CSDS 132: Project 5 Testing Report
Omar Loudghiri

I will only be testing the types I wrote for this project and therefore only the types assigned in the instructions:

I.Types:

1. Chessboard

This is now an interface and will be implemented by JavaFXChessBoard, however most the methods were written previously.

2. SwingChessboard:

This is the same class as the one from project 3, only some names were modified but no core code.

3. SwingChessBoardDisplay : the exact same code as project 3.
4. SwingEuropeanChessBoardDisplay: the same as project 3.
5. Chess Game

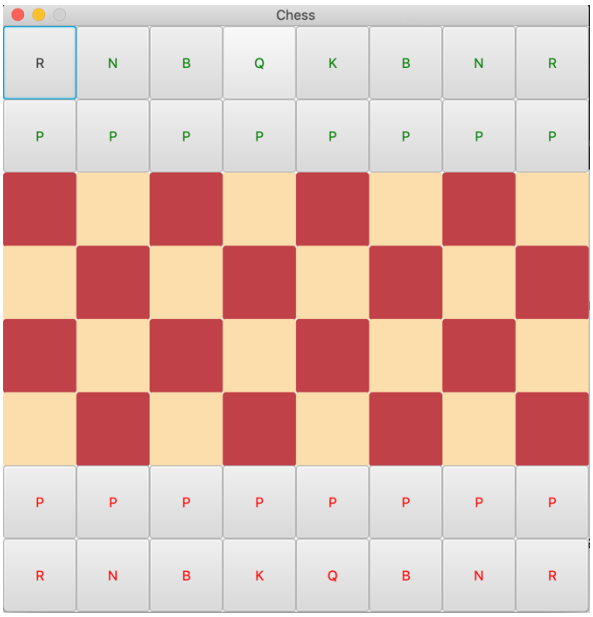
An interface that sets the rules for a game on a chessboard, it is directly taken from the implementation of European chess from project 3.

6. European Chess:

Mostly extends default methods from the chess game interface

7. JavaFXChessBoard:

The methods of this class are mostly graphical and need to be tested while displayed on a java FX display, the start method takes an input that returns the first argument and creates a game based on the input. It creates all the specific needed for a Chessboard. When the start method is ran with the "chess" input it shows a board that is very similar to the swing one, which is on the left. We then check that all the buttons work by clicking on them while having a print method in the even handler and they all work. We also check that the first and second click fucntions work by moving some pieces around.

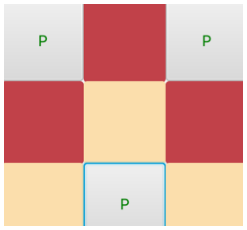


8. JavaFXChessBoardDisplay:

This is an interface for how both games should be displayed. It is tested using the two next classes

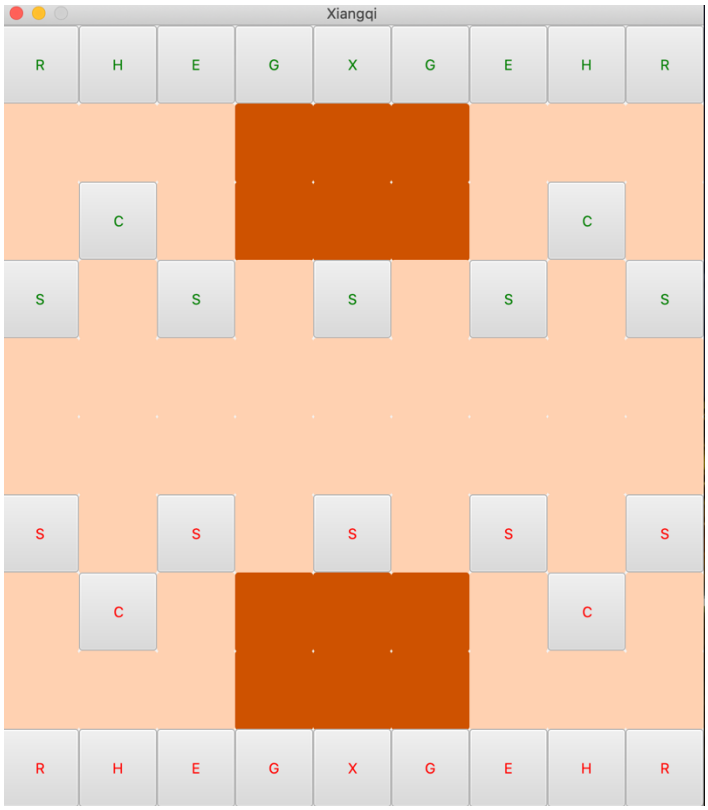
9. JavaFXEuropeanChessDisplay:

This method sets the color code for a chess display which is the one shown above. In order to test the empty square method we move a piece which results in removing it from a place and hiving the display emptysquare being called, same goes for the filled square, we test that the after moving a square and noticing that a new button is created as show below.



CSDS 132: Project 5 Testing Report
Omar Loudghiri

10. JavaFX Xiangqi Display:



This an entirely graphical class
This class sets the rules for the way a xiangqi display looks like here an example of the whole board. We can see that the empty squares are displayed the way a xiangqi board is supposed to be and the pieces are on the right positions. the filled squares and empty squares function the same way as the earlier class, we can test those by moving pieces around, we can also test the highlight method by clicking on a piece we can select.

11. Xiangqi

This class is the first new implementation we can test the getter and setter method of the class with JUNIT. The start method of this class is tested graphically too but some aspects of it are also on the Junit test.
First we test playing side getter and setter methods.
We then test that the num Rows and num Columns return the actual dimensions of the game
We also test that the pieces created by start game are really on the board.

II.New Xiangqi Pieces:

To test those pieces we will create a new piece and add it on a new board and then test all the allowed moves. None of the pieces are supposed to move if they let king be on its own with another king on the same column. We test this by making game with all the pieces and checking that the pieces don't move if they are the only piece in between to kings and by checking that a king cant move into a situation where it opposes a king

1. King Piece(X): we test the king moves by putting on a board and checking the king can move vertically and horizontally but not out of his 3x3 zone. We also check the king can capture properly.
2. Guard Piece(G): also cannot move out of the inner square, the Guard piece can move diagonally and we test that on Junit, we also test it can capture properly.
3. Elephant Piece(E): this piece can move two pieces only if the the first square diagonally on the move path is free. We check it can move that way and check it cannot move when there is a piece, we also check that it can capture. It also cannot cross the center so we check that with Junit
4. Horse Piece (H): this piece is going to be tested the exact same as a horse piece with but now it cannot jump on it's first move, so we test that by adding a piece on the adjacent square, we also test that it can capture.
5. Rook Piece is already tested from project 3.

6. Canon Piece, to test this piece we test it like a rook, it can move any distance vertically or horizontally, and it cannot jump pieces so we test those possibilities. The difference is for capturing moves, when capturing , it must at least jump one piece so we create a piece in the middle between the captured and the canon.
7. Soldier Piece this piece can only move away from its side source, however when it crosses the middle it can move horizontaly, we will test all those situations. Soldiers can also capture like they move so we will test a capture move