Software Testing, SS 2022

Prof. Dr. Reinhold Plösch, Mag. Rudolf Ramler

Assignment 2: Blackbox Testing

Deadline: **April 7, 2022, 23:55**

Assignment: Black-box Tests for RingBuffer

Step 1: State transition diagram

Draw a state transition diagram for the class *RingBuffer* example (without *RingBufferIterator*). Use the notion as presented in the lecture slides. Document your model by means of a graphical diagram.

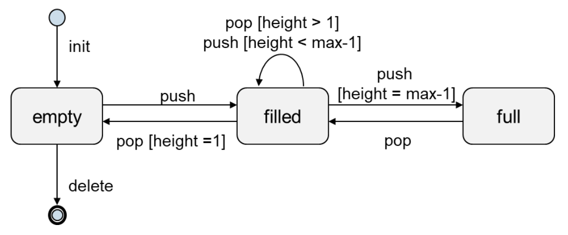
- The diagram must contain all possible states and events.
- Include invalid transitions that trigger exceptions (error cases). Note: These transitions are typically back loops, because invalid actions do not change the system's state.
- You can use an UML tool for the diagram, if you like. Yet a free drawing diagram is perfectly o.k.
- Store the diagram as png-file

Step 2: Logical test cases

Based on your state transition diagram, develop logical test cases in form of sequences of possible (valid and invalid) state transitions. Store your list of test cases as txt-file. For each test case define:

- Initial state of the test object
- Sequence of transitions containing source state, trigger event, input data, and expected target state
- Expected results or expected behavior
- Final state

Here an example for test cases for a Stack-Implementation that allows *push* and *pop* operations:



#	*Sequence*
1	init(2) **-empty-** delete() **-final**
2	init(2) **-empty-** push(a) **-filled-** pop() **-empty**
3	init(3) **-empty-** push(a) **-filled-** push(b) **-filled**
4	init(2) **-empty-** push(a) **-filled-** push(b) **-full-** pop() **-filled**
5	init(3) **-empty-** push(a) **-filled-** *push(b)* ***-filled-*** pop() **-filled**
6	init(2) **-empty-** push(a) **-filled-** push(b) **-full-** push(x)
7	init(2) **-empty-** pop()

Step 3: Implementing unit tests

Implement the derived logical test cases in form of unit tests using JUnit, where each logical test case corresponds to one unit test.

- Name the test class containing the implemented unit tests *RingBufferStateTransitionTest.java*
- Add the unit tests to the *RingBufferExample* package from the first assignment and submit the entire package as zip-file.

Submission & Presentation

Hand in the source code of your tests and any related files you created as zip archive and **upload it to the Moodle course platform before the deadline**.

Work in teams. Only one upload per team is required, i.e., one team member uploads the files to Moodle.

The work has to be **presented in the lecture on May 5th, 2022**. Each team has to be prepared to present their results. The teams that present will be selected at the beginning of the lecture. No slides (ppt etc.) have to be prepared, just bring a laptop with a development environment including all the tools you used and the source code and state transition model (including the derived test cases) of your solution.