Secure and Dependable Systems
Jacobs University Bremen
Dr. Florian Rabe

Homework 5
given: 2017-03-28

Homework 5

You have to submit your solutions as announced in the lecture.
Unless mentioned otherwise, all problems are due 2017-04-06, before the lecture.
There will be no deadline extensions unless mentioned otherwise in the lecture.

Problem 5.1 *Verification: Class Invariants*

Points: 2

Argue informally but rigorously why the formula in the stack example from the notes is a class invariant.

Problem 5.2 *Verification: Class Invariants*

Points: 3

Solve one of the following (using pseudo-code, our example formal system, or a programming language):

1. Give a useful weak class invariant for the following class:

   class $Date(year : int, month : int, day : int)$
   fun $yesterday() : Date = \{\ldots\}$
   fun $tomorrow() : Date = \{\ldots\}$

   You can assume that there are no leap years.

   Implement the methods such that the class invariant is preserved.

2. Give a useful strong class invariant for the following class:

   abstract class $PriorityQueue()$
   private var $data : List[int] = Nil$
   fun $dequeue() : Option[int] = \{$**if** $(data == Nil)\ \{None\}$ **else** $\{Some(data.head)\}\}$
   fun $enqueue(x : int) : unit = \{\ldots\}$

   Implement *insert* such that the class invariant is preserved.

Problem 5.3 *Verification: Pure Functions*

Points: 3

Using the definitions from the notes, formally prove $m + n == n + m$ by induction.
To discharge the subgoals, you may use the theorems $zero\_left$ etc.

Problem 5.4 *Proof Assistants: Practice*

Points: 4

Install Isabelle or Coq (see the links in the lecture notes).

Write a simple pure recursive function and verify that it meets its specification using the tool. Generate executable code from your function.

Submit a reasonable combination of screen shots, shell logs, system output etc. that demonstrates you completed the task.

You may use any example that is already part of the available documentation or tutorials. But you have to prove that you actually installed the system and ran the verification. For example, you can copy an example from the tutorial, rename the function to your name, and then run the verification.