Reasoning About Programs

Week 8 PMT - Loops
To discuss during PMT - do NOT hand in

Sophia Drossopoulou and Mark Wheelhouse

Learning Aims:

A1: practice formal reasoning about loops

A2: reason about loops where the index decreases

A3: reason about conditional statements

A4: design invariants given some code

A5: practice working with predicates tailored for the problem domain

1st Question:

Consider the following Java method:

```
int product (int[] a)
// PRE: a ≠ null                                                  (P)
// POST: r = ∏ a_0[0..a_0.length)                                 (Q)
{
    int res = 1;
    int i = 0;
    // INV: a ≈ a_0 ∧ 0 ≤ i ≤ a.length ∧ res = ∏ a[0..i)            (I)
    // VAR: a.length − i                                          (V)
    while (i < a.length) {
        res = res * a[i];
        ++i;
    }
    // MID: a ≈ a_0 ∧ res = ∏ a[0..a.length)                       (M)
    return res;
}
```

where:

$$\prod \mathtt{a}[x..y) \;=\; \prod_{k=x}^{y-1} \mathtt{a}[k] \;=\; \mathtt{a}[x] * \mathtt{a}[x+1] * \ldots * \mathtt{a}[y-1]$$

and if $m < n$ then $\prod_{x=n}^{m} f(x) \;=\; 1$ by definition.

Show that:

a) The initialization code (lines 5 and 6) establishes the invariant I.

b) The loop re-establishes the invariant I.

c) The mid-condition M holds immediately after the end of the loop.

d) The method satisfies its postcondition Q.

e) The method terminates.

For each question, state clearly what is given and what needs to be shown.

A possible answer:

a) I is established by the initialization code.

Given:

(1)	$\mathtt{a}_0 \neq \mathtt{null}$	from PRE
(2)	$\mathtt{res} = 1$	from code line 5
(3)	$\mathtt{i} = 0$	from code line 6
(4)	$\mathtt{a} \approx \mathtt{a}_0$	implicit from code

To show:

(α)	$\mathtt{a} \approx \mathtt{a}_0$	INV
(β)	$0 \leq \mathtt{i} \leq \mathtt{a.length}$	INV
(γ)	$\mathtt{res} = \prod \mathtt{a}[0..\mathtt{i})$	INV

Proof:

(α) follows directly from (4)

(5)	$0 \leq 0 \leq \mathtt{a.length}$	from (1)

(β) follows from (5) and (3)

(6)	$\prod \mathtt{a}[0..0) = 1$	from def. $\prod$
(7)	$\prod \mathtt{a}[0..\mathtt{i}) = 1$	from (6) and (3)

(γ) follows from (7) and (2)

b) The loop re-establishes the invariant.

Given:

(1)	$\texttt{a} \approx \texttt{a}_0$	from `INV`
(2)	$0 \leq \texttt{i} \leq \texttt{a.length}$	from `INV`
(3)	$\texttt{res} = \prod \texttt{a}[0..\texttt{i})$	from `INV`
(4)	$\texttt{i} < \texttt{a.length}$	loop condition
(5)	$\texttt{res}' = \texttt{res} * \texttt{a[i]}$	from code line 10
(6)	$\texttt{i}' = \texttt{i} + 1$	from code line 11
(7)	$\texttt{a}' \approx \texttt{a}$	implicit from code

To show:

(α)	$\texttt{a}' \approx \texttt{a}_0$	`INV`
(β)	$0 \leq \texttt{i}' \leq \texttt{a}'\texttt{.length}$	`INV`
(γ)	$\texttt{res}' = \prod \texttt{a}'[0..\texttt{i}')$	`INV`

Proof:

(α) follows directly from (1) and (7)

(8)	$0 \leq \texttt{i} < \texttt{a.length}$	from (2) and (4)
(9)	$0 \leq \texttt{i} < \texttt{a}'\texttt{.length}$	from (8) and (7)
(10)	$0 \leq \texttt{i} + 1 \leq \texttt{a}'\texttt{.length}$	from (9)

(β) follows from (10) and (6)

(11)	$\texttt{res}' = \prod \texttt{a}[0..\texttt{i}) * \texttt{a[i]}$	from (3) and (5)
(12)	$\texttt{res}' = \prod \texttt{a}'[0..\texttt{i}) * \texttt{a}'\texttt{[i]}$	from (11) and (7)
(13)	$\texttt{res}' = \prod \texttt{a}'[0..(\texttt{i}+1))$	from (12) and def. of $\prod$

(γ) follows from (13) and (6)

c) M holds immediately after the end of the loop

Given:

