l1"} \wedge x' = x \wedge y' = y$$

Define the computation relation

$$next \triangleq al1l2$$

$$newnext \triangleq al1l_2 \vee newaction$$

Define the initial conditions

$$init \triangleq pc = \text{"l1"} \wedge x = a \wedge y = b$$

Define the invariant from the annotation

$$\begin{array}{l} i \triangleq \\ \quad \wedge\, pc = \text{"l1"} \Rightarrow x = a \;\wedge\; y = b \\ \quad \wedge\, pc = \text{"l2"} \Rightarrow x = b \;\wedge\; y = a \end{array}$$

Define the safety property to check namely the partial correctness

$$safe \triangleq pc = \text{"l2"} \Rightarrow x = b \wedge y = a$$

✎ **Exercise 7**

Determine the verification conditions for the bounded loop structure. We assume that S does not modify i.

$\ell_1 : \{P_{\ell_1}(x)\}$
FOR $i := 1$ **TO** n **DO**
  $\ell_2 : \{P_{\ell_2}(i,x)\}$
    $S(x)$;
  $\ell_3 : \{P_{\ell_3}(i,x)\}$
ENDFOR
$\ell_4 : \{P_{\ell_4}(x)\}$

✔ Solution of exercise 7

(1) $c = \ell_1 \wedge P_{\ell_1}(x) \wedge 1 \leq n \wedge c' = \ell_2 \wedge i' = 1 \wedge x' = x \Rightarrow c' = \ell_2 \wedge P_{\ell_2}(i',x')$
(2) $c = \ell_1 \wedge P_{\ell_1}(x) \wedge \neg(1 \leq n) \wedge c' = \ell_4 \wedge x' = x \Rightarrow c' = \ell_4 \wedge P_{\ell_4}(x')$
(3) $c = \ell_3 \wedge P_{\ell_3}(i,x) \wedge i{+}1 \leq n \wedge c' = \ell_2 \wedge i' = i{+}1 \wedge x' = x \Rightarrow c' = \ell_2 \wedge P_{\ell_2}(i',x')$
(4) $c = \ell_3 \wedge P_{\ell_3}(i,x) \wedge \neg(i{+}1 \leq n) \wedge c' = \ell_4 \wedge x' = x \wedge i' = i{+}1 \Rightarrow c' = \ell_4 \wedge P_{\ell_4}(x')$

The conditions for the body of the loop, from ℓ_2 to ℓ_3, are those of the statement $S(x)$, with i unchanged.

✎ Exercise 8

`appex3_77.tla`

Variables: X,Y,Z
Requires : $x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}$
Ensures : $z_f = max(x_0, y_0)$

$\ell_0 : \{\ldots\}$
if $X < Y$ **then**
  $\ell_1 : \{\ldots\}$
  $Z := Y$;
  $\ell_2 : \{\ldots\}$
else
  $\ell_3 : \{\ldots\}$
  $Z := X$;
  $\ell_4 : \{\ldots\}$
;
$\ell_5 : \{\ldots\}$

Algorithm 1: Maximum of two numbers, not annotated

Question 8.1 Complete algorithm 1 by annotating it.

✔ Solution of question 8.1

Variables: X,Y,Z
Requires : $x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}$
Ensures : $z_f = max(x_0, y_0)$

$\ell_0 : \{x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
if $X < Y$ **then**
  $\ell_1 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
  $Z := Y$;
  $\ell_2 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge z = y_0\}$
else
  $\ell_3 : \{x \geq y \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$
  $Z := X$;
  $\ell_4 : \{x \geq y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge z = x_0\}$
;
$\ell_5 : \{z = max(x_0, y_0) \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z}\}$

Algorithm 2: Maximum of two numbers, annotated

Question 8.2 Check that the annotation is correct, by applying the rules of partial correctness.

Question 8.3 Check that the annotation is correct, by translating the annotated algorithm into a TLA module.

✔ Solution of question 8.3

Listing 3: TLA$^+$ module `appex3_77.tla`

```
------------------------------- MODULE appex3_77 --------------------------------
EXTENDS Naturals, Integers
CONSTANTS x0,y0,z0
VARIABLES  x,y,z,pc
ASSUME x0 \in Nat /\ y0 \in Nat
typeInt(u) == u \in Int
maxi(u,v) == IF u < v THEN v ELSE u
pre ==  x0 \in Nat /\ y0 \in Nat /\ z0 \in Int
-------------------------------------------------------------------------
al011 ==
    /\ pc="l0"
    /\ pc'="l1"
    /\ x<y
    /\ z'=z /\ x'=x /\ y'=y
al112 ==
    /\ pc="l1"
    /\ pc'="l2"
    /\ z'=y
    /\ x'=x /\ y'=y
al215 ==
    /\ pc="l2"
    /\ pc'="l5"
    /\ z'=z /\ x'=x /\ y'=y
al013 ==
    /\ pc="l0"
    /\ pc'="l3"
    /\ x \geq y
    /\ z'=z /\ x'=x /\ y'=y
al314 ==
    /\ pc="l3"
    /\ pc'="l4"
    /\ z'=x
    /\ x'=x /\ y'=y
  al415 ==
    /\ pc="l4"
    /\ pc'="l5"
    /\ z'=z /\ x'=x /\ y'=y
---------------------
Next == al011 \/ al112 \/ al215  \/ al013 \/ al314 \/ al415 \/ UNCHANGED <<x,y,z,pc>>
Init == pc="l0" /\ x=x0 /\ y =y0 /\ z = z0
```

```
----------------------
i ==
    /\ typeInt(x) /\ typeInt(y) /\ typeInt(z)
    /\ pc="10" =>  x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="11" => x<y /\ x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="12" => x<y /\ x=x0 /\ y=y0 /\ z=y0 /\ pre
    /\ pc="13" => x \geq y /\ x=x0 /\ y=y0 /\ z=z0 /\ pre
    /\ pc="14" => x \geq y /\ x=x0 /\ y=y0 /\ z=x0  /\ pre
    /\ pc="15" => z = maxi(x0,y0) /\ x=x0 /\ y=y0  /\ pre
safe ==  pc="15" =>  z = maxi(x0,y0)
safeab == x=x0 /\ y=y0
================================================================================
\* Modification History
\* Last modified Wed Sep 29 20:32:22 CEST 2021 by mery
\* Created Wed Sep 09 18:19:08 CEST 2015 by mery
```

