Programming Languages: Imperative Program Construction
Practicals 10: Swaps in Arrays

Shin-Cheng Mu

Autumn Term, 2021

1. Prove that

$$\begin{array}{l} \{h[0] = 0 \wedge h[1] = 1\} \quad \text{-- hence } h[h[0]] = 0 \\ \mathit{swap}\ h\ (h[0])\ (h[1]) \\ \{h[h[1]] = 1\} \end{array}$$

Solution: Assume $h[0] = 0 \wedge h[1] = 1$, we have

$$\begin{array}{ll} & (h : h[0], h[1] \mapsto h[h[1]], h[h[0]]) \\ = & (h : 0, 1 \mapsto h[1], h[0]) \\ = & (h : 0, 1 \mapsto 1, 0) \quad . \end{array}$$

Therefore, let $h' = (h : h[0], h[1] \mapsto h[h[1]], h[h[0]])$,

$$\begin{array}{ll} & \mathit{wp}\ (\mathit{swap}\ h\ (h[0])\ (h[1]))\ (h[h[1]] = 1) \\ \equiv & h'[h'[1]] = 1 \\ \equiv & h'[0] = 1 \\ \equiv & 1 = 1 \\ \equiv & \mathit{True} \quad . \end{array}$$

2. Given $h : \textbf{array}\ [0..N)\ \textbf{of}\ A$, prove the rule that when h does not occur free in E and F,

$$\begin{array}{l} (\{\langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h[i] = H\ i \rangle\} \wedge h[E] = X \wedge h[F] = Y) \\ \mathit{swap}\ h\ E\ F \\ (\{\langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h[i] = H\ i \rangle\} \wedge h[E] = Y \wedge h[F] = X) \quad . \end{array}$$

Notes:

- Recall that E and F are expressions, while X, Y, H are logical variables. It means that, for example, one can conclude immediately $X[z \backslash w] = X$ for $z \neq X$, while to determine whether $E[z \backslash w] = E$ we have to look into E — $E[z \backslash w] = E$ if z does not occur free in E.
- With $h[E] = X$, for example, we implicitly assume that $\mathit{def}\ (h[E])$ holds.

Solution: Abbreviate $(h : E, F \mapsto h[F], h[E])$ to h'. We reason:

$$
\begin{array}{ll}
& wp\ (swap\ h\ E\ F)\ \langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h[i] = H\ i \rangle \wedge h[E] = Y \wedge h[F] = X) \\
\equiv & \quad \{ \text{definition of } wp;\ N, H, X, Y \text{ are logical variables} \} \\
& \langle \forall i : 0 \leqslant i < N \wedge i \neq (E[h \backslash h']) \wedge i \neq (F[h \backslash h']) : h'[i] = H\ i \rangle) \wedge h'[E[h \backslash h']] = Y \wedge h'[F[h \backslash h']] = X \\
\equiv & \quad \{ h \text{ does not occur free in } E \text{ and } F \} \\
& \langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h'[i] = H\ i \rangle) \wedge h'[E] = Y \wedge h'[F] = X \\
\equiv & \quad \{ \text{function alteration: } h'[i] = h[i] \text{ for } i \neq E \wedge i \neq F \} \\
& \langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h[i] = H\ i \rangle) \wedge h'[E] = Y \wedge h'[F] = X \\
\equiv & \quad \{ \text{function alternation} \} \\
& \langle \forall i : 0 \leqslant i < N \wedge i \neq E \wedge i \neq F : h[i] = H\ i \rangle) \wedge h[F] = Y \wedge h[E] = X \ .
\end{array}
$$

3. Derive the following program, where arrays are manipulated only by swapping.

$$
\begin{array}{l}
\textbf{con}\ N : Int\ \{0 \leqslant N\} \\
\textbf{var}\ h : \textbf{array}\ [0..N)\ \textbf{of}\ Int \\
\textbf{var}\ p : Int \\
? \\
\{0 \leqslant p \leqslant N \wedge \langle \forall i : 0 \leqslant i < p : h[i] \leqslant 0 \rangle \wedge \langle \forall i : p \leqslant i < N : 0 \leqslant h[i] \rangle\} \ .
\end{array}
$$

