Assignment 2, Group 47

Exercise 1

We used getplayer() to realize validity judgement and some function of print of assignment 1, we remove this method this time and make all the variables private.

Exercise 2

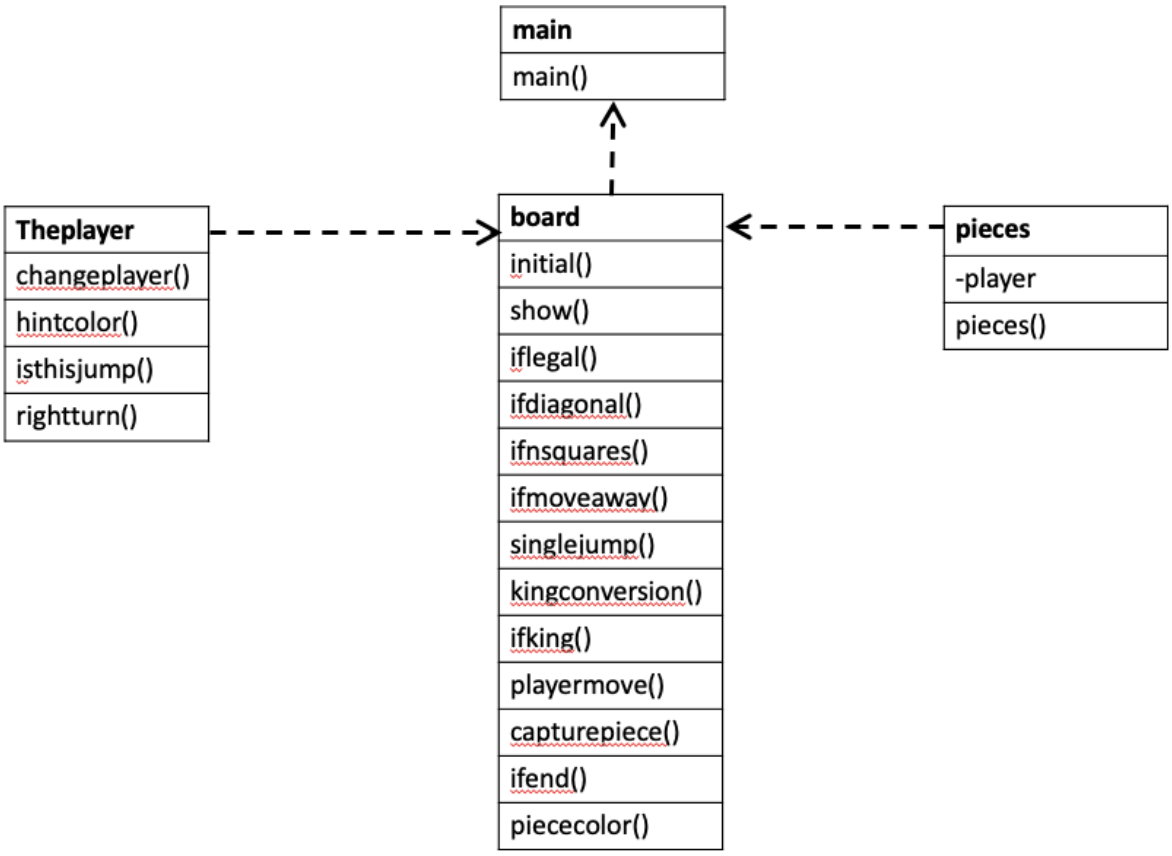
1. We decided to mainly improve our code instead of extending it and changed following things:

- Which classes we have and how the relate to each other. We added the new class Theplayer, to determine which players turn it is, and also to print messages whose turn it is to the user, and also if there is a multiple jump possibility, since we only managed to implement one jump at a time. We tried to accomplish all of these tasks previously, but we refined it inside a new class.
- The way the board is printed:
    - R -> [R_P] or [R_K]
    - W -> [W_P] or [W_K]
    - + -> [ ]
    - We had the white pieces at the top of the board in our first version, so we switched those
    - We added margins
- The input. Before the user had to give four inputs, two for each coordinate and we had those inputs stored in different variables. We changed the input to the format the assignment demands, made the input into an array of characters and assigned (and if necessary converted) each character to our previous variables. We came up with an issue that we could not resolve: we tried to print an error in case the user input does not match the specified input (line 31 in Main), but we decided to comment the line since the piece would move if the input was correct, but the error

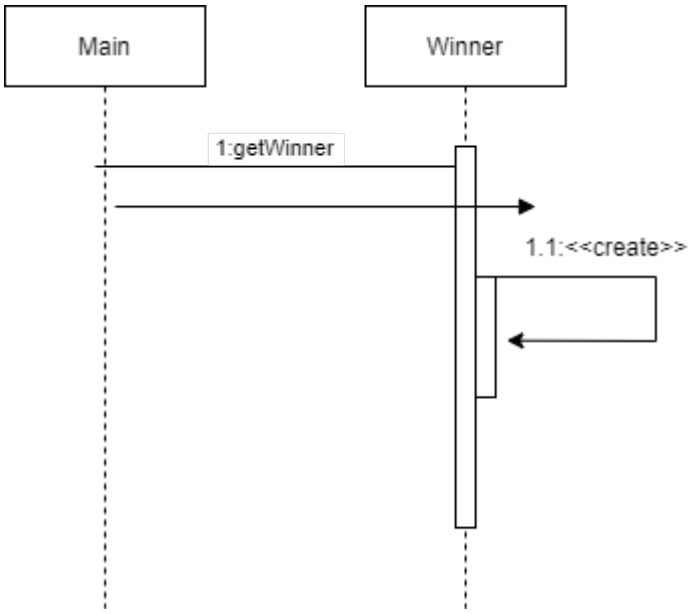
message came anyways. Our Boolean correctInput always seemed to be false. If the input was incorrect, the program would stop.

2. We made a new CRC card for our new class Theplayer and for the UML a class diagram.

Theplayer	
Purpose	Collaborators
Change player	board
Tell user what colors turn it is	
Tell user if there is a possible multiple jump	



Exercise 3

1. Every game has only one winner at the end, we use the Singelton pattern to create the class Winner.
2. 

3.