Use of Pre- and Postconditions

Peter Müller

August 26, 1998

Using Conditions to Proove Correctness

Pre- and postconditions can be used to proove algorithms for correctness. I will show this with a small example. Since the course is no course on formal programming theory, this will only give you a first glance on this subject. However, since abstract data types are so important for object-oriented programming, I thought it useful to introduce them.

Consider the following code fragment which calculates the sum of the first 10 whole numbers:

```
INTEGER i;
INTEGER n;
INTEGER s;

n ← 10;
i ← 1;
s ← 0;
WHILE i ≤ n DO
   s ← s + i;
   i ← i + 1;
END
```

We will now show with pre- and postconditions that this loop really calculates the sum. However, I will only give and explain those conditions which are of interest here. Just before we enter the loop, the following precondition hold:

$$\left\{ s = 0 = \sum_{j=1}^{i-1} j \quad \text{and} \quad i \leq n \right\} \tag{1}$$

I use the notation $\sum_{i=1}^{n} i$ to denote the sum over all numbers from 1 to n. Note that $\sum_{i=1}^{0} i$ is defined to be 0. (We use this in the above loop precondition!)

Since the loop condition holds (i≤n) we can enter the loop. Until now, we have:

```
INTEGER i;
INTEGER n;
INTEGER s;

n ← 10;
i ← 1;
s ← 0;
{s = 0 = ∑_{j=1}^{i−1} j   and   i ≤ n}
WHILE i ≤ n DO
   {s = ∑_{j=1}^{i−1} j   and   i ≤ n}
   s ← s + i;
   i ← i + 1;
END
```

The next statement is the increment of s by the value of i. Using the precondition, this would yield to

$$\left\{ s = \sum_{j=1}^{i-1} j \; + \; i \quad \text{and} \quad i \leq n \right\} \tag{2}$$

This can be simplified to:

$$\left\{ s = \sum_{j=1}^{i} j \quad \text{and} \quad i \leq n \right\} \tag{3}$$

The next line increments i by 1. What does this mean to the current precondition? Well, just that the value of i has changed! As a consequence, every occurrence of i must be adopted to reflect this change, leading to the following postcondition of the i assignment:

$$\left\{ s = \sum_{j=1}^{i-1} j \quad \text{and} \quad i - 1 \leq n \right\} \tag{4}$$

The second part of this condition can be rewritten as:

$$\begin{aligned} i - 1 \leq n \quad &\Rightarrow \quad i \leq n + 1 \\ &\Rightarrow \quad i < n + 1 \quad \text{or} \quad i = n + 1 \\ &\Rightarrow \quad i \leq n \quad \text{or} \quad i = n + 1 \end{aligned}$$

Note that the first term (i≤n) is exactly the loop condition. If this term evaluates to true, the body is executed again. However, the conditions hold *each time the loop body is executed. They are* ***invariant*** *regarding the execution.*

Only if the second part is true, the loop terminates. So after the loop the following condition holds:

$$\left\{ s = \sum_{j=1}^{i-1} j \quad \text{and} \quad i = n + 1 \right\} \tag{5}$$

From this, we can conclude:

$$s = \sum_{j=1}^{i-1} j \quad \text{and} \quad i = n + 1 \quad \Rightarrow \quad s = \sum_{j=1}^{n} j$$

With the fact that n equals 10 we have shown, that the loop really do what we have expected.

Finally, the whole loop with all conditions is shown:

INTEGER i;
INTEGER n;
INTEGER s;

n ← 10;
i ← 1;
s ← 0;
$\left\{ s = 0 = \sum_{j=1}^{i-1} j \quad \text{and} \quad i \leq n \right\}$
WHILE i ≤ n **DO**
  $\left\{ s = \sum_{j=1}^{i-1} j \quad \text{and} \quad i \leq n \right\}$
  s ← s + i;
  $\left\{ s = \sum_{j=1}^{i-1} j \; + \; i \; = \; \sum_{j=1}^{i} j \quad \text{and} \quad i \leq n \right\}$
  i ← i + 1;
  $\left\{ s = \sum_{j=1}^{i-1} j \quad \text{and} \quad (i \leq n \quad \text{or} \quad i = n + 1) \right\}$
END
$\left\{ s = \sum_{j=1}^{i-1} j \quad \text{and} \quad i = n + 1 \Rightarrow s = \sum_{j=1}^{n} j \right\}$