Solution: As the usual practice, we use an up-loop in which n is incremented in the end. Let $P\ n = \langle \forall i : 0 \leqslant i < p : h[i] \leqslant 0 \rangle \wedge \langle \forall i : p \leqslant i < n : 0 \leqslant h[i] \rangle$. The plan is:

$$
\begin{array}{l}
\textbf{con}\ N : Int\ \{0 \leqslant N\} \\
\textbf{var}\ h : \textbf{array}\ [0..N)\ \textbf{of}\ Int \\
\textbf{var}\ p, n : Int \\
p, n := 0, 0 \\
\{0 \leqslant p \leqslant n \leqslant N \wedge P\ n, bnd\ N - n\} \\
\textbf{do}\ n \neq N \rightarrow \ldots n := n + 1\ \textbf{od} \\
\{0 \leqslant p \leqslant N \wedge P\ N\} \ .
\end{array}
$$

Assuming $0 \leqslant p \leqslant n < N$, examine $P\ (n+1)$:

$$
\begin{array}{ll}
& \langle \forall i : 0 \leqslant i < p : h[i] \leqslant 0 \rangle \wedge \langle \forall i : p \leqslant i < n + 1 : 0 \leqslant h[i] \rangle \\
\equiv & \quad \{ \text{since } 0 \leqslant n < N, \text{ split off } i = n \} \\
& \langle \forall i : 0 \leqslant i < p : h[i] \leqslant 0 \rangle \wedge \langle \forall i : p \leqslant i < n : 0 \leqslant h[i] \rangle \wedge 0 \leqslant h[n] \\
\equiv & P\ n \wedge 0 \leqslant h[n] \ .
\end{array}
$$

Therefore, if $0 \leqslant h[n]$ there is nothing more we need to do before $n := n + 1$. We can introduce an **if** and put $n := n + 1$ under a guard $0 \leqslant h[n]$.

To make the **if** total we consider what to do when $h[n] \leqslant 0$. In this case we consider two cases.

Case: $p \neq n$. We aim to construct

$$
\begin{array}{l}
\{\langle \forall i : 0 \leqslant i < p : h[i] \leqslant 0 \rangle \wedge \langle \forall i : p \leqslant i < n : 0 \leqslant h[i] \rangle \wedge h[n] \leqslant 0 \wedge 0 \leqslant p < n < N\} \\
??? \\
\{P\ (n+1) \wedge 0 \leqslant p \leqslant n + 1 \leqslant N\} \\
n := n + 1 \\
\{P\ n \wedge 0 \leqslant p \leqslant n \leqslant N\}
\end{array}
$$

Since $p \neq n$, we can split $i = p$ from $\langle \forall i : p \leqslant i < n : 0 \leqslant h[i] \rangle$, resulting in

$\{\langle\forall i : 0 \leqslant i < p : h[i] \leqslant 0\rangle \wedge 0 \leqslant h[p] \wedge \langle\forall i : p+1 \leqslant i < n : 0 \leqslant h[i]\rangle \wedge 0 \leqslant p < n < N \wedge h[n] \leqslant 0\}$
swap h p n
$\{\langle\forall i : 0 \leqslant i < p : h[i] \leqslant 0\rangle \wedge h[p] \leqslant 0 \wedge \langle\forall i : p+1 \leqslant i < n : 0 \leqslant h[i]\rangle \wedge 0 \leqslant p < n < N \wedge 0 \leqslant h[n]\}$
$p := p + 1$
$\{P\ (n+1) \wedge 0 \leqslant p \leqslant n+1 \leqslant N\}$
$n := n + 1$
$\{P\ n \wedge 0 \leqslant p \leqslant n \leqslant N\}$.