Question 8.4 State and check partial correctness.

✔ Solution of question 8.4

First give the precondition and the postcondition, then check the verification conditions of partial correctness: the safety property $safe$ of the module, $pc = \ell_5 \Rightarrow z = maxi(x_0, y_0)$, is implied by the invariant i.

✎ Exercise 9 `appex3_8.tla`

The aim is to study and annotate the proposed program in order to obtain its partial correctness (that is, without proving termination). A state is a set of precise values of the variables of the program (specified by a predicate); we consider a label (ℓ) between each pair of statements of the program. An annotation is the predicate describing the possible values of the variables in a state of the program. This annotation is written $P_\ell(v)$ and expresses the property satisfied by the variable v at ℓ.

Question 9.1 You are asked:

1. to annotate all the labels of the program;
2. to propose a TLA$^+$ model to check the annotations and partial correctness.

Variables: X
Requires : $x_0 \in \mathbb{N}$
Ensures : $x_f = 0$

$\ell_0 : \{\ldots\}$
while $0 < X$ **do**
  $\ell_1 : \{\ldots\}$
  $X := X - 1;$
  $\ell_2 : \{\ldots\}$
;
$\ell_3 : \{\ldots\}$

Algorithm 3: Example, not annotated

Question 9.2 Check the conditions showing that the annotation is valid and that it establishes partial correctness.

Question 9.3 Translate the annotated algorithm into a TLA module containing the definition of the algorithm as `Next` and `Init`, the definition of the invariant given by the annotations, and the definitions of partial correctness and of the absence of run-time errors.

✔ Solution of exercise 9

Annotation. The annotation (algorithm 4) is built by propagating the assertions through the statements. The verification conditions must then be checked.

Variables: X
Requires : $x_0 \in \mathbb{N}$
Ensures : $x_f = 0$

$\ell_0 : \{ x = x_0 \wedge x_0 \in \mathbb{N}\}$
while $0 < X$ **do**
  $\ell_1 : \{0 < x \leq x_0 \wedge x_0 \in \mathbb{N}\}$
  $X := X - 1;$
  $\ell_2 : \{0 \leq x < x_0 \wedge x_0 \in \mathbb{N}\}$
;
$\ell_3 : \{x = 0\}$

Algorithm 4: Example, annotated

TLA$^+$ model to check the annotation.

MODULE *ex3*

EXTENDS *Naturals*

CONSTANTS *x0*
VARIABLES *x*, *pc*

$al0l_1 \triangleq$
  $\wedge\ pc = \text{"l0"}$
  $\wedge\ pc' = \text{"l1"}$
  $\wedge\ 0 < x$
  $\wedge\ x' = x$

$al0l_3 \triangleq$
  $\wedge\ pc = \text{"l0"}$
  $\wedge\ pc' = \text{"l3"}$
  $\wedge\ x = 0$
  $\wedge\ x' = x$

$al1l_2 \triangleq$
  $\wedge\ pc = \text{"l1"}$
  $\wedge\ pc' = \text{"l2"}$
  $\wedge\ x' = x - 1$

$al2l_1 \triangleq$
  $\wedge\ pc = \text{"l2"}$
  $\wedge\ pc' = \text{"l1"}$
  $\wedge\ 0 < x$
  $\wedge\ x' = x$

$al2l_3 \triangleq$
  $\wedge\ pc = \text{"l2"}$
  $\wedge\ pc' = \text{"l3"}$
  $\wedge\ 0 = x$
  $\wedge\ x' = x$

$next \triangleq al0l_1 \vee al0l_3 \vee al1l_2 \vee al2l_1 \vee al2l3$

$init \triangleq pc = \text{"l0"} \wedge x = x0$

$i \triangleq$

$$\begin{array}{l} \wedge\, pc = \text{"l0"} \;\Rightarrow\; x = x0 \\ \wedge\, pc = \text{"l1"} \;\Rightarrow\; 0 < x \wedge x \leq x_0 \\ \wedge\, pc = \text{"l2"} \;\Rightarrow\; 0 \leq x \wedge x \leq x0 \\ \wedge\, pc = \text{"l3"} \;\Rightarrow\; x = 0 \end{array}$$

$$safe \triangleq pc = \text{"l3"} \;\Rightarrow\; x = 0$$

$$safeplus \triangleq x \geq 0$$

Modification History
Last modified Thu Sep 10 09:35:48 CEST 2015 by mery
Created Wed Sep 09 18:07:50 CEST 2015 by mery

✎ Exercise 10

Question 10.1 Let t be an array (with values in $\mathbb{N}$). Give a predicate $max(m,t,a,b) = \ldots$ expressing that a number $m \in \mathbb{N}$ is the maximum of the array t on the interval $a\,..\,b$.

