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

Shin-Cheng Mu

Autumn Term, 2021

1. Prove that

   $\{h[0] = 0 \wedge h[1] = 1\}$ -- hence $h[h[0]] = 0$
   $\mathit{swap}\ h\ (h[0])\ (h[1])$
   $\{h[h[1]] = 1\}$

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

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

   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.

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

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

4. The following is a specification of sorting:

   $\mathbf{con}\ N : \mathit{Int}\ \{0 \leqslant N\}$
   $\mathbf{var}\ h : \mathbf{array}\ [0..N)\ \mathbf{of}\ \mathit{Int}$
   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:

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