Loop Invariant Proofs

April 11, 2025

Green Proof: `evens(lst)`

Consider the function:

```
def evens(lst):
    acc = []
    i = 0
    while i < len(lst):
        if lst[i] % 2 == 0:
            acc.append(lst[i])
        i = i + 1
    return acc
```

Theorem 1. *For any list of integers* `lst`*,*

$$\forall x \in \mathbb{Z}, \quad x \in \texttt{evens(lst)} \iff \Big(x \in \texttt{lst} \text{ and } x \text{ is even}\Big).$$

Proof. Let `lst` be an arbitrary list of integers. We prove that `evens(lst)` returns exactly the even elements of `lst` by establishing the following loop invariant.

Loop Invariant. Define, for the index i at the start of the nth iteration of the while loop,

$$R(n): \quad \forall x \in \mathbb{Z}, \quad x \in \texttt{acc} \iff \Big(x \in \texttt{lst}[0:i] \text{ and } x \text{ is even}\Big).$$

Initialization. Prior to the first iteration, the program sets `acc = []` and `i = 0`. Since the sublist `lst[0:0]` is empty, we have that for every integer x both $x \in \texttt{acc}$ and $x \in \texttt{lst}[0:0]$ are false. Therefore, $R(0)$ holds.

Maintenance. Assume that $R(n)$ holds before an iteration of the loop in which $i <$ `len(lst)`. Let $a = \texttt{lst}[i]$. Two cases arise:

- **Case 1:** a is even.
  In this case, the conditional in the loop is true, so a is appended to `acc`. After appending, we have that `acc` now contains all even numbers in `lst`[0:i] plus a, which is exactly the set of even numbers in `lst`[0:i+1]. Then the program increments i to $i + 1$. Hence, the invariant holds for the next iteration.

- **Case 2:** a is odd.
  In this case, the conditional is false, so `acc` remains unchanged. Upon incrementing i to $i + 1$, note that the even numbers in `lst`[0:i+1] remain the same as those in `lst`[0:i] (since a is not even). Thus, $R(n + 1)$ still holds.

Termination. The loop terminates when $i = \texttt{len(lst)}$. At this point, by the invariant $R(n)$, we have

$$\forall x \in \mathbb{Z}, \quad x \in \texttt{acc} \iff \Big(x \in \texttt{lst}[0 : \texttt{len(lst)}] \text{ and } x \text{ is even}\Big).$$

Since `lst[0:len(lst)]` is simply `lst`, the returned list `acc` is exactly the set of even numbers in `lst`. This completes the proof. □

Blue Proof: `dbl(lst)`

Consider the function:

```
def dbl(lst):
    i = 0
    while i < len(lst):
        lst[i] = 2 * lst[i]
        i = i + 1
```

Let `lst_0` denote the original list (i.e., the input to the function).

Theorem 2. *For any list `lst`, after executing `dbl(lst)`, we have*

$$\forall j,\ 0 \leq j < \texttt{len(lst)}, \quad \texttt{lst}[j] = 2 \times \texttt{lst\_0}[j].$$

Proof. We prove correctness by establishing the following loop invariant.

Loop Invariant. Let i be the index at the start of the nth iteration. Then:

$$P(n): \quad \forall j\ (0 \leq j < i) \quad \texttt{lst}[j] = 2 \times \texttt{lst\_0}[j],$$

$$\forall j\ (i \leq j < \texttt{len(lst)}) \quad \texttt{lst}[j] = \texttt{lst\_0}[j].$$

Initialization. Initially, $i = 0$ and no changes have been made to `lst`; hence, for every index j, $\texttt{lst}[\text{j}] = \texttt{lst\_0}[\text{j}]$. The first condition holds vacuously (since no index satisfies $0 \leq j < 0$), and the second condition holds for all indices. Thus, $P(0)$ is true.

Maintenance. Assume that before a given iteration with $i < \texttt{len(lst)}$, the invariant $P(n)$ holds. Let the current index be i and denote $a = \texttt{lst}[i]$ (which, by the invariant, equals $\texttt{lst\_0}[i]$). The loop then updates:

$$\texttt{lst}[i] := 2 \times \texttt{lst}[i] = 2 \times \texttt{lst\_0}[i].$$

After this update, the first part of the invariant holds for index i (and for all previous indices by the inductive hypothesis), and the remaining elements (for indices j such that $i + 1 \leq j < \texttt{len(lst)}$) remain unchanged. When i is incremented to $i + 1$, the invariant $P(n + 1)$ holds.

Termination. The loop terminates when $i = \texttt{len(lst)}$. At termination, the invariant guarantees that for every index j with $0 \leq j < \texttt{len(lst)}$, we have

$$\texttt{lst}[j] = 2 \times \texttt{lst\_0}[j],$$

which is precisely the desired postcondition. □

Conclusion

Using loop invariants and following the initialization, maintenance, and termination steps as in the red team's proof, we have demonstrated that:

1. In `evens(lst)`, the accumulator `acc` contains exactly the even numbers from `lst`.
2. In `dbl(lst)`, every element of the list is replaced with twice its original value.