Loop-invariants: Tutorial sheet 5

Jeremy Bradley

7 February 2005

1. Given the following method which generates the product of an array of integers:

```
        int product (int a []) {
            // pre: none
[1]         int res = 1;
[2]         int i = 0;
[L]         while (i < a.length) {
[3]             res = res * a[i];
[4]             ++i;
                // loop invariant: ?
            }
            return res;
        }
```

   The postcondition for `product` is:

   - $a = a_0$

     $\wedge$ $(a.length = 0) \rightarrow res = 1$

     $\wedge$ $(a.length > 0) \rightarrow res = \prod_{j=0}^{a.length-1} a[j]$

   (a) Assuming $a.length = 0$ show that pre-condition $\vdash$ post-condition

   (b) Construct a loop variant for the loop in `product`

   (c) Construct a loop invariant for the position marked in the code

   (d) Assuming $a.length > 0$, show that the invariant is established initially

   (e) Show that the invariant is re-established from the kth iteration of the loop to the $(k+1)$th iteration

2. The following method checks an array of integers to see if they are sorted into descending order.

```
        boolean isSorted (int [] a) {
            // pre: none
[1]         boolean res = true;
[2]         int i = 0;
[L]         while (i < a.length-1) {
                // invariant: <here>
                if (a[i] < a[i+1])
[3]                 res = false;
[4]             ++i;
            }
            return res;
        }
```

Show that the loop invariant below is established initially and re-establishes itself during iteration:

$$(a = a_0) \wedge (0 < i_k + 1 < a.length) \wedge \left(res_k = \bigwedge_{j=0}^{i_k - 1} (a[j] \geq a[j+1]) \right)$$

[You may assume that $\bigwedge_{n=0}^{m} P(n) = true$ for $m < 0$]

3. The method `newfind` does not require that the element x exists in the array a. It either returns the a-index that corresponds to x or it returns $a.length$ if x cannot be found.

```
       int newfind (int x, int a []) {
           // pre: none
[1]        int i = 0;
[2]        int res = 0;

[L]        while ( (i < a.length) && (a[i] != x) ) {
               // invariant: ?
[3]            i++;
           }
[4]        res = i;
           return res;
       }
```

Given that the post-condition of `newfind` is:

- $a = a_0$

$\wedge$ $(0 \leq res \leq a.length)$

$\wedge$ $(0 \leq res < a.length) \rightarrow (a[res] = x)$

$\wedge$ $(res = a.length) \rightarrow ((0 \leq j < a.length) \rightarrow a[j] \neq x)$

(a) Write down a loop variant for the loop

(b) Construct the kth loop invariant at the position shown in the code

(c) Show that the loop is re-established in the $(k+1)$th invariant