

# Advanced Aspects of Object-Oriented Programming (SS 2009)

## Practice Sheet 8

Date of Issue: 16.06.09  
Deadline: 22.06.09  
(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.

- Summarise 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?
- 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.
- 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;  
    }  
}
```

- It is obvious that the value *n* can only decrease over time. How can you specify this using JML?
- Specify the JDK class `java.io.ByteArrayInputStream` using *JML*.

### Exercise 2 Abstraction

- Explain the necessity for JML's concept of *model fields*. Why are these especially important in the context of interface specifications?
- Specify the following Queue interface according to the „well-known behaviour“ of this datastructure.

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