Prof. P. Thiemann
M. Mehlmann

Sommer Term 2009

Software Engineering

http://proglang.informatik.uni-freiburg.de/teaching/swt/2009/

Exercise Sheet 4

Exercise 1: Design by Contract (7 Points)

The homepage of the lecture provides the code of a class `Stack` implementing stacks. Unfortunately, all pre- and postconditions as well as the invariants are missing. Please add them to the code.

Exercise 2: Design by Contract and Inheritance (9 Points)

Assume the following interface `Map`, which specifies mappings from keys (of type `K`) to values (of type `V`).

```
interface Map<K,V>
{
    boolean containsKey(K key);
    V get(K key);
    void put(K key, V value);
}
```

Beyond the usual requirements for the methods of `Map` (see `http://java.sun.com/j2se/1.5.0/docs/api/java/util/Map.html`), we impose the additional restrictions that (i) keys equal to `null` are not supported and (ii) the method `get` returns `null` if there is no mapping for the given key.

1. Add suitable pre- and postconditions to the methods of the interface.
2. Devise two subinterfaces of `Map`. Both subinterfaces should override existing pre- and postconditions of the methods of `Map`. One of the subinterfaces should specialize the methods correctly, the other should contain one illegal method specialization. Please justify your results.