✔ Solution of question 10.1

$$max(m,t,a,b) \stackrel{\text{def}}{=} m \in t[a\,..\,b] \wedge (\forall i \cdot i \in a\,..\,b \Rightarrow t(i) \leq m)$$

(m must be one of the values of t on $a\,..\,b$, hence $t[a\,..\,b]$ rather than the whole range $\text{ran}(t)$.)

Question 10.2 In the same way, give a predicate $sorted(t,a,b)$ specifying that the array t is sorted on the interval $a\,..\,b$.

✔ Solution of question 10.2

$$sorted(t,a,b) \stackrel{\text{def}}{=} \forall i, j \cdot ((i \in a\,..\,b \wedge j \in a\,..\,b \wedge i \leq j) \Rightarrow t(i) \leq t(j))$$

✎ Exercise 11 `appex3_10.tla`

Algorithm 5 computes the maximum of a sequence of integer values. You are asked:

- to define the precondition and the postcondition;
- to annotate this algorithm;
- to check the verification conditions for partial correctness;
- to check the conditions for the absence of run-time errors.

Variables: F,N,M,I

Requires : $\begin{pmatrix} n_0 \in \mathbb{N} \wedge \\ n_0 \neq 0 \wedge \\ f_0 \in 0\,..\,n_0-1 \to \mathbb{N} \end{pmatrix}$

Ensures : $\begin{pmatrix} m_f \in \mathbb{N} \wedge \\ m_f \in ran(f_0) \wedge \\ (\forall j \cdot j \in 0\,..\,n_0-1 \Rightarrow f_0(j) \leq m_f) \end{pmatrix}$

$M := F(0);$
$I := 1;$
while $I < N$ **do**
  if $F(I) > M$ **then**
    $M := F(I);$
  ;
  $I{+}{+};$
;

Algorithm 5: Maximum of a list, not annotated

✔ **Solution of exercise 11**

The annotated algorithm is algorithm 6. To shorten the annotations, *Pre* stands for

$$Pre \stackrel{\text{def}}{=} n_0 \in \mathbb{N} \wedge n_0 \neq 0 \wedge f_0 \in 0\,..\,n_0-1 \to \mathbb{N} \wedge m_0, i_0 \in \mathbb{Z} \wedge n = n_0 \wedge f = f_0.$$

Variables: F,N,M,I

$\ell_0 : \{Pre \wedge i = i_0 \wedge m = m_0\}$
$M := F(0);$
$\ell_1 : \{Pre \wedge i = i_0 \wedge m = f(0)\}$
$I := 1;$
$\ell_2 : \{Pre \wedge i = 1 \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i-1]) \wedge (\forall j \cdot j \in 0\,..\,i-1 \Rightarrow f(j) \leq m)\}$
while $I < N$ **do**
  $\ell_3 : \{Pre \wedge i \in 1..n-1 \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i-1]) \wedge (\forall j \cdot j \in 0\,..\,i-1 \Rightarrow f(j) \leq m)\}$
  if $F(I) > M$ **then**
    $\ell_4 : \{Pre \wedge i \in 1..n-1 \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i-1]) \wedge (\forall j \cdot j \in 0\,..\,i-1 \Rightarrow f(j) \leq m) \wedge f(i) > m\}$
    $M := F(I);$
    $\ell_5 : \{Pre \wedge i \in 1..n-1 \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i]) \wedge (\forall j \cdot j \in 0\,..\,i \Rightarrow f(j) \leq m)\}$
  ;
  $\ell_6 : \{Pre \wedge i \in 1..n-1 \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i]) \wedge (\forall j \cdot j \in 0\,..\,i \Rightarrow f(j) \leq m)\}$
  $I{+}{+};$
  $\ell_7 : \{Pre \wedge i \in 2..n \wedge m \in \mathbb{N} \wedge m \in ran(f[0..i-1]) \wedge (\forall j \cdot j \in 0\,..\,i-1 \Rightarrow f(j) \leq m)\}$
;
$\ell_8 : \{Pre \wedge i = n \wedge m \in \mathbb{N} \wedge m \in ran(f[0..n-1]) \wedge (\forall j \cdot j \in 0\,..\,n-1 \Rightarrow f(j) \leq m)\}$

Algorithm 6: Maximum of a list, annotated

A first TLA$^+$ model, with the array $f = [j \in 0..n-1 \mapsto j]$:

