Software-Modellierung und Verifikation
Informatik 2
Prof. J.-P. Katoen
RWTH Aachen

Priv.-Doz. T. Noll noll@cs.rwth-aachen.de
D. Willems willems@cs.rwth-aachen.de

7. Exercise sheet *Semantics and Verification of Software 2007*

Due to Wed., 6 June 2007, *before* the exercise course begins.

Exercise 7.1: (2 points)

Consider the following extension of the *WHILE* language:

- Let r be a meta variable for arrays out of the domain $\{[z_0, \ldots, z_{n-1}] \mid n \in \mathbb{N}\}$.
- Arithmetic expressions are extended by $|r|$ and $r[a]$.
- Commands are extended by $r := [a_0, \ldots, a_{n-1}]$ and $r[a] := a'$.

Let the semantics be given by:

- $\mathfrak{L}[\![|r|]\!]I\sigma = n$ for $\sigma(r) = [z_0, \ldots, z_{n-1}]$
- $\mathfrak{L}[\![r[i]]\!]I\sigma = z_k$ for $\mathfrak{L}[\![i]\!]I\sigma = k$
- $$\frac{}{\{A[r \mapsto [z_0, \ldots, z_{n-1}]\} \; r := [a_0, \ldots, a_{n-1}] \; \{A\}} \quad \text{for } z_k = \mathfrak{L}[\![a_k]\!]I\sigma \text{ for all } 0 \le k < n$$
- $$\frac{}{\{A[r \mapsto [z_0, \ldots, z_{n-1}]\} \; r[a] := a' \; \{A\}} \quad \text{for } z_k = \mathfrak{L}[\![r[k]]\!]I\sigma \text{ for all } 0 \le k, \mathfrak{L}[\![a]\!]I\sigma < n \text{ with } \mathfrak{L}[\![a]\!]I\sigma \ne k \text{ and } z_{\mathfrak{L}[\![a]\!]I\sigma} = \mathfrak{L}[\![a']\!]I\sigma$$

Analyse the *insertion sort* algorithm c_{sort} following steps (a) to (d):

$c_{sort} \equiv$ **while** $(p < |r|)$ **do**
  $q := p - 1;$
  $v := r[p];$
  if $(r[p-1] > r[p])$ **then** $r[p] := r[p-1]$ **else skip**;
  $p := p + 1;$
  $c_{find};$
  $r[q+1] := v;$

$c_{find} \equiv$ **while** $(q \ge 0 \wedge r[q] > v)$ **do**
  $r[q+1] := r[q];$
  $q := q - 1;$

(a) Using the Hoare rules, prove that c_{find} satisfies the invariant $A_{r,p} \;\wedge\; -1 \le q < p \le |r| \;\wedge\; r[q+1] \ge v$ where $A_{r,p} \equiv \forall 0 \le i < j < p : r[i] \le r[j]$ states that the values in r are monotonically increasing in the first p entries.

(b) Starting with the postcondition of c_{find} (see (a)), derive $\{A_{r,p} \wedge p = |r|\}$ as postcondition of c_{sort}.

(c) Starting with the precondition of c_{find} (see (a)), derive $\{A_{r,p} \wedge 0 \le p < |r|\}$ as precondition of c_{sort}.

(d) Analyse the precondition of c_{sort} to determine for which p and r the algorithm "works". What property (additional to $\{A_{r,p} \wedge p = |r|\}$) would have to be checked to verify that c_{sort} truely sorts r (in words).

Exercise 7.2:

Using the Hoare rules, show that c_{find} from exercise 7.1 terminates.

Note: All parts can be solved independently!