Case: $p = n$. In this case the range $p \leqslant i < n$ is empty and the precondition reduces as such:

$\{\langle\forall i : 0 \leqslant i < p : h[i] \leqslant 0\rangle \wedge 0 \leqslant p = n < N \wedge h[n] \leqslant 0\}$
???
$\{P\ (n+1) \wedge 0 \leqslant p \leqslant n+1 \leqslant N\}$
$n := n + 1$
$\{P\ n \wedge 0 \leqslant p \leqslant n \leqslant N\}$.

It turns out that the same code still works:

$\{\langle\forall i : 0 \leqslant i < p : h[i] \leqslant 0\rangle \wedge 0 \leqslant p = n < N \wedge h[n] \leqslant 0\}$
swap h p n
$p := p + 1$
$\{P\ (n+1) \wedge 0 \leqslant p \leqslant n+1 \leqslant N\}$
$n := n + 1$
$\{P\ n \wedge 0 \leqslant p \leqslant n \leqslant N\}$.

Therefore the code is:

con $N : Int\ \{0 \leqslant N\}$
var h : **array** $[0..N)$ **of** *Int*
var $p, n : Int$
$p, n := 0, 0$
$\{0 \leqslant p \leqslant n \leqslant N \wedge P\ n, bnd\ N - n\}$
do $n \neq N \rightarrow$
  if $0 \leqslant h[n] \rightarrow n := n + 1$
  | $h[n] \leqslant 0 \rightarrow$ *swap h p n*
      $p, n, := p + 1, n + 1$
  fi
od
$\{0 \leqslant p \leqslant N \wedge P\ N\}$.

4. The following is a specification of sorting:

con $N : Int\ \{0 \leqslant N\}$
var h : **array** $[0..N)$ **of** *Int*
sort
$\{\langle\forall i\ j : 0 \leqslant i \leqslant j < N : h[i] \leqslant h[j]\rangle\}$.

where *sort* mutates the array h only by swapping. Derive a $O(N^2)$ algorithm for sorting. The algorithm will contain a loop within a loop. The outer loop uses as invariant $P_0 \wedge P_1$, where

$P_0 \equiv \langle\forall i : 0 \leqslant i < n : \langle\forall j : i \leqslant j < N : h[i] \leqslant h[j]\rangle\rangle$,
$P_1 \equiv 0 \leqslant n \leqslant N$.

The inner loop uses Q as *part of* its invariant:

$$Q \equiv \langle \forall j : k \leqslant j < N : h[n] \leqslant h[j] \rangle \ \ .$$

Solution: The invariant is designed such that $n := 0$ establishes $P_0 \wedge P_1$, while $P_0 \wedge P_1 \wedge n = N$ meets the postcondition. Therefore, the outline of the program could be:

$$\begin{array}{l}
\textbf{con } N : \mathit{Int} \ \{0 \leqslant N\} \\
\textbf{var } h : \textbf{array } [0..N) \textbf{ of } \mathit{Int} \\
\textbf{var } n : \mathit{Int} \\
n := 0 \\
\{P_0 \wedge P_1, \mathit{bnd} : N - n\} \\
\textbf{do } n \neq N \rightarrow \\
\quad \mathit{inner_loop} \\
\quad \{P_0 \wedge P_1 \wedge \langle \forall j : n \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge n \neq N\} \quad \text{-- (*)} \\
\quad n := n + 1 \\
\textbf{od} \\
\{\langle \forall i\ j : 0 \leqslant i \leqslant j < N : h[i] \leqslant h[j] \rangle\} \ \ .
\end{array}$$

The assertion (*) before $n := n + 1$ is calculated by:

$$\begin{array}{cl}
 & (P_0 \wedge P_1)[n \backslash n + 1] \\
\equiv & \langle \forall i : 0 \leqslant i < n + 1 : \langle \forall j : i \leqslant j < N : h[i] \leqslant h[j] \rangle \rangle \wedge 0 \leqslant n + 1 \leqslant N \\
\Leftarrow & \quad \{ \text{ with } 0 \leqslant n < N, \text{ split off } i = n \ \} \\
 & \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N : h[i] \leqslant h[j] \rangle \wedge \\
 & \langle \forall j : n \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge 0 \leqslant n < N \\
\equiv & \quad \{ \text{ def. of } P_0 \text{ and } P_1 \ \} \\
 & P_0 \wedge P_1 \wedge \langle \forall j : n \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge n \neq N \rangle \ \ .
\end{array}$$

