Software Development I – Exercises **Winter Term 2017/2018**

Übungen zu Softwareentwicklung 1 **Assignment 2**

Name: ______________________ Teaching Assistant: ______________

Student ID (Matr.Nr.): ______________ Points (max. 24): ______________

Group: ☐ G1 ☐ G2 ☐ G3 ☐ G4 ☐ G5 ☐ G6 Deadline: **Tue., October 24, 2017 22:00**

Instructor: ☐ M. Haslgrübler ☐ C. Wirth ☐ T. Forstner Editing time (hours): __________

Preferred language for comments, proposals for improvements from TA's: ☐ DE ☐ EN

Problem 1: Camera-Rating 2+5+4+1 points

The magazine "My Camera and I" rates digital cameras based on only three factors. These 3 factors are "image quality", "battery life" and "zoom range". Each of these factors has a range from 0 to 10 (worst to best). For its Top 10 List of cameras the magazine uses a weighted mean to calculate the overall score for a camera

The weighted overall-score for cameras is calculated based on the formula below:

$$\text{overall score} = \left(6 \cdot \text{image quality} + 1 \cdot \text{battery life} + 3 \cdot \text{zoom range}\right) / 10$$

Develop a Java program `CameraRating.java` that calculates the overall score (the score of each single factor is given by the user as an integer number >= 0). Additionally the overall score should be presented correctly rounded (round half up) as integer and as integer without rounding (truncation of the floating part).

Examples

```
Image quality (0-10):1
Battery life (0-10): 5
Zoom range (0-10): 4
Calculated overall score: 2.3
Rounded overall score: 2
Integer overall score: 2

Image quality (0-10):9
Battery life (0-10): 9
Zoom range (0-10): 8
Calculated overall score: 8.7
Rounded overall score: 9
Integer overall score: 8

Image quality (0-10):1
Battery life (0-10): 4
Zoom range (0-10): 5
Calculated overall score: 2.5
Rounded overall score: 3
Integer overall score: 2
```

a) Write a short **textual description** (="Lösungsidee") of how you could implement a program that helps you with the camera rating but don't think about actual programming at this point!

b) Write then the Java code without worrying about input errors (e.g. factor values < 0 etc.). Don't forget to use the provided `Input.java` class for user input.

   - Pay attention to using adequate and reasonable data types (e.g., no need to use `long` as data type for input variables).
   - **Document your program well (i.e., include comments wherever feasible).**

c) Look at your code and the output it generates and **create a simple paper test plan** to help you improve your code in problem 1.b)

   Think about the following issues: What could go wrong during the execution of the program? Are there some limits to the format of the inputs (you have to look at all variables!)? What about the values, are all possible values acceptable? In your test plan, state the discovered problematic situations with input examples and what the expected output (e.g. an error message) in those situations should be.

   Execute a dry-run according to your test plan and document the results in a table.

Test case	**Input**	**Expected output**	**Acutal Output**
Negative value	-45	"The value of a score cannot be negative"	
…	…	…	

d) Explain with a few words why the presentation of the results of the overall score differs between "calculated overall score", "rounded overall score" and "Integer overall score".

Hints:

- The `Input.java` class will handle some input errors, but not all. See its source code for details.
- For correct rounding you can use the method `Math.round`()

Note: While in this question 1, both textual solution statement ("Lösungsidee") and test plan were requested explicitly, from now on (i.e. starting with Problem 2 below) your answers to programming questions are supposed to always include a textual description and a test plan

See the end of this document for the required material to be submitted.

Problem 2: Understanding Other's Code 6+4+2 points

A colleague of you plans a trip in the USA. In the USA the tip is an important part of the server's salary in a restaurant. So your colleague has developed a program for converting US-Dollar into Euro and calculating an appropriate amount of tip. Unfortunately, his implementation is erroneous and does not work as expected. Your task is to help him and fix the program. You find the specification and the source code in the following:

Develop a program capable of the following:

The program reads in a non-negative US-Dollar value and converts it to Euro (85 Euro = 100 US-Dollar). Furthermore, the program asks the user for the satisfaction level of the service and calculates the appropriate amount of tip. If the service was regular (r), the tip is 15% of the restaurant bill. In case of an especially good (g) service the tip is 20% of the restaurant bill.

The program outputs the restaurant bill in Euro without the tip and the appropriate amount of the tip in Euro and the bill including the tip in Euro.

```java
public class TipCalculator {
	public static void main(String[] args) {
		float sumDollar = 0.0f;
		float tipRate = 0;
		int conversion = 85 / 100;
		System.out.print("Enter sum of restaurant bill in US-Dollar: ");
		sumDollar = Input.readFloat();
		System.out.print("Enter kind of service (r for regular, g for good): ");
		char service = Input.readChar();
		if (service == 'r') {
			tipRate = 15;
		}
		if (service == 'g') {
			tipRate = 20;
		}
		float sumEuro = sumDollar * conversion;
		float tip = sumEuro * service;
		int totalBill = (int)(sumEuro * tip);
		System.out.println("restaurant bill in Euro: " + sumEuro);
		System.out.println("appropriate tip in Euro: " + tip);
		System.out.println("bill including tip in Euro: " + totalBill);
	}
}
```

a) Copy the code to a text editor or to Eclipse IDE and **run the** code. The results of the program seem strange. Describe the errors, fix the errors and adapt the code. Write then the Java code without worrying about input errors and add appropriate Java comments to the code.

b) Write a paper test-plan for your program. Make sure to think about all the problematic input a user can provide and the problematic states that could occur in the program. Execute a dry-run according to your test plan and document the results.

c) If somebody has a (converted) restaurant bill of exactly 85 Euro the tip assuming regular service would be exactly 12.75 Euro. What is the exact result of your Java program? Discuss the problem with a few words.

Requested material for all programming problems:

For each exercise, hand in the following:

a) Approach to solving the problem (textual representation)

b) Source code (Java classes including English(!) comments); in addition to the source code Java source files have to be converted to PDF and are to be included in the ZIP file,

c) Test plan for analyzing boundary values (e.g., minimal value allowed, maximum number of input, etc.) and exceptional cases (e.g., textual input when a number is required, etc.). State the expected behavior of the program for each input and make sure there is no “undefined” behavior leading to runtime exceptions. List all your test cases in a table (test case #, description, user input, expected (return) values).

d) The output of your java program for all test cases in your test plan.

Pay attention to using adequate and reasonable data types and meaningful English variable names for your implementation, check the user input carefully and print out meaningful error messages.