EECS3311 Software Design
Fall 2019
Exercise: Proving Correctness of Loops

Chen-Wei Wang

Consider the following query which involves the use of a loop to find the maximum value from an integer array:

```
find_max (a: ARRAY [INTEGER]): INTEGER
    require
        not_empty: a.count > 0
    local
        i: INTEGER
    do
        from
            i := a.lower
            Result := a [i]
        invariant
            -- Predicate Equivalent: ∀j | a.lower ≤ j < i • Result ≥ a[j]
            across
                a.lower |..| (i − 1) as j
            all
                Result >= a [j.item]
            end
        until
            i > a.upper
        loop
            if a [i] > Result then
                Result := a [i]
            end
            i := i + 1
        variant
            a.upper − i + 1
        end
    ensure
        -- Predicate Equivalent: ∀j | a.lower ≤ j ≤ a.upper • Result ≥ a[j]
        across
            a.lower |..| a.upper as j
        all
            Result >= a [j.item]
        end
    end
```

Your Tasks

Prove or disprove that the above program is *totally* correct, which involves the following steps:

1. State formally, in terms of Hoare Triples, the obligations for proving that:
   - The loop is *partially* correct (without considering termination); and
   - The loop terminates.
2. For each of the Hoare triple $\{ \ Q \ \} \ S \ \{ \ R \ \}$ in Step 1, calculate the corresponding weakest precondition (i.e., $wp \ (S, \ R)$).
3. Prove or disprove that the calculated wp is equal to or weaker than the corresponding precondition (i.e., prove or disprove that $Q \Rightarrow wp \ (S, \ R)$).

1 Partial Correctness

1.1 Establishing the Loop Invariant

Proof Obligation:

$$\{ \ a.count > 0 \ \}$$

```
i := a.lower; Result := a[i]
```

$$\{ \ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ \}$$

Notice that the augmented constraint $\boxed{a.lower \leq j \wedge j \leq a.upper}$ is due to the array indexing expression `a[j]`. Similar augmentation is performed for each occurrence of an array indexing expression.

1.2 Maintaining the Loop Invariant

Proof Obligation:

$$\{ \ \neg(i > a.upper) \wedge (\ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ ) \ \}$$

```
if a[i] > Result then Result := a[i] end; i := i + 1
```

$$\{ \ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ \}$$

1.3 Establishing the Postcondition

Proof Obligation:

$$(i > a.upper) \wedge (\ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ )$$
$$\Rightarrow (\ \forall j \mid a.lower \leq j \leq a.upper \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ )$$

2 Termination

2.1 Loop Variant Stays Positive

Proof Obligation:

$$\{ \ \neg(i > a.upper) \wedge (\ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ ) \ \}$$

```
if a[i] > Result then Result := a[i] end; i := i + 1
```

$$\{ \ a.upper - i + 1 \geq 0 \ \}$$

2.2 Loop Variant Decreases

Proof Obligation:

$$\{ \ \neg(i > a.upper) \wedge (\ \forall j \mid a.lower \leq j \leq i - 1 \bullet \boxed{a.lower \leq j \wedge j \leq a.upper} \wedge \textbf{Result} \geq a[j] \ ) \ \}$$

```
if a[i] > Result then Result := a[i] end; i := i + 1
```

$$\{ \ a.upper - i + 1 < a.upper_0 - i_0 + 1 \ \}$$

Solution to Proving (1.2)

We first calculate the *wp* for the loop body to maintain the LI:

$$
\begin{array}{ll}
& wp(\texttt{if a[i] > Result then Result := a[i] end; i := i + 1}, \boxed{\forall j \mid a.lower \le j \le i-1 \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]}) \\
= & \{wp \texttt{ rule for seq. comp. }\} \\
& wp(\texttt{if a[i] > Result then Result := a[i] end}, \boxed{wp(\texttt{i := i + 1}, \boxed{\forall j \mid a.lower \le j \le i-1 \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]})}) \\
= & \{wp \texttt{ rule for assignment}\} \\
& wp(\texttt{if a[i] > Result then Result := a[i] end}, \boxed{\forall j \mid a.lower \le j \le \mathbf{i} \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]}) \\
= & \{wp \texttt{ rule for conditional}\} \\
& \quad a[i] > \textbf{Result} \implies wp(\texttt{Result := a[i]}, \boxed{\forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]}) \\
& \quad \wedge \\
& \quad a[i] \le \textbf{Result} \implies wp(\texttt{Result := Result}, \boxed{\forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]}) \\
= & \{wp \texttt{ rule for assignment, twice}\} \\
& \quad a[i] > \textbf{Result} \implies \forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge \mathbf{a[i]} \ge a[j] \\
& \quad \wedge \\
& \quad a[i] \le \textbf{Result} \implies \forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]
\end{array}
$$

We then prove that the precondition (i.e., $\neg$(exit condition) and LI) is no weaker than the above calculated *wp*:

- To prove:

$$
\begin{array}{l}
\neg(i > a.upper) \wedge (\ \forall j \mid a.lower \le j \le i-1 \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]\ ) \\
\quad \implies a[i] > \textbf{Result} \implies \boxed{\forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge a[i] \ge a[j]}
\end{array}
$$

  Proof:

$$
\begin{array}{ll}
& \boxed{\forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge a[i] \ge a[j]} \\
\equiv & \{\texttt{split range: } \forall j \mid a.lower \le j \le i \bullet P(j) \equiv (\forall j \mid a.lower \le j \le i-1 \bullet P(j)) \wedge P(i)\} \\
& (\forall j \mid a.lower \le j \le \mathbf{i\ \text{-}\ 1} \bullet a.lower \le j \wedge j \le a.upper \wedge a[i] \ge a[j]) \wedge (a.lower \le \mathbf{i} \wedge \mathbf{i} \le a.upper \wedge a[i] \ge a[\mathbf{i}]) \\
\equiv & \{\texttt{antecedent: } a[i] > \texttt{Result}\texttt{; and RHS of precond: } \forall j \mid a.lower \le j \le i-1 \bullet a.lower \le j \wedge j \le a.upper \wedge \texttt{Result} \ge a[j]\} \\
& true \wedge (a.lower \le i \wedge i \le a.upper \wedge a[i] \ge a[i]) \\
\equiv & \{\texttt{LHS of precond: } \neg(i > a.upper) \texttt{ and } a[i] \ge a[i] \equiv true\} \\
& true
\end{array}
$$

- To prove:

$$
\begin{array}{l}
\neg(i > a.upper) \wedge (\ \forall j \mid a.lower \le j \le i-1 \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]\ ) \\
\quad \implies a[i] \le \textbf{Result} \implies \forall j \mid a.lower \le j \le i \bullet a.lower \le j \wedge j \le a.upper \wedge \textbf{Result} \ge a[j]
\end{array}
$$

  (*Exercise*)