**Software Development Project**

**Report**

|  |  |
| --- | --- |
| **PROJECT TEAM** | |
| **Student No.** | **Student Names** |
| **16138457** | **Lorenzo Cipriani** |
| **16134613** | **Nassima Kara** |
| **16135091** | **Sebastien Zekpa** |

**Morra (Evens or Odds)**

**22nd April 2017**

# Introduction

The main goal of the team was to use, for the development of the project, all and only the topics covered during the lectures. Even if the game itself is not so complicated to implement in java, the team tried to show some key aspects learned; e,g:

* Instantiable classes (the main class is just an entry point then the flow of the program is managed by the GameController, this instantiates the 2 players of the game and manage their states)
* Use of design pattern (GameController is a lightweight implementation of the Chain Of Responsibility pattern)
* Incapsulation (all attributes defined as private/protected and only getters&setters are used to access the state of the objects)
* Validation of user’s input
* Conditional statements (if and switch)
* Loops (while, for)
* Abstract classes and inheritance (Player is an abstract class extended by Computer and Human)
* Method overloading (e.g.: Player.hasOdds() & Player.hasOdds(boolean).)
* Method overriding (showFingers() in Player is an abstract method that must be overridden by Human and Computer for their own different behaviours when that method is called.
* Use of standard developers tools: Eclipse IDE, Papyrus UML, Git, Ant

**Team roles**

**Lorenzo**

I work with the team to analyse the requirements from the problem and design the game and follow the necessary steps required to achieve a successful coding. We first establish the activity diagram, then worked on the class diagram to define the content and limitations of the game. I was in charge of writing the class Morra.java game. I mentored my two peers to write the code for Human.java, Computer.java, Player.java, Morra.java,GameController.java, package-info.java and reviewed the mistakes and actions to implement to respond to the requirements.

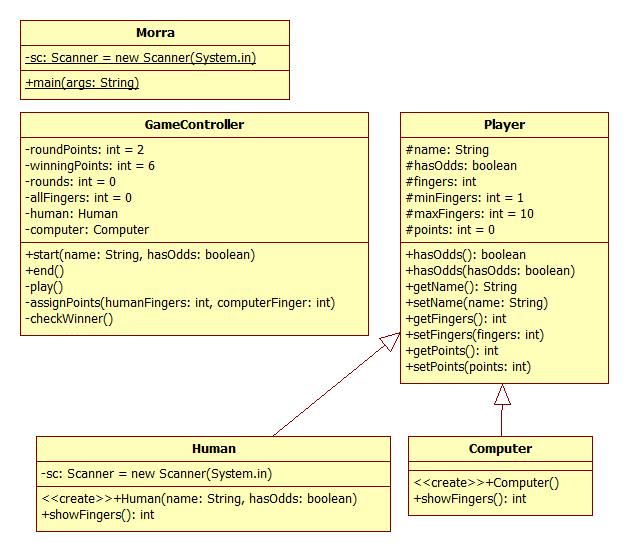
**Nassima**

I was in charge of the GameController.java. This class controls the gameplay and the players, it determines the rules that the players (human or computer) should follow during the game.

**Sebastien : I was in charge of the Player.java: It’s the only abstract class of the game. Human.java and computer java: These classes have been created to differentiate Human class from Computer class, because they have important difference in behaviours. I have also built the class diagram.**

# Diagrams

**Overview of the game (class diagram)**



**Overview of the game (Activity diagram)**

Activity Diagram as a prerequisite to build the game, one of my tasks was to build the activity diagram of the game.

The activity diagram purpose is to:

* Draw the activity flow of a system
* Describe the sequence from one activity to another
* Describe the parallel, branched and concurrent flow of the system

At the start of the program, the system asks the player the type of number he wishes to play. The human player will be asked first to provide a number between 1 and 10. Once this condition is met, the computer will provide a number. As both players have chosen their respective numbers, the result will be displayed. The game controller will check if the number is “odd” or “even” then assign the points to the winner.

If the score is higher than 6, the score is recorded and stored, on the contrary if the score is less than 6, we go back to the step 5 of the game.