MODULE *algo_maximum*

computing the maximum value of an array f

EXTENDS *Naturals*, *TLC*
CONSTANTS n
VARIABLES m, i, l

$f \triangleq [j \in 0..n-1 \mapsto j]$

$Init \triangleq \wedge i = 0$
$\quad \wedge m = 0$
$\quad \wedge l = \text{"l0"}$

$$
\begin{aligned}
l0l_1 \triangleq\ & \wedge l = \text{"l0"} \\
& \wedge m' = f[0] \\
& \wedge i' = i \\
& \wedge l' = \text{"l1"}
\end{aligned}
$$

$$
\begin{aligned}
l1l_2 \triangleq\ & \wedge l = \text{"l1"} \\
& \wedge m' = m \\
& \wedge i' = 1 \\
& \wedge l' = \text{"l2"}
\end{aligned}
$$

$$
\begin{aligned}
l2l_3 \triangleq\ & \wedge l = \text{"l2"} \\
& \wedge i < n \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l3"}
\end{aligned}
$$

$$
\begin{aligned}
l2l_8 \triangleq\ & \wedge l = \text{"l2"} \\
& \wedge (i \geq n) \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l8"}
\end{aligned}
$$

$$
\begin{aligned}
l3l_4 \triangleq\ & \wedge l = \text{"l3"} \\
& \wedge f[i] > m \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l4"}
\end{aligned}
$$

$$
\begin{aligned}
l3l_6 \triangleq\ & \wedge l = \text{"l3"} \\
& \wedge (f[i] \leq m) \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l6"}
\end{aligned}
$$

$$
\begin{aligned}
l4l_5 \triangleq\ & \wedge l = \text{"l4"} \\
& \wedge m' = f[i] \\
& \wedge i' = i \\
& \wedge l' = \text{"l5"}
\end{aligned}
$$

$$
\begin{aligned}
l5l_6 \triangleq\ & \wedge l = \text{"l5"} \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l6"}
\end{aligned}
$$

$$
\begin{aligned}
l6l_7 \triangleq\ & \wedge l = \text{"l6"} \\
& \wedge m' = m \\
& \wedge i' = i + 1 \\
& \wedge l' = \text{"l7"}
\end{aligned}
$$

$$
\begin{aligned}
l7l_2 \triangleq\ & \wedge l = \text{"l7"} \\
& \wedge m' = m \\
& \wedge i' = i \\
& \wedge l' = \text{"l2"}
\end{aligned}
$$

$$
\begin{aligned}
Next \triangleq\ & \vee l0l1 \\
& \vee l1l2 \\
& \vee l2l3
\end{aligned}
$$

$$
\begin{array}{l}
\quad \vee\ l2l8 \\
\quad \vee\ l3l4 \\
\quad \vee\ l3l6 \\
\quad \vee\ l4l5 \\
\quad \vee\ l5l6 \\
\quad \vee\ l6l7 \\
\quad \vee\ l7l2
\end{array}
$$

$$
\begin{array}{l}
Safel_3 \triangleq\ l = \text{"l3"} \Rightarrow \wedge\ (i \in 1..n-1) \\
\qquad\qquad\qquad\qquad\quad \wedge\ (\exists\, k\ :\ (k \in 0..i-1) \wedge f[k] = m) \\
\qquad\qquad\qquad\qquad\quad \wedge\ (\forall\, j\ :\ j \in 0..i-1 \Rightarrow f[j] \leq m) \\
safety \triangleq\ l = \text{"l8"} \Rightarrow (\forall\, k \in 0..n-1 : m \geq f[k])
\end{array}
$$

partial correctnes

$$
Safety_2 \triangleq\ l \neq \text{"l8"}
$$

The complete model, with the precondition, the invariant and the absence of run-time errors (`runtimeerrors`). In the action `l7l3`, the line `/\ pc= "l3"` must read `/\ pc'= "l3"` (the prime is missing).

