ASE Exercise 2

Task 1 (Basics Quiz)

1. What is the console output of the following lines of code?

```
if (false) {
	System.out.println("I am in IF- block");
} else if (true) {
	System.out.println("I am in ELSE-IF- block");
} else {
	System.out.println("I am in ELSE- block");
}
```

2. How do you create an integer array with 3 rows and 4 columns? How do you access the value at 3rd row and 4th column?

3. What is the value returned by the following function given the input parameter value = 6?

```
float QuestionThree(float value) {
	return value % (value * value);
}
```

4. What is the console output of the following snippet of code?

```
int i = 1;
while (i > 0) {
	i++;
	System.out.println("This is iteration: " + i);
	break;
}
```

5. What is the returned value of the following function, given the input parameters are first = 5 and second = 2?

```
float QuestionFive(int first, int second) {
	return first / second;
}
```

6. Give an example of an explicit typecast!

Task 2 (Looping)

In the following task you will practice more about loops in the java.

Given an array of integers as input, so your code should work for arrays of any length that consist of integers, print out every element of the array as long as the sum of the printed values is smaller than the product of the first and last element of the array.

Example:

Given the example array `int testarray = {1, 2, 3, 4};` The product of first and last element is 1 * 4 = 4. Printed to the screen is: 1,2, because the sum is 1+2=3 which is still smaller as 4.

1. Implement the task using a *while* loop
2. Implement the task using a *do-while* loop
3. Implement the task using a *for* loop
4. **Extension:** besides the three loop-types above, Java also offers an extended for-loop, known as *for-each* loop.
   Instead of counting and accessing each element of the array by its index, a *for-each* loop will automatically cycle through the elements of an array and puts each element after the other in a temporary variable.
   Example:

```
for(int value : testarray){
    System.out.println(value);
}
```

   Now implement the task using a *for-each* loop.

Task 3 (OOP)

In the following task, you are going to implement the class hierarchy of a software for managing a city information system.

1. Implement a class `Structure`. Thereby, the class should satisfy the subsequent requirements:
    - Three `protected double` fields: `length`, `width` and `height`.
    - One `protected string` field `location`.
    - One `public` getter method `getLocation()` which returns the `location`.

    Hint: See slides 32-35 and 56-60 for information.

2. Implement a class `Building`. This class **extends** the class `Structure` and has:
    - Two `protected int` fields: `rooms` and `floors`.
    - One `public` method `computeFloorSpace()` which computes the floor space (area) of the building. The floor space is defined as `floors*length*width`. Which return type does this method have?
    - One `public` getter method `getRoomSize()` which returns the average size of the rooms size (total area/rooms).

    Hint: See slides 42 and 43 for information.

3. Implement the interface `Rating`. The interface has one `public` method `computeRating()` which returns the rating as an `int`.
    Hint: See slides 69 and 70 for information.

4. Implement a class `Hotel` that extends `Building` and implements `Rating`. The class has:
    - One `private String` field: `name`. And the getter-method for this field.
    - A `public` constructor `Hotel(String hotelName, int hotelRooms)` which takes the parameters `hotelName` and `hotelRooms` and assigns them to their respective fields.
    - It implements the `computeRating()` method. The rating is calculated based on the average room size and with the following rules:
        - If the average room size is below 20 the rating is one.
        - If the average room size is above 20 but below 50 the rating is two.
        - If the average room size is above 50 the rating is three.

    Hint: See slides 63-68 for information.

5. Implement a class `CityManager`, which represents our entry point for the new application, and therefore provides the `main()`-method. Within the `main()`-method create two new buildings and one hotel. Thereby use the following specification:
    - Building 1 has five rooms and two floors. The dimensions (length, width, height) are 25m, 100m, 8m.
    - Building 2 has two rooms and one floor. The dimensions (length, width, height) are 100m, 50m, 6m.
    - The hotel has 500 rooms and five floors. It has the name “ASE Gloria” and the dimensions (length, width, height) are 100m, 40m, 50m.

    After the initialization of these structures, implement the code to generate the following output on the console (use method calls or direct instance field access whenever possible):

```
Our city has the following buildings:
        Building 1 with 2 floors and 5 rooms. It offers a floor space of 5000.0
        Building 2 with 1 floors and 2 rooms. It offers a floor space of 5000.0
Further, we have a hotel:
        ASE Gloria with 5 floors and 500 rooms. It has a rating of 2 stars.
```

Task 4 (Recursive Methods)

In the following task you will practice the application of recursive function calls. Implement a method, that recursively calculates the sum of all integers up to the given value.

Do not use the gaussian method!

Example: `f(10) = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55`

Hint: First think about a mathematic description of the given problem in a recursive way, then implement the function in java.