(1)	$\texttt{a} \approx \texttt{a}_0$	from `INV`
(2)	$0 \leq \texttt{i} \leq \texttt{a.length}$	from `INV`
(3)	$\texttt{res} = \prod \texttt{a}[0..\texttt{i})$	from `INV`
(4)	$\texttt{i} \geq \texttt{a.length}$	negation of loop condition

To show:

(α)	$\texttt{a} \approx \texttt{a}_0$	`MID`
(β)	$\texttt{res} = \prod \texttt{a}[0..\texttt{a.length})$	`MID`

Proof:

(α) follows directly from (1)

(5)	$\texttt{i} = \texttt{a.length}$	from (2) and (4)

(β) follows from (3) and (5)

d) From (a), (b) and (c) we know that M holds at the end of the loop. The `return` statement tells us that $\mathtt{r} = \mathtt{res}$, and thus we obtain the postcondition Q.

Given:

(1)	$\mathtt{a} \approx \mathtt{a}_0$	from `MID`
(2)	$\mathtt{res} = \prod \mathtt{a}[0..\mathtt{a.length})$	from `MID`
(3)	$\mathtt{r} = \mathtt{res}$	from code line 14

To show:

(α)	$\mathtt{r} = \prod \mathtt{a}_0[0..\mathtt{a}_0\mathtt{.length})$	`POST`

Proof:

(4)	$\mathtt{res} = \prod \mathtt{a}_0[0..\mathtt{a}_0\mathtt{.length})$	from (2) and (1)

(α) follows from (4) and (3)

e) To show termination of the method it is sufficient to show that the loop terminates (as straight line code always terminates). Thus, we will show that:

(α) The variant is bounded.
(β) The variant decreases after every loop iteration.

Given:

(1)	$\mathtt{a} \approx \mathtt{a}_0$	from `INV`
(2)	$0 \le \mathtt{i} \le \mathtt{a.length}$	from `INV`
(3)	$\mathtt{res} = \prod_{j=0}^{\mathtt{i}-1} \mathtt{a[}j\mathtt{]}$	from `INV`
(4)	$\mathtt{i} < \mathtt{a.length}$	loop condition
(5)	$\mathtt{i}' = \mathtt{i} + 1$	from code line 11
(6)	$\mathtt{a}' \approx \mathtt{a}$	implicit from code

To show:

(α) $\mathtt{a.length} - \mathtt{i} \ge 0$
(β) $\mathtt{a}'\mathtt{.length} - \mathtt{i}' < \mathtt{a.length} - \mathtt{i}$

Proof:

(7)	$0 \le \mathtt{i} < \mathtt{a.length}$	from (2) and (4)

(α) follows from (7)

(8)	$\mathtt{a.length} - (\mathtt{i} + 1) < \mathtt{a.length} - \mathtt{i}$	by definition
(9)	$\mathtt{a}'\mathtt{.length} - (\mathtt{i} + 1) < \mathtt{a.length} - \mathtt{i}$	from (8) and (6)

(β) follows from (9) and (5)

2nd Question:

The function eqs calculates the number of equal elements for arrays `a` and `b`, provided they have the same length:

$$eqs(\mathtt{a},\mathtt{b}) = \begin{cases} \mid \{\, j \mid 0 \le j < \mathtt{a.length} \;\wedge\; \mathtt{a}[j] = \mathtt{b}[j] \,\} \mid & \text{if } \mathtt{a.length} = \mathtt{b.length} \\ \text{undefined} & \text{otherwise} \end{cases}$$

As an example, $eqs(\text{‘}\mathtt{DEFGH}\text{’}, \text{‘}\mathtt{DXFXH}\text{’}) = 3$.

Consider the following Java method that claims to perform the same calculation:

```
int eqNo (int[] a, int[] b)
// PRE:  a ≠ null  ∧  b ≠ null  ∧  a.length = b.length
// POST: r = eqs(a₀, b₀)
{
    int res = 0;
    int i = a.length;
    // INV: I
    // VAR: V
    while (i > 0) {
        i--;
        if (a[i] == b[i]) { res++; }
    }
    // MID: M
    return res;
}
```

a) Complete the specification of the `eqNo` method.
   i) Write a midcondition M which holds after the loop.
   ii) Write a loop invariant I.
   iii) Write a loop variant V.

b) Prove that the loop invariant I is established before entering the loop.

c) Prove that the loop re-establishes the loop invariant I.

d) Prove that the mid-condition M holds immediately after the termination of the loop.

e) Prove that `eqNo` is partially correct.

f) Prove that `eqNo` terminates.

