J. Hoenicke
A. Nutz

14.06.2017
Please hand in your solution until
21.06.2017, via email to
`nutz@informatik.uni-freiburg.de`.

Tutorials for “Formal methods for Java”
Exercise sheet 7

Exercise 1: Dynamic Logic

For each of the following dynamic logic formulae find an equivalent formula without modalities.

(a) $[x = y + 1;]x = 3$

(b) $\langle x = y + 1;\rangle x < y$

(c) $[y = x\texttt{++} + 1;]x = 3 \lor y = 2$

(d) $[\mathbf{if}(y \texttt{==} 0)\{x = x + 1;\}\mathbf{else}\{x = x - 1;\}]x = 5$

(e) $\langle\mathbf{while}(x \texttt{!=} 0)\{x = x - 1;\}\rangle\mathbf{false}$

(f) $[\mathbf{while}(x \texttt{!=} 0)\{x = x - 1;\}]\mathbf{false}$

(g) $\langle\mathbf{while}(x \texttt{!=} 0)\{x = x - 1;\}\rangle x = 0$

(h) $[\mathbf{while}(x \texttt{!=} 0)\{x = x - 1;\}]x = 0$

Exercise 2: Integer square roots

Consider the following Java class:

```
class IntSqurt {
   /*@ requires n > 0;
     @ ensures \result * \result <= n
     @      && (\result + 1) * (\result + 1) > n
     @*/
   static int squrt(int n){
      int result = 0;
      int s = 1;
      while (s <= n) {
         result = result + 1;
         s = s + 2 * result + 1;
```

```
        }
        return result;
    }
}
```

Use the KeY prover to prove correctness of method `IntSqurt.squrt`. Find an invariant/variant proof that proves total correctness. Hand in either the KeY proof file, or a Java source file where the loop is annotated such that KeY can prove the program correct without further interaction.

Hints:

- The smallest working loop invariant we found consists of three parts (equalities/inequalities).
- For showing total correctness (the termination part), you also need the `decreasing` JML annotation.
- If you have an open proof goal remaining after applying the KeY tactic, you can use Z3 to give you a counterexample to your proof goal (e.g. a valuation of the variables that violates one of the proof goals, that you thus have to exclude).
  - In KeY's "Proof"-view, when you have an open goal selected, click "Run Z3" in the menu bar on top.
  - In the popup-window you should see a line "Counter Example.", click "Info" next to it.
  - Click the tab "Solver Output" and inspect the contents. For instance a line like `(define-fun x () Int 2)` means that the variable `x` is assigned the value 2.
- To use Z3, you need to point KeY to your Z3 executable (in KeY's preference page). If you don't have Z3 installed, yet: Go to `https://github.com/Z3Prover/z3/releases`, chose a version suitable for your operating system.