Prof. Dr. A. Poetzsch-Heffter
M.Sc. Peter Zeller
Dipl.-Inf. C. Feller

**University of Kaiserslautern**
**Department of Computer Science**
**Software Technology Group**

# Exercise Sheet 9: Specification and Verification with Higher-Order Logic (Summer Term 2014)

## Exercise 1 Hoare Logics

Please download the file `Sheet9_hoare.thy`.

This file contains the Imp language with additional constructs for working with Arrays. The program semantics is defined in terms of a Big Step Semantics, similar to Sheet 7. In addition to the semantics, the file also contains a notion for Hoare triples and a Hoare logic calculus in the “WP-form” as presented in the lecture on slide 476.

a) Consider the following specification of the program `progA`:

```
⊢ {λs. vx⊳s = VX ∧ vy⊳s = VY} progA {λs. vx⊳s = VY ∧ vy⊳s = VX }
```

Here `vx⊳s` denotes the state of variable `vx` in state `s`.

Write down a definition of `progA` which satisfies this specification. Then use the Hoare logic to prove the correctness of your program with respect to this specification. Before you do the proof in Isabelle, write it down on paper by putting assertions between each code line.

b) Show that the specification in a) does not specify the program completely. To show this, define a different program `progB` which has a different behavior than `progA` but also satisfies the specification.

c) (optional) Give a specification of `progA` which specifies the program completely (except for termination). More precisely, your specification `swapSpec` should satisfy the following property:

```
⟦swapSpec pa; swapSpec pb; ⋀s. ∃s'. ⟨pa, s⟩→s'; ⋀s. ∃s'. ⟨pb, s⟩→s'⟧
      ⟹ ⟨pa, s⟩→s'  ⟷  ⟨pb, s⟩→s'
```

d) Consider the following program:

```
IF vx ≤ vy THEN
  SKIP
ELSE
  vx ::= vx + vy;
  vy ::= vx − vy;
  vx ::= vx − vy
END
```

Give pre- and post-conditions which specify that this program sorts the two variables `vx` and `vy`, so that after executing the program, `vx` contains the smaller number. Then proof that your specification actually holds for the given program.

e) Prove the following Hoare Triple on paper by adding assertions to the program:

```
{n ≥ 0}
i := 1;
sum := 0;
WHILE i <= n DO
        sum := sum + i;
        i := i + 1
END
{sum = Σ{0..n}}
```

Make sure that the assertion steps precisely match the rules of the Hoare Logic.

f) Transfer your proof from e) to Isabelle.

g) Write a program which swaps two elements in the array `ar1`. The two positions are given by the variables `vx` and `vy`. Specify the programs with a pre- and post-condition and proof that your program satisfies the specification.

h) (optional) Prove that the following program swaps the values of `x` and `y`.

```
z := x − y;
WHILE 1 ≤ z DO
 x := x − 1;
 y := y + 1;
 z := z − 1
END
```

i) (optional) Prove that the following program calculates the nth Fibonacci number:

```
x := 0;
y := 1;
i := 0;
WHILE i < n DO
  y := x + y;
  x := y − x;
  i := i + 1
END;
r = x
```

j) (optional) Prove that the following program calculates the result of dividing `x` by `y`. At the end of the program `i` contains the result of the division.

```
r := x;
i := 0;
WHILE y ≤ r DO
  r := r − y;
  i := i + 1
END
```

k) (optional) Implement a program which calculates the sum of all numbers in the array `ar1`. Proof that your implementation works correctly.

l) (optional) Implement a program which uses binary search to check if the number given by `vx` is contained in the sorted array `ar1`. Proof that your implementation works correctly.