Hint: You may find it helpful to use the function $eqsAux$ which calculates the numbers of equal elements between index k and the end of the array for arrays `a` and `b`:

$$eqsAux(\mathtt{a},\mathtt{b},k) = \begin{cases} \mid \{ j \mid k \le j < \mathtt{a.length} \wedge \mathtt{a}[j] = \mathtt{b}[j]\} & \text{if } \mathtt{a.length} = \mathtt{b.length} \text{ and } k \ge 0 \\ \text{undefined} & \text{otherwise} \end{cases}$$

For example, $eqsAux(\text{‘}\mathtt{DEFGH}\text{’}, \text{‘}\mathtt{DXFXH}\text{’}, 4) = 1$, and also $eqsAux(\text{‘}\mathtt{DEFGH}\text{’}, \text{‘}\mathtt{DXFXH}\text{’}, 2) = 2$, and finally, $eqsAux(\text{‘}\mathtt{DEFGH}\text{’}, \text{‘}\mathtt{DXFXH}\text{’}, 0) = 3$.

Observe that $k \le k'$ implies that $eqsAux(\mathtt{a},\mathtt{b},k') \le eqsAux(\mathtt{a},\mathtt{b},k)$ for any `a` and `b`.
Also, $eqsAux(\mathtt{a},\mathtt{b},0) = eqs(\mathtt{a},\mathtt{b})$ for any `a` and `b`.

A possible answer:

a) i) $M \equiv \mathtt{a} \approx \mathtt{a}_0 \wedge \mathtt{b} \approx \mathtt{b}_0 \wedge \mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, 0)$

ii) $I \equiv \mathtt{a} \approx \mathtt{a}_0 \wedge \mathtt{b} \approx \mathtt{b}_0 \wedge 0 \leq \mathtt{i} \leq \mathtt{a.length} \wedge \mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i})$

iii) $V \equiv \mathtt{i}$

b) We prove that the loop invariant is established before entering the loop:

Given:

(1)	$\mathtt{a}_0 \neq \mathtt{null}$	from PRE
(2)	$\mathtt{b}_0 \neq \mathtt{null}$	from PRE
(3)	$\mathtt{a}_0\mathtt{.length} = \mathtt{b}_0\mathtt{.length}$	from PRE
(4)	$\mathtt{res} = 0$	from code, line 5
(5)	$\mathtt{i} = \mathtt{a.length}$	from code, line 6
(6)	$\mathtt{a} \approx \mathtt{a}_0$	implicit from code
(7)	$\mathtt{b} \approx \mathtt{b}_0$	implicit from code

To show:

(α)	$\mathtt{a} \approx \mathtt{a}_0$	INV
(β)	$\mathtt{b} \approx \mathtt{b}_0$	INV
(γ)	$0 \leq \mathtt{i} \leq \mathtt{a.length}$	INV
(δ)	$\mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i})$	INV

Proof:

(α) follows directly from (6)

(β) follows directly from (7)

(8) $0 \leq \mathtt{a.length} \leq \mathtt{a.length}$ — from (1) and (6)

(γ) follows from (8) and (5)

(9) $eqsAux(\mathtt{a}_0, \mathtt{b}_0, \mathtt{a}_0\mathtt{.length}) = 0$ — from (3) and def. $eqAux$

(10) $eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i}) = 0$ — from (9), (6), (7) and (5)

(δ) follows from (10) and (4)

c) We prove that the loop re-establishes the loop invariant:

Given:

(0)	$\mathtt{a}_0\mathtt{.length} = \mathtt{b}_0\mathtt{.length}$	from PRE
(1)	$\mathtt{a} \approx \mathtt{a}_0$	from INV
(2)	$\mathtt{b} \approx \mathtt{b}_0$	from INV
(3)	$\mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i})$	from INV
(4)	$0 \leq \mathtt{i} \leq \mathtt{a.length}$	from INV
(5)	$\mathtt{i} > 0$	loop condition
(6)	$\mathtt{i}' = \mathtt{i} - 1$	from code line 10
(7)	$\mathtt{a}' \approx \mathtt{a}$	implicit from code
(8)	$\mathtt{b}' \approx \mathtt{b}$	implicit from code

To show:

(α)	$\mathtt{a}' \approx \mathtt{a}_0$	INV
(β)	$\mathtt{b}' \approx \mathtt{b}_0$	INV
(γ)	$0 \leq \mathtt{i}' \leq \mathtt{a}'\mathtt{.length}$	INV
(δ)	$\mathtt{res}' = eqsAux(\mathtt{a}', \mathtt{b}', \mathtt{i}')$	INV