Listing 4: TLA$^+$ module `appex3_10.tla`

```
------------------------------ MODULE appex3_10 ------------------------------
(* computing the maximum value of an array f *)

EXTENDS Naturals, TLC,Integers
CONSTANTS undef,n0,f0,i0,m0,min,max
VARIABLES n,f,m,i,pc
---------------------------------
def0 == [j \in 0..n0-1 |-> n0-j]

---------------------------------
(* precondition *)

ASSUME n0 \in Nat /\ n0 # 0  /\ f0 = def0 /\ i0 \in Int

Init == /\ i = i0
        /\ m = m0
        /\ f=f0
        /\ n=n0
        /\ pc = "l0"
----------------------------------
l0l1 == /\ pc = "l0"
        /\ m' = f[0]
        /\ pc'= "l1"
        /\ UNCHANGED <<n,f,i>>


l1l2 == /\ pc = "l1"
        /\ i' = 1
        /\ pc'= "l2"
        /\ UNCHANGED <<n,f,m>>


l2l3 == /\ pc = "l2"
        /\ i < n
        /\ pc'= "l3"
        /\ UNCHANGED <<n,f,m,i>>


l2l8 == /\ pc = "l2"
        /\ (i \geq n)
        /\ m' = m
        /\ i' = i
        /\ pc'= "l8"
        /\ UNCHANGED <<n,f>>

l3l4 == /\ pc = "l3"
        /\ f[i] > m
```

```
        /\ m' = m
        /\ i' = i
        /\ pc'= "l4"
        /\ UNCHANGED <<n,f>>


l3l6 == /\ pc = "l3"
        /\ (f[i] \leq m)
        /\ m' = m
        /\ i' = i
        /\ pc'= "l6"
        /\ UNCHANGED <<n,f>>

l4l5 == /\ pc = "l4"
        /\ m' = f[i]
        /\ i' = i
        /\ pc'= "l5"
        /\ UNCHANGED <<n,f>>


l5l6 == /\ pc = "l5"
        /\ m' = m
        /\ i' = i
        /\ pc'= "l6"
        /\ UNCHANGED <<n,f>>

l6l7 == /\ pc = "l6"
        /\ m' = m
        /\ i' = i + 1
        /\ pc'= "l7"
       /\ UNCHANGED <<n,f>>


l7l3 == /\ pc = "l7"
        /\ i < n
        /\ m' = m
        /\ i' = i
        /\ pc= "l3"
        /\ UNCHANGED <<n,f>>

 l7l8 ==
        /\ pc = "l7"
        /\ i \geq n
        /\ m' = m
        /\ i' = i
        /\ pc'= "l8"
        /\ UNCHANGED <<n,f>>



Next == \/ l0l1
        \/ l1l2
        \/ l2l3
        \/ l2l8
        \/ l3l4
        \/ l3l6
        \/ l4l5
        \/ l5l6
        \/ l6l7
        \/ l7l3
        \/ l7l8
        \/ UNCHANGED <<n,m,i,f,pc>>

pre0 == n0 \in Nat /\ n0 # 0  /\ f0 = def0 /\ i0 \in Int
pre1 == f=f0 /\ n=n0 /\ pre0

   zinf == min..max
  ninf == 0..max


  Dl0l1 ==  0\leq 0 /\ 0 \leq n0-1
  Dl1l2 == 1 \in zinf
 inv ==
```

```
    /\ pc \in {"l0","l1","l2","l3","l4","l5","l6","l7","l8"}
    /\ n \in Int /\ f = def0 /\ i \in Int /\ m \in Int
    /\ pc="l0" =>   f=f0 /\ n=n0 /\ m=m0 /\ i = i0/\ pre0 /\ Dl0l1
    /\ pc="l1" =>  f=f0 /\ n=n0 /\ m=f[0] /\ i = i0 /\ pre0 /\ Dl1l2
    /\ pc ="l2" =>f=f0 /\ n=n0 /\ m=f[0] /\ i = 1 /\ pre0
    /\ pc="l3" =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\ (i < n
        ) /\  pre1
    /\ pc="l4" =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\ (i < n
        ) /\ (f[i]>m) /\pre1
    /\ pc="l5"  =>  (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m) /\ (i < n)
         /\ (f[i]>m) /\pre1
    /\ pc="l6" =>   (\E j \in 0..i : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m)  /\pre1
    /\ pc="l7" =>   (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i-1: f[k] \leq m) /\ (i \
        leq n)  /\pre1
    /\ pc="l8" =>   (\E j \in 0..i-1 : f[j]=m) /\ (\A k \in 0..i: f[k] \leq m)  /\pre1 /\
         i=n


runtimeerrors ==  m \in zinf /\ i \in zinf /\ n \in zinf


================================================================================
```

✎ Exercise 12 appex3_12.tla

We consider the algorithm *squareroot* (algorithm 7), which computes the integer square root of a natural number $x \in \mathbb{N}$.

Question 12.1 Complete this algorithm by proposing three assertions:

- $P_{\ell_2}(z, y1, y2, y3)$
- $P_{\ell_4}(z, y1, y2, y3)$
- $P_{\ell_5}(z, y1, y2, y3)$

Question 12.2 For each pair (ℓ, ℓ') of labels corresponding to an elementary step, we check the following property: $\forall y_1, y_2, y_3, z, y'_1, y'_2, y'_3, z'. P_\ell(y_1, y_2, y_3, z) \wedge cond_{\ell,\ell'}(y_1, y_2, y_3, z) \wedge (y'_1, y'_2, y'_3, z') = f_{\ell,\ell'}(y_1, y_2, y_3, z) \Rightarrow P_{\ell'}(y'_1, y'_2, y'_3, z')$. State and check this property for the following pairs of labels: (ℓ_1, ℓ_2); (ℓ_1, ℓ_4); (ℓ_2, ℓ_3); (ℓ_3, ℓ_2); (ℓ_3, ℓ_4); (ℓ_4, ℓ_5).

Question 12.3 We assume that all the verification conditions associated with the pairs of successive labels of the algorithm hold. Which two conditions must be shown to deduce that the algorithm is partially correct with respect to the precondition and the postcondition? Give these conditions explicitly and explain why they hold.

Question 12.4 Explain why this algorithm has no run-time errors, provided that the initial data belong to a domain, to be defined, included in the domain of computer integers, that is, the integers that can be coded on n bits. The set of computer integers on n bits is written $\mathbb{Z}_n$ and defined by $\{i \mid i \in \mathbb{Z} \wedge -2^{n-1} \leq i \wedge i \leq 2^{n-1}-1\}$.

Variables: X,Y1,Y2,Y3,Z
Requires : $x_0 \in \mathbb{N}$
Ensures : $z_f^2 \leq x_0 \wedge x_0 < (z_f+1)^2$

$\ell_0 : \{x_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge y1_0 \in \mathbb{Z} \wedge y2_0 \in \mathbb{Z} \wedge y3_0 \in \mathbb{Z} \wedge (x, y1, y2, y3, z) = (x_0, y1_0, y2_0, y3_0, z_0)\}$
$(y_1, y_2, y_3) := (0, 1, 1)$;
$\ell_1 : \{x_0 \in \mathbb{N} \wedge z_0 \in \mathbb{Z} \wedge y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge y1\cdot y1 \leq x \wedge (x, z) = (x_0, z_0)\}$
while $y_2 \leq x$ **do**
  $\ell_2 : \{\ldots\}$
  $(y_1, y_2, y_3) := (y_1+1, y_2+y_3+2, y_3+2)$;
  $\ell_3 : \{\ldots\}$
;
$\ell_4 : \{\ldots\}$
$z := y_1$;
$\ell_5 : \{\ldots\}$

