Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Prof. Dr. Sascha Alda
Moritz Balg, Dr. Rita Cornely

7. Januar 2025

Übungen zur Vorlesung Software Engineering I
WS 2024 / 2025

Übungsblatt Nr. 10 (das letzte offizielle Blatt!! Juchuu!!)
(Bearbeitung bis 15.1.2025, 9:00 Uhr)

Aufgabe 1 (Testgetriebene Entwicklung mit Junit5, 20 Punkte. Abgabe bis 15.1.25; Vorbesprechung am 8.1.25. Einarbeitung von Kapitel 7, Abschnitt 6 notwendig):

Auf Github finden sie eine Java-Klasse `MyPrettyRectangleTest`, welche einen Testfall für eine Klasse `MyPrettyRectangle` implementiert:

https://github.com/aldaGit/codesSEws24/tree/main/test/org/hbrs/se1/ws24/tests/uebung10

Mit dieser Klasse `MyPrettyRectangle` kann man Rechtecke mittels der Angabe der Eck-Koordinaten (links unten und rechts oben) erzeugen. Durch die in dem Testfall enthaltenen Test-Methoden angedeutet sind vier verschiedene Methoden, die in der Klasse `MyPrettyRectangle` enthalten sein sollen: eine Methode zur Überprüfung, ob ein Rechteck ein anderes enthält, eine zur Berechnung des Mittelpunkts, eine zu Berechnung der Fläche sowie zur Berechnung einer Bounding Box. Des Weiteren finden sie eine Test-Methode zur Überprüfung der Objekt-Identität.

Mit Hilfe des Ansatzes der Testgetriebenen Entwicklung (vgl. Kapitel 7) sowie unter Zuhilfenahe der IDE IntelliJ und dem Framework JUnit5 sollen sie die Methoden der Klasse `MyPrettyRectangle` nun ausprogrammieren. Als Unterstützung dienen ihnen dazu die Testmethoden nebst den Testdaten, die in der Methode `setup()` angegeben sind. Sofern nicht schon vorgegeben, sollten sie die notwendigen Assertions dazu implementieren, um zu testen, ob ihre Implementierung korrekt ist. Weitere *wichtige* Hinweise finden sie außerdem in den Kommentarfeldern der Methoden.

Aufgabe 2 (Wiederholung Methoden zur Entwicklung von Black-Box-Tests; 10 Punkte; Abgabe bis 15.1.25; *kurze* Vorbesprechung am 8.1.25)

a.)
In Kapitel 7 haben Sie das Zustandsmodell zum Testen der Datenstruktur eines Stacks kennengelernt (Folie 30). Leiten Sie aus diesem Modell einen Junit5-Test ab, der eine *eigene* Implementierung eines Stacks, der eine maximale Zahl von *n* Objekten aufnehmen kann, hinreichend testet. Testen sie vor allem die Zustände, die in dem Modell angegeben sind. Tipp: Leiten sie die neue Entwicklung des Stacks von der vorhandenen

Klasse `Stack` aus dem Package `java.util` ab und überschreiben Sie vor allem die Methode `push` sinnvoll. Der Test sollte für *n = 4* durchgeführt werden.

b.)
Gegeben sei ein Source Code zur Validierung der Solvenz einer Person. Leiten sie daraus einen Black-Box-Test ab, der mindestens vier Äquivalenzklassen sowie, daraus resultierend, mindestens vier TestCases enthält. Formulieren sie die Äquivalenzklassen auf Basis des Parameters `age` (Attribut der Klasse Person). Benutzen sie zur Formulierung des Black-Box-Tests das Template aus LEA (TemplateTesteCase v1.6 (Reiter: Simple Test Suite). Ein Junit-Test ist hier nicht notwendig.

```
package org.hbrs.se1.app;
public class CheckPerson {

    /**
     * Methode zur Überprüfung der Solvenz einer Person, basierend
     * auf dem Alter einer Person. Laut Spezifikation sind negative
     * Altersangaben sowie ein Alter (age) = 0 keine gültigen Werte
     * Die Rückgabewert geben die Solvenz-Stufe an.
     *
     */
    public int checkSolvency( Person person ) {
        int age = person.getAge();
        if (age <= 0) throw new ArithmeticException("Illegal Value!");

        if ( age > 0 && age < 18 ) {
            return 0;
        }
        else if ( age >= 18 && age < 65) {
            return 1;
        }
        else {
            return 2;
        }
    }
}

public class Person {
    private String name;
    private int age;
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    String getName() {
        return name;
    }
    int getAge() {
        return age;
    }
}
```