B Exercises 3
Generalised Substitutions and Proof Obligations

The objective of this set of tutorial exercises is to use the semantics of generalised substitutions to compute proof obligations for operations.

The proof obligation for maintaining the machine invariant for the operation:

\[ \text{result} \leftarrow Op(\text{args}) \triangleq \text{PRE P THEN G END} \]

is \( I \land P \Rightarrow [G] I \)

where \( I \) is the state invariant.

Remember that substitution distributes through conjunction, that is

\[ [G] (R_1 \land R_2) = [G] R_1 \land [G] R_2 \]

This allows the computation of separate proof obligations for each conjunct, rather than one larger proof obligation.

1. The Simple machine
   (a) Calculate the proof obligations for each operation of the simple machine.
   (b) Remove the precondition of the Decrement operation and re-compute the proof obligation.
   Comment on the result.

2. The Bank machine. Compute the preconditions of the Deposit WithDraw and Balance operations.
   Note very carefully that
   \[ f(x) := y \triangleq f := f \circ \{ x \mapsto y \} \]

3. Assuming that varA and varB are both integers show that
   \[ \text{varA} := \text{varB} - \text{varA} ; \text{varB} := \text{varB} - \text{varA} ; \text{varA} := \text{varA} + \text{varB} \]
   is equivalent to
   \[ \text{varA, varB} := \text{varB, varA} \]

4. Traffic lights Compute the proof obligations for the operations of the simple SimpleTwoWay traffic light machine.

5. Examples from Wordsworth
   (a) Section 2.6: exercises 2.3 and 2.4.
   (b) Section 3.5: exercises 3.6, 3.7 and 3.8.
MACHINE Simple
VARIABLES num
INVARIANT num ∈ N
INITIALISATION num := 0

OPERATIONS
Set (val) ≜
  PRE val ∈ N
  THEN num := val
  END ;
val ← Get ≜
  BEGIN val := num END ;
Increment ≜
  BEGIN num := num + 1 END ;
Decrement ≜
  PRE 1 ≤ num
  THEN num := num - 1
  END

END
MACHINE Bank (maxaccount)

CONSTRAINTS

maxaccount ∈ N

SETS

ACCOUNT

PROPERTIES

card (ACCOUNT) = maxaccount

VARIABLES

accounts , balance

INARIANT

accounts ⊆ ACCOUNT ∧ balance ∈ accounts → N

INITIALISATION

accounts , balance := {}, {}

OPERATIONS

account ←− NewAccount ≡

PRE accounts ▦ ACCOUNT

THEN

ANY acc

WHERE acc ∈ ACCOUNT - accounts

THEN account := acc ||

accounts := accounts ∪ { acc } ||

balance (acc) := 0

END

END;

Deposit (account, amount) ≡

PRE account ∈ accounts ∧ amount ∈ N

THEN balance (account) := balance (account) + amount

END;

WithDraw (account, amount) ≡

PRE account ∈ accounts ∧ amount ∈ N ∧ amount ≤ balance (account)

THEN balance (account) := balance (account) - amount

END;

bal ←− Balance (account) ≡

PRE account ∈ accounts

THEN bal := balance (account)

END;

holdings ←− Holdings ≡

BEGIN

holdings := ∑ account . (account ∈ accounts | balance (account))

END

END
MACHINE SimpleTwoWay

SETS
   DIRECTION = { NorthSouth, EastWest } ;
   LIGHT = { Red, Green, Amber }

VARIABLES
   lights

INVARIANT
   lights ∈ DIRECTION → LIGHT ∧
   ( lights ( NorthSouth ) ∈ { Green, Amber } ⇒ lights ( EastWest ) = Red ) ∧
   ( lights ( EastWest ) ∈ { Green, Amber } ⇒ lights ( NorthSouth ) = Red )

INITIALISATION
   lights := { NorthSouth ↦ Red, EastWest ↦ Red }

OPERATIONS
   ToRed ( dir ) ≜
      PRE dir ∈ DIRECTION ∧ lights ( dir ) = Amber
      THEN lights ( dir ) := Red
      END ;
   ToGreen ( dir ) ≜
      PRE dir ∈ DIRECTION ∧ lights ( dir ) = Red ∧
      ( dir = NorthSouth ⇒ lights ( EastWest ) = Red ) ∧
      ( dir = EastWest ⇒ lights ( NorthSouth ) = Red )
      THEN lights ( dir ) := Green
      END ;
   ToAmber ( dir ) ≜
      PRE dir ∈ DIRECTION ∧ lights ( dir ) = Green
      THEN lights ( dir ) := Amber
      END

END