Algorithm 7: *squareroot*, partly annotated

✔ **Solution of exercise 12**

The annotated algorithm is algorithm 8.

precondition : $x \in \mathbb{N}$
postcondition : $z^2 \leq x \wedge x < (z+1)^2$
local variables: $y_1, y_2, y_3 \in \mathbb{N}$

$\ell_0 : \{x \in \mathbb{N} \wedge z \in \mathbb{Z} \wedge y1 \in \mathbb{Z} \wedge y2 \in \mathbb{Z} \wedge y3 \in \mathbb{Z}\}$
$(y_1, y_2, y_3) := (0, 1, 1);$
$\ell_1 : \{y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge y1\cdot y1 \leq x\}$
while $y_2 \leq x$ **do**
  $\ell_2 : \{y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge y2 \leq x\}$
  $(y_1, y_2, y_3) := (y_1+1, y_2+y_3+2, y_3+2);$
  $\ell_3 : \{y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge y1\cdot y1 \leq x\}$
;
$\ell_4 : \{y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge y1\cdot y1 \leq x \wedge x < y2\}$
$z := y_1;$
$\ell_5 : \{y2 = (y1+1)\cdot(y1+1) \wedge y3 = 2\cdot y1+1 \wedge x < y2 \wedge z = y1 \wedge$
  $z\cdot z \leq x \wedge x < (z+1)\cdot(z+1)\}$

Algorithm 8: *squareroot*, annotated

MODULE *algo_squareroot*

EXTENDS *Integers, TLC*
CONSTANTS x x is the input
VARIABLES $pc, y_1, y2, y_3, z$

$vars \triangleq \langle pc, y_1, y2, y_3, z\rangle$
$al0l_1 \triangleq pc = \text{"l0"} \wedge y'_1 = 0 \wedge y'_2 = 1 \wedge y'_3 = 1 \wedge pc' = \text{"l1"} \wedge z' = z$

$al1l_2 \triangleq pc = \text{"l1"} \wedge y_2 \leq x \wedge pc' = \text{"l2"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z\rangle$

$al1l_4 \triangleq pc = \text{"l1"} \wedge y_2 > x \wedge pc' = \text{"l4"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z\rangle$

$al2l_3 \triangleq pc = \text{"l2"} \wedge y'_1 = y_1+1 \wedge y'_2 = y_2+y3+2 \wedge y'_3 = y_3+2 \wedge pc' = \text{"l3"} \wedge z' = z$

$al3l_2 \triangleq pc = \text{"l3"} \wedge y_2 \leq x \wedge pc' = \text{"l2"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z\rangle$

$al3l_4 \triangleq pc = \text{"l3"} \wedge y_2 > x \wedge pc' = \text{"l4"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3, z\rangle$

$al4l_5 \triangleq pc = \text{"l4"} \wedge z' = y_1 \wedge pc' = \text{"l5"} \wedge \text{UNCHANGED } \langle y_1, y2, y_3\rangle$

$Init \triangleq y_1 = 0 \wedge y_2 = 0 \wedge y_3 = 0 \wedge z = 0 \wedge pc = \text{"l0"}$

$Next \triangleq al0l_1 \vee al1l_2 \vee al1l_4 \vee al2l_3 \vee al3l_2 \vee al3l_4 \vee al4l_5$

$MAX \triangleq 32768$ 16 bits
$D \triangleq 0..32768$
x \leq 32760

$Safety_absence \triangleq (y_1 \in D) \wedge (y_2 \in D) \wedge (y_3 \in D) \wedge (z \in D)$

$i \triangleq$
$\wedge pc = \text{"l0"} \Rightarrow y_1 \in D \wedge y_2 \in D \wedge y_3 \in D \wedge z \in D$
$\wedge pc = \text{"l1"} \Rightarrow y_2 = (y_1+1)\cdot(y_1+1) \wedge y_3 = 2\cdot y_1+1 \wedge y_1\cdot y1 \leq x \wedge Safety_absence$
$\wedge pc = \text{"l2"} \Rightarrow y_2 = (y_1+1)\cdot(y_1+1) \wedge y_3 = 2\cdot y_1+1 \wedge y_1\cdot y1 \leq x \wedge y_2 \leq x \wedge Safety_absence$
$\wedge pc = \text{"l3"} \Rightarrow y_2 = (y_1+1)\cdot(y_1+1) \wedge y_3 = 2\cdot y_1+1 \wedge y_1\cdot y1 \leq x \wedge Safety_absence$
$\wedge pc = \text{"l4"} \Rightarrow y_2 = (y_1+1)\cdot(y_1+1) \wedge y_3 = 2\cdot y_1+1 \wedge y_1\cdot y1 \leq x \wedge x < y_2 \wedge Safety_absence$
$\wedge pc = \text{"l5"} \Rightarrow y_2 = (y_1+1)\cdot(y_1+1) \wedge y_3 = 2\cdot y_1+1 \wedge z\cdot z \leq x \wedge x < (z+1)\cdot(z+1) \wedge Safety_absence$

$$\mathit{Safety_partialcorrectness} \triangleq pc = \text{"l5"} \Rightarrow \begin{array}{l} \wedge\, y_2 = (y_1{+}1)\cdot(y_1{+}1) \\ \wedge\, y_3 = 2\cdot y_1{+}1 \\ \wedge\, z\cdot z \leq x \wedge x < (z{+}1)\cdot(z{+}1) \end{array}$$

✎ Exercise 13

For each question, show that the annotation is correct or incorrect, according to the verification conditions stated as follows: $\forall v, v'.P_\ell(v) \wedge cond_{\ell,\ell'}(v) \wedge v' = f_{\ell,\ell'}(v) \Rightarrow P_{\ell'}(v')$. You must answer by stating and proving the *verification conditions*.

