

Activity: Proof of a Simple Loop

A. Why?

Practice makes perfect.

B. Outcomes

By the end of the activity you should

- Be able to create initialization code for an invariant.
- Be able to create a loop test given an invariant and final postcondition.
- Be able to create the precondition needed by a progress step for a loop.

C. Questions

1. Suppose we want to use a loop to sum integers but in a downward direction. The program specification is unchanged: $\{n \geq 0\} \text{ Program } \{s = \text{sum}(0, n)\}$. The loop invariant will be different, say $r \equiv 0 \leq j \leq n \wedge s = \text{sum}(j, n)$, and we'll loop until $j=0$.

```

{ n ≥ 0 } Initialization;
{ inv r ≡ 0 ≤ j ≤ n ∧ s = sum(j, n) }
while 0 < j do { p ∧ 0 < j } Loop Body { p }
od
{ p ∧ 0 ≥ j }
{ s = sum(0, n) }

```

- (1a) Write and verify some initialization code for this program: Find expressions e_1 and e_2 such that $\{n \geq 0\} \{r[s := e_2][j := e_1]\} j := e_1; \{r[s := e_2]\} s := e_2 \{r\}$. Calculate the substitutions $r[s := e_2]$ and $r[s := e_2][j := e_1]$. Is the predicate logic obligation $n \geq 0 \rightarrow r[s := e_2][j := e_1]$ valid? (I.e., true for all states?)

```

{ n ≥ 0 } { r[s := n][j := n] } j := n; { r[s := n] } s := n { r }
r[s := n] ≡ (0 ≤ j ≤ n ∧ s = sum(j, n)) [s := n] ≡ 0 ≤ j ≤ n ∧ n = sum(j, n)
r[s := n][j := n] ≡ (0 ≤ j ≤ n ∧ n = sum(j, n)) [j := n] ≡ (0 ≤ n ≤ n ∧ n = sum(n, n))

```

Yes, the predicate logic obligation is valid.

Suggested answer: $\{r[s := 0][j := n]\} j := n; \{r[s := 0]\} s := 0 \{r\}$
 $r[s := 0][j := n] \equiv 0 \leq n \leq n \wedge 0 = \text{sum}(n, n)$ [i.e., we need $n = 0$]

Does $n \geq 0 \rightarrow n = 0$? No, so this suggested answer has a bug.

- (1b) Say we want to initialize $s = n$. Find expressions e_3 and e_4 such that $\{n \geq e_3\} \{r[s := n][j := e_4]\} j := e_4; \{r[s := n]\} s := n \{r\}$.

Calculate the substitutions. What is the predicate logic obligation, and is it valid?

This is like part (1a): We need $e_3 \equiv 0$ and $e_4 \equiv n$.

(1c) Say we use $j := j-1$ as our progress step. Find an expression e_5 such that

$$\{r \wedge 0 < j\} \{r[j := j-1][s := e_5]\} s := e_5; \{r[j := j-1]\} j := j-1 \{r\}$$

Calculate the substitutions. What is the predicate logic obligation, and is it valid?

Suggested answer is $s := s+j; j := j-1$.

$$\{r \wedge 0 < j\} \{r[j := j-1][s := s+j]\} s := s+j; \{r[j := j-1]\} j := j-1 \{r\}$$

$$r[j := j-1] \equiv 0 \leq j-1 \leq n \wedge s = \text{sum}(j-1, n)$$

$$r[j := j-1][s := s+j] \equiv 0 \leq j-1 \leq n \wedge s+j = \text{sum}(j-1, n)$$

Does $r \wedge 0 < j \rightarrow (0 \leq j-1 \leq n \wedge s+j = \text{sum}(j-1, n))$?

$$r \equiv 0 \leq j \leq n \wedge s = \text{sum}(j, n)$$

$$r \wedge 0 < j \leftrightarrow 0 < j \leq n \wedge s = \text{sum}(j, n)$$

$0 < j$ implies $0 \leq j-1$

$$s = \text{sum}(j, n)$$

$$s+(j-1) = j-1 + \text{sum}(j, n) = \text{sum}(j-1, n)$$

So we need $s := s+(j-1); j := j-1$.