Proof:

(α) follows directly from (1) and (7)

(β) follows directly from (2) and (8)

(9)	$0 < \mathtt{i} \leq \mathtt{a.length}$	from (4) and (5)
(10)	$0 < \mathtt{i} \leq \mathtt{a}'\mathtt{.length}$	from (9) and (7)
(11)	$0 \leq \mathtt{i} - 1 \leq \mathtt{a}'\mathtt{.length}$	from (10)

(γ) follows from (11) and (6)

Case 1: $\mathtt{a[i']} = \mathtt{b[i']}$

(12)	$\mathtt{res}' = \mathtt{res} + 1$	from code, line 11 and case
(13)	$\mathtt{res}' = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i}) + 1$	from (12) and (3)
(14)	$\mathtt{res}' = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i} - 1)$	from (13), (0), case and *eqAux* definition
(15)	$\mathtt{res}' = eqsAux(\mathtt{a}', \mathtt{b}', \mathtt{i} - 1)$	from (14), (7) and (8)

Case 2: $\mathtt{a[i']} \neq \mathtt{b[i']}$

(16)	$\mathtt{res}' = \mathtt{res}$	from code, line 11 and case
(17)	$\mathtt{res}' = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i})$	from (16) and (3)
(18)	$\mathtt{res}' = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i} - 1)$	from (17), (0), case and *eqAux* definition
(19)	$\mathtt{res}' = eqsAux(\mathtt{a}', \mathtt{b}', \mathtt{i} - 1)$	from (18), (7) and (8)

(δ) follows from (15), (19) and (6)

d) We know that the loop invariant and the negation of the loop condition hold immediately at the end of the loop. Therefore, we need to do the following:

Given:

(1)	$\mathtt{a} \approx \mathtt{a}_0$	from INV
(2)	$\mathtt{b} \approx \mathtt{b}_0$	from INV
(3)	$\mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, \mathtt{i})$	from INV
(4)	$0 \leq \mathtt{i} \leq \mathtt{a.length}$	from INV
(5)	$\mathtt{i} \leq 0$	negated loop condition

To show:

(α)	$\mathtt{a} \approx \mathtt{a}_0$	MID
(β)	$\mathtt{b} \approx \mathtt{b}_0$	MID
(γ)	$\mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, 0)$	MID

Proof:

(α) follows directly from (1)

(β) follows directly from (2)

(6) $\mathtt{i} = 0$ from (4) and (5)

(γ) follows from (3) and (6)

e) We show that the program is partially correct:

Given:

(1)	$\mathtt{a} \approx \mathtt{a}_0$	from MID
(2)	$\mathtt{b} \approx \mathtt{b}_0$	from MID
(3)	$\mathtt{res} = eqsAux(\mathtt{a}, \mathtt{b}, 0)$	from MID
(4)	$\mathtt{r} = \mathtt{res}$	from code line 14
(5)	$\forall \mathtt{a}, \mathtt{b}.\, eqsAux(\mathtt{a}, \mathtt{b}, 0) = eqs(\mathtt{a}, \mathtt{b})$	*Lemma*

To show:

(α)	$\mathtt{r} = eqs(\mathtt{a}_0, \mathtt{b}_0)$	POST

Proof:

(6)	$\mathtt{res} = eqs(\mathtt{a}, \mathtt{b})$	from (3) and *Lemma*
(7)	$\mathtt{res} = eqs(\mathtt{a}_0, \mathtt{b}_0)$	from (6), (1) and (2)

(α) follows from (7) and (4)

f) We need to show that the loop terminates. For this, we need to show that:

  (α) The variant is bounded.

  (β) The variant decreases after every loop iteration.

Given:

(1)	$\texttt{a} \approx \texttt{a}_0$	from $\texttt{INV}$
(2)	$\texttt{b} \approx \texttt{b}_0$	from $\texttt{INV}$
(3)	$0 \leq \texttt{i} \leq \texttt{a.length}$	from $\texttt{INV}$
(4)	$\texttt{res} = eqsAux(\texttt{a}, \texttt{b}, \texttt{i})$	$\texttt{INV}$
(5)	$\texttt{i} > 0$	loop condition
(6)	$\texttt{i}' = \texttt{i} - 1$	from code line 10

To show:

(α) $\texttt{i} \geq 0$

(β) $\texttt{i}' < \texttt{i}$

Proof:

(α) follows directly from (5)

(7)	$\texttt{i} - 1 < \texttt{i}$	by definition

(β) follows from (7) and (6)