Question 13.1

$\ell_1 : x = 12 \wedge y = 2 \wedge z = 3{\cdot}x$
$x := z{+}y$
$\ell_2 : x = 38 \wedge y = 2$

Question 13.2

$\ell_1 : x = 3 \wedge y = 9$
$x := 3{\cdot}y$
$\ell_2 : x = 27 \wedge y = 9$

Question 13.3 Let p be a number that is not a multiple of 3, that is, different from 3, 6, 9, 12, ...

$\ell_1 : x = 3{+}z \wedge y = 1 \wedge z = 3 \wedge x = y$
$x := p{\cdot}y$
$\ell_2 : x = z \wedge y = z \wedge z = 4{\cdot}p$

Question 13.4 Let r be a cube, that is, a number of the form $r = q^3$.

$\ell_1 : x = r \wedge u = x^r \wedge z = 6 \wedge x = u$
$y := r{\cdot}r{\cdot}r$
$\ell_2 : x = z \wedge y = z \wedge z = 4{\cdot}p$

✎ Exercise 14 `appex3_13.tla`

Consider the following annotated algorithm, with its precondition and postcondition. We assume that $x1$ and $x2$ are constants.

Variables: X1,X2,Y1,Y2,Y3,Z
Requires : $x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0$
Ensures : $z_f = {x1_0}^{x2_0}$

$\ell_0 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, y1, y2, y3, z) = (x1_0, x2_0, y1_0, y2_0, y3_0, z_0)\}$
$(y_1, y_2, y_3) := (x_1, x_2, 1);$
$\ell_1 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0) \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2}\}$
while $y_2 \neq 0$ **do**
  $\ell_2 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0) \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 < y2 \leq x2\}$
  if $odd(y_2)$ **then**
    $\ell_3 : \{\ldots\}$
    $y_2 := y_2 - 1;$
    $\ell_4 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0) \wedge y_3 \cdot y1 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge even(y2)\}$
    $y_3 := y_3 \cdot y_1;$
    $\ell_5 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0) \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge even(y2)\}$
  ;
  $\ell_6 : \{\ldots\}$
  $y_1 := y_1 \cdot y_1;$
  $\ell_7 : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0) \wedge y_3 \cdot {y_1}^{y_2\ div\ 2} = {x_1}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge even(y2)\}$
  $y_2 := y_2\ div\ 2;$
  $\ell_8 : \{\ldots\}$
;
$\ell_9 : \{\ldots\}$
$z := y_3;$
$\ell_{10} : \{x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2) = (x1_0, x2_0) \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y2 = 0 \wedge z = {x1}^{x2}\}$

Algorithm 9: Indian exponentiation, partly annotated

Question 14.1 Complete the annotations associated with each label $\ell \in \{\ell_3, \ell_6, \ell_8, \ell_9\}$. You must write the complete annotation of each requested control point.

✔ **Solution of question 14.1**

With $A \stackrel{\text{def}}{=} x1_0 \in \mathbb{N} \wedge x2_0 \in \mathbb{N} \wedge x1_0 \neq 0 \wedge y1_0, y2_0, y3_0, z_0 \in \mathbb{Z} \wedge (x1, x2, z) = (x1_0, x2_0, z_0)$:

- $\ell_3 : \{A \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 < y2 \leq x2 \wedge odd(y2)\}$
- $\ell_6 : \{A \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 \leq y2 \leq x2 \wedge even(y2)\}$
- $\ell_8 : \{A \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge 0 \leq y2 \leq x2\}$
- $\ell_9 : \{A \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y2 = 0\}$

A TLA$^+$ model of the annotated algorithm:

