Extra Exercises on Hoare's Method

Binary Search[1]

Prove the partial correctness of:

```
binary_search:
begin
  i:=1; j:=n; found:=0;
  while i <= j do
    k:=(i+j) div 2;
    if y = a[k]
       then found:=1; i:=j+1;
       else if y < a[k]
              then j:=k-1;
              else i:=k+1;
            fi
    fi
  od
end
```

With respect to the precondition and postcondition:

$$\{Pre\} \equiv \{n \geq 1\}$$

$$\begin{array}{rcl} \{Post\} & \equiv & \{\exists\, h((1 \leq h \leq n \wedge y = a[h]) \Rightarrow found = 1) \\ & & \nexists h((1 \leq h \leq n \wedge y = a[k]) \Rightarrow found = 0)\} \end{array}$$

As usual, we split the proof into three substeps:

1. $\{Pre\}$ `i:=1; j:=n; found:=0;` $\{I\}$
2. $\{I\}$ `while ... od` $\{I \wedge i > j\}$
3. $\{I \wedge i > j\} \Rightarrow \{Post\}$

[1]Also on pg. 441 of textbook

Choice of loop invariant The loop invariant must say that if $found = 1$ there is an index h is such that $a[h] = y$; otherwise, if $found = 0$ then all elements in a before index i are less than y and all elements after index j are greater than y. Moreover, we notice that i is always less than or equal to $j+1$. All in all, the following is an acceptable loop invariant.

$$\begin{array}{lll} \{I\} & \equiv & (found = 1 \Rightarrow \exists h(1 \leq h \leq n \land y = a[h])) \\ & & \land\ (found = 0 \Rightarrow \forall h((1 \leq h \leq i-1) \Rightarrow a[h] < y) \\ & & \qquad\qquad\qquad\qquad \land\ \forall h((j+1 \leq h \leq n) \Rightarrow a[h] > y)) \\ & & \land\ i \leq j+1 \end{array}$$

Notice that throughout the whole correctness proof we assume that the array a is ordered. We don't have to check the invariance of this condition, since a is never modified, but we can use it whenever we need it.

Proof of step 1 By backward substitution of $\{I\}$ through `i:=1; j:=n; found:=0;` we get:

$$\begin{array}{lll} \{I'\} & \equiv & (0 = 1 \Rightarrow \ldots)\ \land\ (0 = 0 \Rightarrow \forall h((1 \leq h \leq 1-1) \Rightarrow a[h] < y) \\ & & \qquad\qquad\qquad\qquad \land\ \forall h((n+1 \leq h \leq n) \Rightarrow a[h] > y)) \\ & & \land\ 1 \leq n+1 \end{array}$$

which clearly reduces to $\{I'\} \equiv n \geq 0$, implied by the precondition $n \geq 1$.

Proof of step 2 As usual, we apply the inference rule for the while loop, thus reducing to proving $\{I \land i \leq j\}$ `/loop body/` $\{I\}$.

Now, let us first consider the outer `if`. Its `then` branch is dealt with by proving the following:

$\{I \land i \leq j \land y = a[(i+j)\texttt{div}\ 2]\}$
```
  found:=1; i:=j+1;
```
$\{I\}$

where we used the little "trick" to put the result of the assignment `k:=(i+j)div 2` directly in the annotation (as we did in a passage of the *bubble-sort* exercise). By backward substitution of I through these instructions, we get, after trivial simplifications:

$$\{J\} \quad \equiv \quad \exists h(1 \leq h \leq n \land y = a[h])$$

Clearly, $\{J\}$ is implied simply by $y = a[(i+j)\texttt{div}\ 2]$, as $1 \leq (i+j)\texttt{div}\ 2 \leq n$.

The `else` part is a bit more complicated, as it involves a nested `if`. Thus, we have to prove the following two substeps.

- $\{X\} \equiv \{I \land i \leq j+1 \land y \neq a[k] \land y < a[k] \land k = (i+j)\texttt{div}\ 2\}$ `j:=k-1` $\{I\}$
- $\{Y\} \equiv \{I \land i \leq j+1 \land y \neq a[k] \land y \geq a[k] \land k = (i+j)\texttt{div}\ 2\}$ `i:=k+1` $\{I\}$

By backward substituting I through the two (different) assignments, we get respectively

$$\begin{aligned}\{X'\} \quad \equiv \quad & (found = 1 \Rightarrow \exists h(1 \le h \le n \wedge y = a[h])) \\ & \wedge\ (found = 0 \Rightarrow \forall h((1 \le h \le i-1) \Rightarrow a[h] < y) \\ & \qquad\qquad\qquad\qquad \wedge\ \forall h((k \le h \le n) \Rightarrow a[h] > y)) \\ & \wedge\ i \le k\end{aligned}$$

$$\begin{aligned}\{Y'\} \quad \equiv \quad & (found = 1 \Rightarrow \exists h(1 \le h \le n \wedge y = a[h])) \\ & \wedge\ (found = 0 \Rightarrow \forall h((1 \le h \le k) \Rightarrow a[h] < y) \\ & \qquad\qquad\qquad\qquad \wedge\ \forall h((j+1 \le h \le n) \Rightarrow a[h] > y)) \\ & \wedge\ k \le j\end{aligned}$$

Now, it is fairly easy to prove $X \Rightarrow X'$ and $Y \Rightarrow Y'$ by discussing the cases $found = 0$ and $found = 1$. In particular, notice that we exploit the fact $y < a[k]$ (resp. $y > a[k]$) together with the fact that a is ordered: therefore for all indices p less than k (resp. greater than k) it is also $a[p] < y$ (resp. $a[p] > y$).

