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

**Übungen zu Softwareentwicklung 1** **Assignment 10**

Name: _____ Teaching Assistant: _____

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

Group: ☐ G1 ☐ G2 ☐ G3 ☐ G4 ☐ G5 ☐ G6 Deadline: **Tue., January 23, 2018 22:00**

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

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

---

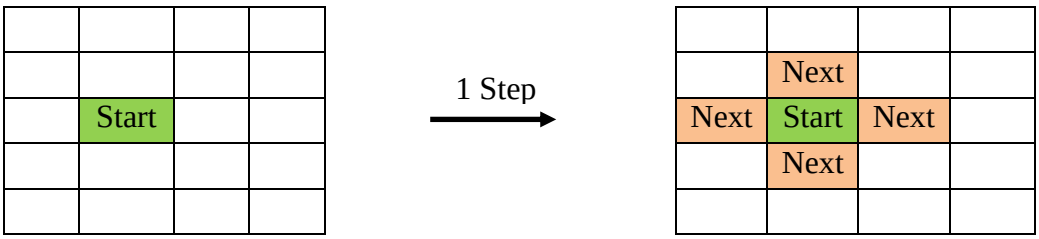
**Problem: Recursion** **24 points**

Create a class Map that stores a 2-D array of type char with elements either a hash sign ('#') or a capital letter of o ('O'). You can define the size of the map by yourself with one restriction. The columns should be twice the rows. Use a boolean random generator to initialise every element in your 2-D array with '#' or 'O'.

Besides the constructor, the class Map should have the following two methods:

- toString … returns a string that is a textual representation of the 2-D array
- eliminateArea … eliminates a connected area inside the map by setting the elements of the connected area to ' ' starting with a certain row and column value.

This eliminating process can be easily done by using the eliminateArea method recursively. In this case, the eliminateArea method is called for every neighbouring element that has the same value as the starting value (for remembering the start value use it as parameter). Be aware of the 2D-array boundaries, to check every element in the 2-D array only once and to eliminate only values which are in the connected area.




Furthermore, your program should meet the following conditions:

- Print your map before and after you call the eliminateArea method. Additionally, you can print the map after every iteration to see how your recursive algorithm performs.
- The user inputs the starting point of our eliminating algorithm (check for correct inputs).

**Example :**

- green background indicates starting point
- gray background indicates connected area

```
###O#OOOOO##O###OO#O
#O###O#O###O##OO##O#
#OO#O##O#O#OOO#O#O#O
OO####OO#OOOOOOO##OO
O#O##OO###OOO######O
#O##OOOO##O#OO#O##O#
O########O##OO#O#O#O
OO#OOOOO##OOOO#O##OO
###O##OO##OOOOO#OOOO
#OO##OOO##O#####O#OO

Eliminating certain area
row value (0 .. 9): 1
column value (0 .. 19): 3

   O  OOOOO  O###OO#O
  O    O O    O##OO##O#
  OO  O   O  O OOO#O#O#O
OO      OO  OOOOOOO##OO
O#O    OO    OOO######O
#O    OOOO   O#OO#O##O#
O          O##OO#O#O#O
OO  OOOOO   OOOO#O##OO
    O##OO   OOOOO#OOOO
```

---

## Requested material for all programming problems:

- **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 <u>boundary values</u> (e.g., minimal value allowed, maximum number of input, etc.) and <u>exceptional cases</u> (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.