Listing 5: TLA$^+$ module `appex3_13.tla`

```
---------------------- MODULE appex3_13 --------------------------------
EXTENDS Naturals, Integers, TLC

CONSTANT MAXINT,x10,x20,MININT
ASSUME x10 \in Nat /\ x20 \in Nat /\ x10 # 0

(*
-termination

--fair algorithm Power {
  variables x1=x10,x2=x20,y1, y2,y3,z;
{
    l0: assert x1=x10 /\ x2=x20 /\ y1,y2,y3,z \in Int;
    y1:=x1;
```

```
    y2:=x2;
    y3:=1;
    l1: assert
   l1:while (y2 /= 0) {

      l2:if ( y2 % 2 # 0) {
        l3:y2:=y2-1;
        l4:y3:=y3*y1;
        l5:skip;
      };
      l6:y1 := y1*y1;
      l7:y2:= y2 \div   2;
      l8:skip;
    };
    l9: z := y3;
    l10: print <<x1, x2,z>>;
}

}
*)
\* BEGIN TRANSLATION
CONSTANT defaultInitValue
VARIABLES x1, x2, y1, y2, y3, z, pc

vars == << x1, x2, y1, y2, y3, z, pc >>

Init == (* Global variables *)
        /\ x1 = x10
        /\ x2 = x20
        /\ y1 = defaultInitValue
        /\ y2 = defaultInitValue
        /\ y3 = defaultInitValue
        /\ z = defaultInitValue
        /\ pc = "l0"

l0 == /\ pc = "l0"
      /\ PrintT(<<x1, x2>>)
      /\ y1' = x1
      /\ y2' = x2
      /\ y3' = 1
      /\ pc' = "l1"
      /\ UNCHANGED << x1, x2, z >>

l1 == /\ pc = "l1"
      /\ IF y2 /= 0
            THEN /\ pc' = "l2"
            ELSE /\ pc' = "l9"
      /\ UNCHANGED << x1, x2, y1, y2, y3, z >>

l2 == /\ pc = "l2"
      /\ IF y2 % 2 # 0
            THEN /\ pc' = "l3"
            ELSE /\ pc' = "l6"
      /\ UNCHANGED << x1, x2, y1, y2, y3, z >>

l3 == /\ pc = "l3"
      /\ y2' = y2-1
      /\ pc' = "l4"
      /\ UNCHANGED << x1, x2, y1, y3, z >>

l4 == /\ pc = "l4"
      /\ y3' = y3*y1
      /\ pc' = "l5"
      /\ UNCHANGED << x1, x2, y1, y2, z >>

l5 == /\ pc = "l5"
      /\ TRUE
      /\ pc' = "l6"
      /\ UNCHANGED << x1, x2, y1, y2, y3, z >>

l6 == /\ pc = "l6"
      /\ y1' = y1*y1
      /\ pc' = "l7"
```

```
      /\ UNCHANGED << x1, x2, y2, y3, z >>

l7 == /\ pc = "l7"
      /\ y2' = (y2 \div   2)
      /\ pc' = "l8"
      /\ UNCHANGED << x1, x2, y1, y3, z >>

l8 == /\ pc = "l8"
      /\ TRUE
      /\ pc' = "l1"
      /\ UNCHANGED << x1, x2, y1, y2, y3, z >>

l9 == /\ pc = "l9"
      /\ z' = y3
      /\ pc' = "l10"
      /\ UNCHANGED << x1, x2, y1, y2, y3 >>

l10 == /\ pc = "l10"
       /\ PrintT(<<x1, x2,z>>)
       /\ pc' = "Done"
       /\ UNCHANGED << x1, x2, y1, y2, y3, z >>

(* Allow infinite stuttering to prevent deadlock on termination. *)
Terminating == pc = "Done" /\ UNCHANGED vars

Next == l0 \/ l1 \/ l2 \/ l3 \/ l4 \/ l5 \/ l6 \/ l7 \/ l8 \/ l9 \/ l10
           \/ Terminating

Spec == /\ Init /\ [][Next]_vars
        /\ WF_vars(Next)

Termination == <>(pc = "Done")

\* END TRANSLATION

L == {"l0","l1"}
D == MININT..MAXINT

DD(X) == X=defaultInitValue => X \in D

i ==
    /\ pc \in L
    /\ DD(y1) /\ DD(y2) /\ DD(y3) /\ DD(z)
    /\ pc="l1" => y1=x1 /\ y2=x2 /\ y3=1
Q1 == pc # "Done"
Qpc == pc ="Done" => z=x1^x2


=============================================================================
\* Modification History
\* Last modified Mon Oct 04 20:34:54 CEST 2021 by mery
\* Created Wed Sep 09 17:02:47 CEST 2015 by mery
```

Question 14.2 For each pair (ℓ, ℓ') of labels corresponding to an elementary step, we check the following property: $\forall y_1, y_2, y_3, z, y_1', y_2', y_3', z'. P_\ell(y_1, y_2, y_3, z) \wedge cond_{\ell,\ell'}(y_1, y_2, y_3, z) \wedge (y_1', y_2', y_3', z') = f_{\ell,\ell'}(y_1, y_2, y_3, z) \Rightarrow P_{\ell'}(y_1', y_2', y_3', z')$. State and check this property for the following pairs of labels: (ℓ_0, ℓ_1); (ℓ_1, ℓ_2); (ℓ_3, ℓ_4); (ℓ_6, ℓ_7); (ℓ_7, ℓ_8); (ℓ_1, ℓ_9); (ℓ_9, ℓ_{10}). This check also confirms the annotations completed in the previous question.

Question 14.3 We assume that all the verification conditions associated with the pairs of successive labels of the algorithm hold. Which two conditions must be shown to deduce that the algorithm is partially correct with respect to the precondition and the postcondition? Give these conditions explicitly and explain why they hold.

Question 14.4 In mathematics, the power $x_1^{x_2}$ is defined here for a non-zero value x_1, which is why the precondition requires x_1 to be different from 0. However, if a zero value of x_1 is used, the algorithm still runs and returns a value. One day, a mathematician applied this algorithm without checking whether x_1 was zero, and lost his temper! He accuses you of not having supplied the right algorithm for his requirements, and claims damages. Explain briefly why the text of the algorithm and its proof of correctness are enough to clear you, by making clear the role of the precondition and of the postcondition.