Computer Science 81, Spring 2010
Assignment 7
Due Tue. Mar. 9

The following program in Japeish is designed to compute the index `m` of the minimum value in a non-empty array. (If there is more than one value having the minimum, then this program computes the last one, but this is not part of the stated expectation.)

```
WHERE DISTINCT a, i, m IS

{0 < length(a) }
( m := 0; i := 1 )

{ … your invariant here … }

while i < length(a)
   do
      if a[i] ≤ a[m]
             then m := i
             else skip
      fi;
      i := i+1
      od

{∀j.(((0 ≤ j) ∧ (j < length(a))) → a[m] ≤ a[j])}
```

1. Devise an invariant that will be sufficiently strong to enable a total correctness proof of the program using Hoare logic.

2. Devise a variant that will enable proof of termination.

3. Construct a proof of the program using Hoare logic. I don't insist that you use JAPE for this, although it is strongly suggested. The program given is ready to enter once you've added the invariant.

4. State any lemmas that you use.

5. As with problem a06, I offer to check your invariant before you go too far with the proof, and provide hints if you get stuck.

6. Similar to the discussion in class on March 2, I found it helpful, in my proof, to bifurcate and use ∨-Elimination to deal with the verification conditions related to the conditional. The specific rule I used to create the bifurcation was
   A < B+1 |– A < B ∨ A = B

7. **Extra credit** (up to 100%): Finding the minimum index can be part of a *selection sort* program. Construct such a program and provide *its* assumption, expectation, and invariant. You might want to organize the program so that the min-finding sub-program is an opaque box using its assumption and expectation only.