Assignment 1

Exercise 1

Part 1

Task

Draw the structure of the project's packages and classes. You have to: (i) decide the level of abstraction you want to use in this depiction, (ii) use natural language to explain your decision, and (iii) describe what you understood from this depiction of the system

Diagram of the project's packages and classes

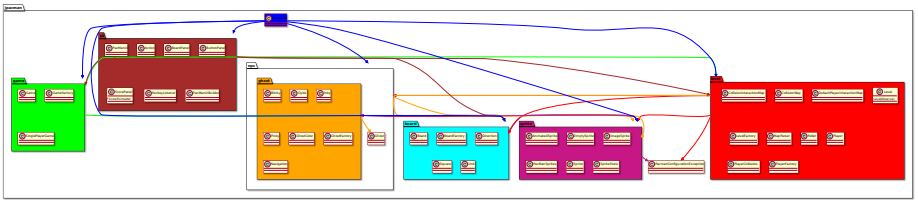

Figure 1: pacman-project-structure

Why was this level of abstraction used?

The graph above depicts a simplified version of a class diagram for the Java Pacman Game. Since there are a lot of dependencies between the classes drawing all of the lines between them would've caused lots of confusion and it would be very difficult to differ between the distinct lines going from one class to another. Hence to simplify, we used a color coding scheme for each package and only drew the connecting lines between the different packages.

Description of the system based on the diagram above

Hereby we learn, that the entire game is divided into 6 larger packages. Most of the classes are inside their respective packages, only the classes "Launcher" and "PacManConfigurationException" throw exceptions. Initially, we can see that the class "Launcher" depends on all other packages and imports classes from them. Classes in the packages "sprite" and "board" are also imported in most of the other packages. This is logical, since both the board and the actual PacMan figure (Sprite) are both fundamental parts of the implementation of a PacMan game. These two packages are the main packages of the Java PacManImplementation.

Part 2

Task

Draw a call graph, starting from what you deem the most prominent entry point. You have to: (i) decide how many levels you want to have in the call graph, (ii) use natural language to explain your decision, and (iii) explain what you have understood from this call graph about the dynamic behavior of the system. Hint: this project may have several entry points; those contained in an example or test folder/class are hardly the most prominent ones

Call graph diagram

Why was this level of abstraction used?

Description of the dynamic behaviour of the system based on the diagram above

Exercise 2 - A Checkers Game - Design

Part 1

Task

Following the Responsibility Driven Design, start from the game's requirements and rules and derive classes, responsibilities, and collaborations (use CRC cards). Describe each step you make and store the final cards in your answers

CRC Cards

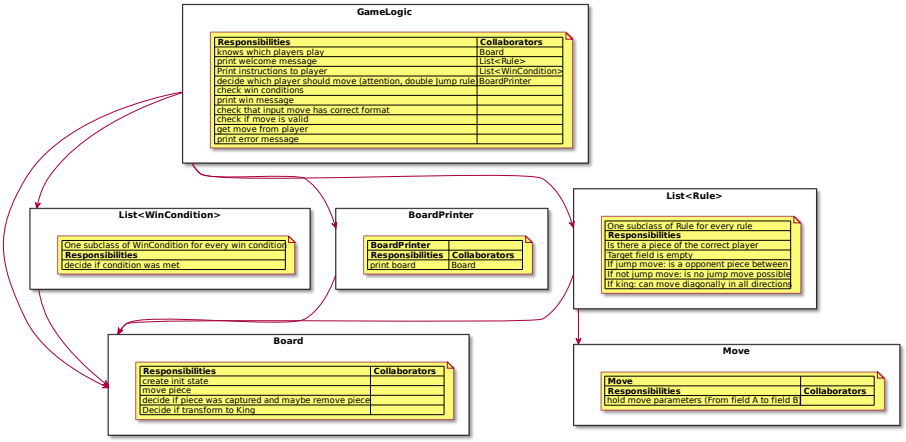

Figure 2: checkers crc cards

Description of the steps which lead to these CRC cards

Part 2

Task

Following the Responsibility Driven Design, describe the main classes you designed to be your project in terms of responsibilities and collaborations

Description of the main classes

Part 3

Task

Why do you consider the other classes as less important? Following the Responsibility Driven De- sign, reflect if some of those non-main classes have similar/little responsibility and could be changed, merged, or removed

Part 4

Task

Draw the class diagram of the aforementioned main elements of your game (do not forget to use elements such as parametrized classes or association constrains, if necessary)

Class diagramm

Part 5

Task

Draw the sequence diagram to describe how the main elements of your game interact (consider asynchrony and constraints, if necessary)

Sequence diagram