We now try to construct the *inner_loop*. Compare the hint Q and the assertion (*), we note that

- $P_0 \wedge P_1 \wedge Q \wedge k = n$ establishes (*), and
- letting $k := N - 1$ establishes Q, and
- being in the outer loop, we have $P_1 \wedge n \neq N$, which is $0 \leqslant n < N$, therefore by choosing $k := N - 1$ we still have $0 \leqslant n \leqslant k < N$.

Therefore we start with trying:

$$\begin{array}{l}
\{P_0 \wedge P_1 \wedge n \neq N\} \\
k := N - 1 \\
\{P_0 \wedge Q \wedge 0 \leqslant n \leqslant k < N\} \\
\textbf{do } k \neq n \rightarrow \\
\quad ? \\
\quad k := k - 1 \\
\textbf{od} \\
\{P_0 \wedge P_1 \wedge \langle \forall j : n \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge n \neq N\} \\
n := n + 1
\end{array}$$

To construct ? we examine $Q[k \backslash k - 1]$, assuming $0 \leqslant n < k < N$:

$$
\begin{aligned}
& \langle \forall j : k \leqslant j < N : h[n] \leqslant h[j] \rangle [k \backslash k-1] \\
\equiv\ & \langle \forall j : k-1 \leqslant j < N : h[n] \leqslant h[j] \rangle \\
\equiv\ & \quad \{ \text{ with } 0 \leqslant n < k < N \text{, split off } j = k-1 \ \} \\
& \langle \forall j : k \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge h[n] \leqslant h[k-1] \\
\equiv\ & Q \wedge h[n] \leqslant h[k-1] \ \ .
\end{aligned}
$$

If $h[n] \leqslant h[k-1]$ already holds, we need only a *skip*. If $h[n] \geqslant h[k-1]$ holds instead, we do a *swap* $h\ n\ (k-1)$, whose validity can be established by:

$$
\begin{aligned}
& \mathit{wp}\ (\mathit{swap}\ h\ n\ (k-1))\ ((P_0 \wedge Q \wedge 0 \leqslant n \leqslant k < N)[k \backslash k-1]) \\
\equiv\ & \quad \{ \text{ calculation above, } k \text{ not occurring free in } P_0 \ \} \\
& \mathit{wp}\ (\mathit{swap}\ h\ n\ (k-1))\ (P_0 \wedge Q \wedge h[n] \leqslant h[k-1] \wedge 0 \leqslant n \leqslant k-1 < N) \\
\equiv\ & \quad \{ \text{ let } h' = (h \mathbin{:} n, k-1 \mapsto h[k-1], h[n]) \ \} \\
& P_0[h \backslash h'] \wedge Q[h \backslash h'] \wedge h'[n] \leqslant h'[k-1] \wedge 0 \leqslant n \leqslant k-1 < N \ \ .
\end{aligned}
$$

Consider the first three terms, $h'[n] \leqslant h'[k-1]$ equivals $h[k-1] \leqslant h[n]$ by the definition of function alteration (this is why we do *swap* $h\ n\ (k-1)$ in the first place). For the second term, we have $Q[h \backslash h'] \Leftarrow Q \wedge h[k-1] \leqslant h[n]$:

$$
\begin{aligned}
& Q[h \backslash h'] \\
\equiv\ & \langle \forall j : k \leqslant j < N : h'[n] \leqslant h'[j] \rangle \\
\equiv\ & \quad \{ \ h' = (h \mathbin{:} n, k-1 \mapsto h[k-1], h[n]) \text{ and thus } h'\ j = h\ j \text{ for } k \leqslant j < N \ \} \\
& \langle \forall j : k \leqslant j < N : h[k-1] \leqslant h[j] \rangle \\
\Leftarrow\ & \langle \forall j : k \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge h[k-1] \leqslant h[n] \\
\equiv\ & Q \wedge h[k-1] \leqslant h[n] \ \ .
\end{aligned}
$$

