Report P4 ADSOF

Pablo Sancho Saiz & Daniel Varela Sánchez

**Sections 1,2,3 & 4**

In the first section we created the Player, Token (and all its inherited classes NormalToken, FixedToken, EnhancerToken, MultiplierToken and WallToken), classes as it was asked. In this section there wasn’t further complication from implementing the IToken interface in the Token class.

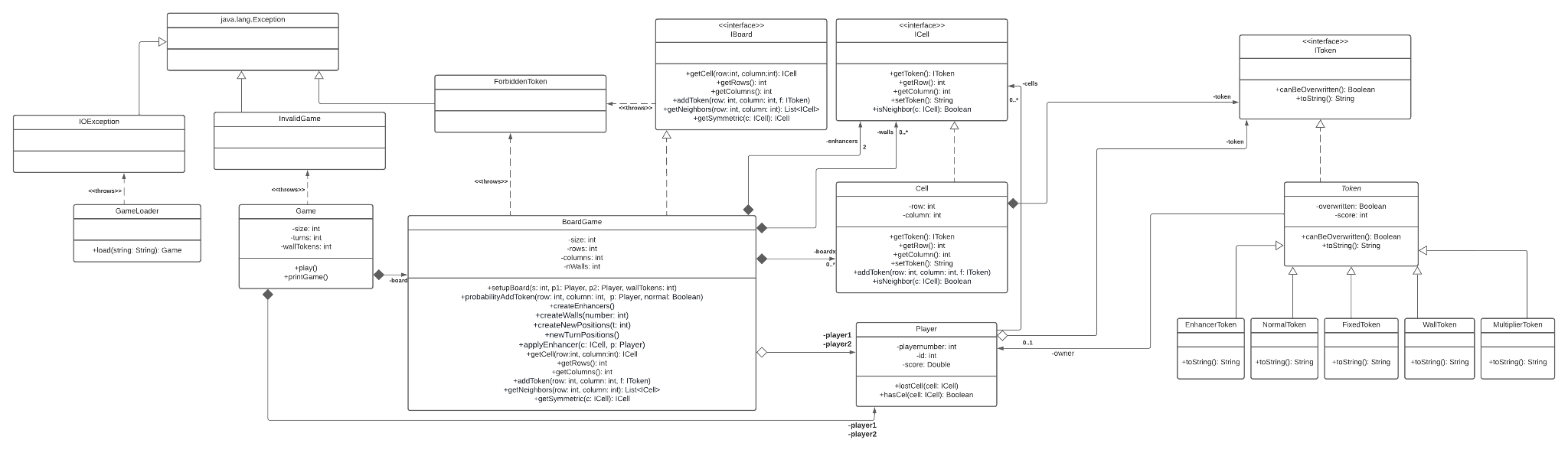
Then, in the second section it was turned to create the board and cell classes, both of them had to implement the IBoard and ICell interfaces respectively. The problem here was that we didn’t know how to implement in an optimal way the board with all the cells so that we can easily access them, and what we thought was to create a list (rows) of lists (columns). Also we made the ForbiddenToken exception.

The most problematic section was the third one since it requires that the implementation of the game is nearly finished. This time we'd create the buffered reader to take as parameters the size of the board, the turns and the maximum walls per player. Finally the InvalidGame exception was created to handle if the parameters entered by the user are wrong.

As we were saying firstly in the last paragraph, we had to solve things like the neighbor cells, score of the players, functionalities of the enhancers as well as the probability of choosing the type of token introduced by the user, and the change of positions of walls and enhancers in each turn.

Finally, the last section of coding was the easiest one to implement since what you’ve got to do is a class and instead of being the user who enters the parameters, is this class who reads from a file and initializes a new game.

This is the Class Diagram:



**Section 5**

1) Imagine that you are asked to create another board game (e.g., chess or checkers). What classes or interfaces could you reuse? Would any additional methods need to be added?

We could reuse all the interfaces and the following classes:

* Token.java
* Player.java
* Cell.java

And we could also reuse the LoadGame, Game and BoardGame classes so that we would have to make changes only in the methods that check for possible moves (like getNeighbours or isNeighbour), the method play() that limits the game to a certain amount of moves in the class Game should also be modified and a new method to check if the game has ended should be added.

2) Suppose we create an additional token whose value is temporary. That is, when it is placed for the first time it has a value of 3 points but after 3 turns, its value returns to 1. How would you include this new functionality? No need to program anything,

just indicate how you would do it.

A new attribute startMove will be added and required when creating this type of token, then a new method updateTurn(int t) in the class Board will be added and this will go token by token calling the method update() so that in the temporalTokens it changes their value from 3 to 1 when the difference between the startTurn attribute and the t parameter is equal to 3.

**Execution:**

To execute our tests, you must do **make**, when you’re in the makefile directory, and then, for example, **java tests.GameMainFromFile ../txt/InvalidGameSize.txt** since they’re in another directory.