Homework 1

Metode formale în ingineria software 2020-2021
Formal Methods in Software Engineering 2020-2021

Deadline: Wednesday, November 25, 22:00. The answer will be included into a subfolder HW1 on Google drive (in the same folder with seminar/laboratory work). The subfolder HW1 will include:
– a pdf file containing a detailed description of the solution for each exercise and instructions how to test the source code;
– a Google doc file including the source code.

Remark

The homework is an individual task. The collaborations are not allowed; solutions with a high level of similarity will be rejected.
The homework must be signed and include the following
Sworn declaration
I hereby declare, under oath, that this homework has been my independent work and has not been aided with any prohibited means. I declare, to the best of my knowledge and belief, that all passages taken from published and unpublished sources or documents have been reproduced whether as original, slightly changed or in thought, have been mentioned as such at the corresponding places of the thesis, by citation, where the extent of the original quotes is indicated.

Exercise 1

a) Explain why in the Hoare axiom for assignment (slide 17, Floyd-Hoare Logic presentation) it is required the expression E to have no side-effects. Use (counter)examples to support your arguments.

b) Investigate if the Floyd axiom (slide 71, Floyd-Hoare Logic presentation) has the similar problems or if this requirement can be removed.

b) Describe a solution for the case when the expression can have side-effects.

Exercise 2 The semantics of a for statement

for (e_1 ; e_2 ; e_3) St

can be given by describing its simulation with `while`:

```
$e_1$;
while ($e_2$) {
  St
  $e_3$;
}
```

a) Explain how a correctness formula $(\!|P|\!)$ `for` (e_1; e_2; e_3) St $(\!|Q|\!)$ can be verified, using the above definition for `for`.

b) Design a proof rule for the `for` statement and show its correctness using a).

Exercise 3 A queue is a container of objects (a linear collection) that are inserted and removed according to the first-in first-out (FIFO) principle. The operations allowed on queues are:

`enqueue`(x, q) – add x as the newest element in q;

`dequeue`(q) – removes the oldest element from q;

`getFront`(q) – returns the oldest element from q.

Tasks:

a) Use the Dafny specification of the lists algebraic datatype (Dafny Reference Manual, page 88) to specify queues (together with their operations). Note that this is not an implementation of the queues, it must be just a specification (use only the specification language from Dafny).

b) Define in Dafny an implementation of the queues, using arrays.

c) Prove the correctness of the implementation b) w.r.t. the specification a).

d) Prove the following fundamental (characterization) properties of queues:
`dequeue`(`enqueue`(x, q)) = `enqueue`$(x,$ `dequeue`$(q))$, if q is non-empty
`dequeue`(`enqueue`(x, q)) = q), if q is empty
`getFront`(`enqueue`(x, q)) = `getFront`(q), if q is non-empty
`getFront`(`enqueue`(x, q)) = x, if q is empty

Prof. dr. Dorel Lucanu