Consider $P_0[h \backslash h']$, assuming $0 \leqslant n < k < N$:

$$
\begin{aligned}
& P_0[h \backslash h'] \\
\equiv\ & \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N : h'[i] \leqslant h'[j] \rangle \rangle \\
\equiv\ & \quad \{ \ h' = (h \mathbin{:} n, k-1 \mapsto h[k-1], h[n]) \text{ and thus } h'\ i = h\ i \text{ for } 0 \leqslant i < n \ \} \\
& \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N : h[i] \leqslant h'[j] \rangle \rangle \\
\equiv\ & \quad \{ \text{ with } 0 \leqslant n < k < N \text{, split off } j = n \text{ and } j = k-1 \ \} \\
& \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N \wedge j \neq n \wedge j \neq k-1 : h[i] \leqslant h'[j] \rangle \wedge \\
& \qquad\qquad h[i] \leqslant h'[n] \wedge h[i] \leqslant h'[k-1] \rangle \\
\equiv\ & \quad \{ \ h'\ j = h\ j \text{ within } i \leqslant j < N \wedge j \neq n \wedge j \neq k-1 \ \} \\
& \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N \wedge j \neq n \wedge j \neq k-1 : h[i] \leqslant h[j] \rangle \wedge \\
& \qquad\qquad h[i] \leqslant h[k-1] \wedge h[i] \leqslant h[n] \rangle \\
\equiv\ & \quad \{ \text{ split off } j = n \text{ and } j = k-1 \text{, reversed } \} \\
& \langle \forall i : 0 \leqslant i < n : \langle \forall j : i \leqslant j < N : h[i] \leqslant h[j] \rangle \rangle \\
\equiv\ & P_0 \ \ .
\end{aligned}
$$

Therefore we have

$$
\begin{aligned}
& P_0[h \backslash h'] \wedge Q[h \backslash h'] \wedge h'[n] \leqslant h'[k-1] \wedge 0 \leqslant n \leqslant k-1 < N \ \ . \\
\Leftarrow\ & \quad \{ \text{ calculation above, some of them assuming } 0 \leqslant n < k < N \ \} \\
& P_0 \wedge Q \wedge h[k-1] \leqslant h[n] \wedge 0 \leqslant n < k < N \ \ .
\end{aligned}
$$

Note that, in deriving the inner loop we cannot forget about P_0 — one still has to prove that P_0 is preserved.

In conclusion, the program we derived is:

$$
\begin{array}{l}
\textbf{con } N : \mathit{Int}\ \{0 \leqslant N\} \\
\textbf{var } h : \textbf{array } [0..N)\ \textbf{of } \mathit{Int} \\
\textbf{var } n, k : \mathit{Int} \\
n := 0 \\
\{P_0 \wedge 0 \leqslant n \leqslant N, \mathit{bnd} : N - n\} \\
\textbf{do } n \neq N \rightarrow \\
\quad k := N - 1 \\
\quad \{P_0 \wedge Q \wedge 0 \leqslant n \leqslant k < N, \mathit{bnd} : k\} \\
\quad \textbf{do } k \neq n \rightarrow \textbf{if } h[n] \leqslant h[k-1] \rightarrow \mathit{skip} \\
\qquad\qquad\qquad\ \ \mid\ h[n] \geqslant h[k-1] \rightarrow \mathit{swap}\ h\ n\ (k-1) \\
\qquad\qquad\qquad\ \ \textbf{fi} \\
\qquad\qquad\qquad\ \ k := k - 1 \\
\quad \textbf{od} \\
\quad \{P_0 \wedge \langle \forall j : n \leqslant j < N : h[n] \leqslant h[j] \rangle \wedge 0 \leqslant n < N\} \\
\quad n := n + 1 \\
\textbf{od} \\
\{\langle \forall i\ j : 0 \leqslant i \leqslant j < N : h[i] \leqslant h[j] \rangle\}\ \ .
\end{array}
$$