Prof. Dr. A. Poetzsch-Heffter
Dipl.-Inform. Kathrin Geilmann

University of Kaiserslautern
Department of Computer Science
Software Technology Group

Advanced Aspects of Object-Oriented Programming (SS 2010)

Practice Sheet 8

Date of Issue: 04.06.10
Deadline: 09.06.10
(until 10 a.m. as PDF via E-Mail)

Exercise 1 Introduction to JML

The *Java Modeling Language* allows to specify properties of Java software by using special annotations. The JML homepage (`http://www.jmlspecs.org`) provides tutorials, papers, and tools you can use to solve the following exercises.

a) Summarize advantages of using a formal specification technique such as *JML* in comparison to informal approaches. Could you think of any drawbacks caused by the usage of formal specification techniques?

b) Make yourself familiar with the conceptual framework provided by JML. Use the paper „Design by Contract with JML“ by Leavens and Cheon as a starting point.

c) Install one of the JML implementations and make yourself familiar with its usage. For eclipse you can take the JML plugin from `http://pm.inf.ethz.ch/research/universes/tools/eclipse/`. If you don't use eclipse take a version of the common JML tools (`http://www.eecs.ucf.edu/~leavens/JML/download.shtml`). Use the tools for the following exercises to check your specifications and to execute your annotated programs.

d) Specify the following class. Give pre- and postconditions for the constructor and methods and a non-trivial class and loop invariant.

```
class Behaelter {

    int[] a;
    int n;

    Behaelter( int[] input ){
        n = input.length;
        a = new int[n];
        System.arraycopy(input, 0, a, 0, n);
    }

    int extractMin() {
        int m = Integer.MAX_VALUE;
        int mindex = 0;
        for (int i = 0; i < n; i++) {
            if (a[i] < m) {
                mindex = i;
                m = a[i];
            }
        }
        n--;
        a[mindex] = a[n];
        return m;
    }
}
```

e) It is obvious that the value `n` can only decrease over time. How can you specify this using JML?

f) Specify the JDK class `java.io.ByteArrayInputStream` using *JML*.

Exercise 2 Abstraction

a) Explain the necessity for JML's concept of *model fields*. Why are these especially important in the context of interface specifications?

b) Specify the following `Queue` interface according to the „well-known behaviour“ of this datastructure. *Hint: Look which model provided by JML may be appropiate.*

```
public interface Queue {
  Object  peek()                  throws EmptyQueueException;
  Object  dequeue()               throws EmptyQueueException;
  void    enqueue(Object item);
  boolean isEmpty();
  int     size();
}

class EmptyQueueException extends Exception {}
```

c) Take the interface with the specification and modify it, such that a) it declares a class Queue instead of an interface, b) implements all methods, c) relates the implemented methods to the specified model using depends and representation clauses.