Proof of step 3 When I and $i > j$, it is $i = j + 1$. Thus we have:

$$\begin{aligned}\{I''\} \quad \equiv \quad & (found = 1 \Rightarrow \exists h(1 \le h \le n \wedge y = a[h])) \\ & \wedge\ (found = 0 \Rightarrow \forall h((1 \le h \le i-1) \Rightarrow a[h] < y) \\ & \qquad\qquad\qquad\qquad \wedge\ \forall h((i \le h \le n) \Rightarrow a[h] > y)) \\ & \wedge\ i = j + 1\end{aligned}$$

Now, we can show the implication $\{I''\} \Rightarrow \{Post\}$ by discussing on $found$. In fact, if $found = 1$ then $\exists h(1 \le h \le n \wedge y = a[h])$, so it is false that $\nexists h(1 \le h \le n \wedge y = a[k])$ and both implications of the postcondition are satisfied. Conversely, if $found = 0$ then $\forall h((1 \le h \le n) \Rightarrow a[h] \ne y)$ since it is either $> y$ or $< y$, thus in this case the postcondition is satisfied as well. This concludes the whole partial correctness proof.

Find the Maximum[2]

Prove partial correctness of the following program that finds the maximum of array a.

```
begin
  m:=a[1]; h:=2;
  while h <= N do
    k:=a[h];
    if k>m
      then m:= k;
    fi
    h:=h+1;
  od
end
```

[2]Also midterm of 5/5/2000.

With respect to the precondition and postcondition:

$$\{Pre\} \equiv \{N \geq 1\}$$

$$\{Post\} \equiv \max(m, a, N+1)$$

where we have defined the predicate:

$\max(m, a, h) \equiv \exists\, i(1 \leq i < h \wedge a[i] = m \wedge \forall j((1 \leq j < h) \Rightarrow (i \neq j \Rightarrow a[j] \leq m)))$

with the obvious meaning.

Invariant As invariant, it is easy to understand that we can choose:

$$\{I\} \equiv \{h \leq N+1 \wedge \max(m, a, h)\}$$

So we split the proof into the three canonical steps:

1. $\{N \geq 1\}$ `m:=a[1]; h:=2;` $\{I\}$
2. $\{I\}$ `while ... od` $\{I \wedge h > N\}$
3. $\{I \wedge h > N\} \Rightarrow \{Post\}$

Proof of step 1 By backward substituting I through the first two assignments, we get: $\{2 \leq N+1 \wedge \max(a[1], a, h)\} \equiv \{N \geq 1\}$, which is exactly the precondition.

Proof of step 3 We notice that $I \wedge h > N$ is the same as $h = N+1 \wedge max(m, a, N+1)$, which is a superset of the postcondition.

Proof of step 2 We first apply the rule for the `while` loop, so that we have to prove:
$\{I \wedge h \leq N\}$
```
k:=a[h];
if ... fi
h:=h+1
```
$\{I\}$

First, we backsubstitute I through the last assignment, thus reducing to proving:
$\{I \wedge h \leq N\}$
```
k:=a[h]; if ... fi
```
$\{h \leq N \wedge \max(m, a, h+1)\}$

To handle the `if` we split as usual into two subgoals. They are:

- $\{I \wedge h \leq N \wedge k = a[h] \wedge k > m\}$ `m:=k` $\{h \leq N \wedge \max(m, a, h+1)\}$

- $\{I \wedge h \leq N \wedge k = a[h] \wedge k \leq m\} \Rightarrow \{h \leq N \wedge \max(m, a, h+1)\}$

where, as in the previous exercise, we used the little "trick" of putting the result of the first assignment directly in the local annotation.

For the second part, it is simple to realize that $\forall j((1 \leq j < h) \Rightarrow (i \neq j \Rightarrow a[j] \leq m))$ and $a[h] = k \leq m$ is the same as $\forall j((1 \leq j \leq h) \Rightarrow (i \neq j \Rightarrow a[j] \leq m))$ (assuming $\exists i : a[i] = m$ which is the case), so the implication follows immediately.

For the first part, we do one backward substitution, thus having to show:
$\{I \wedge h \leq N \wedge k = a[h] \wedge k > m\} \Rightarrow \{h \leq N \wedge \max(k, a, h+1)\} \equiv \{J\}$.
Now, the antecedents rewrites as:
$\{\exists\ i(1 \leq i < h \wedge a[i] = m \wedge \forall j((1 \leq j < h) \Rightarrow (i \neq j \Rightarrow a[j] \leq m))) \wedge h \leq N \wedge k = a[h] \wedge k > m\}$
Basically, the above says that:

- $a[i] = m$;
- all elements $a[j]$ up to h other than $a[i]$ are such that $a[j] \leq m$;
- $a[h] = k > m$.

Thus, we can restate the above as:

- $a[h] = k$;
- all elements $a[j]$ up to $h + 1$ other than $a[i]$ are such that $a[j] \leq k$ (since $k > m$);

These two statements can be formally written as $\max(k, a, h+1)$, thus directly implying the expression $\{J\}$. This concludes the whole partial correctness proof.