Swen 261 Team 5 Rhamz - Code Metrics.

For Code Metrics, we only had problems with 4 classes. The GetGameRoute & GetHomeRoute classes fail the OCavg metric. The TestGame class failed the MVC metric. And the Game class failed the OCavg & MVC metrics.

In the UI tiers, the GetGameRoute & GetHomeRoute classes, We looked over the complexity and I didn't feel as if anything needs to be implemented here because all the methods in those classes are as simple as can get, and there isn't anything that can be worked on, comments wise to fix the code nor simplify the number of lines. Most of the code is a part of the Spark library and is all necessary, such as the vm.put() method. We also have a comment for each line, but those are very necessary because the code gets hard to follow.

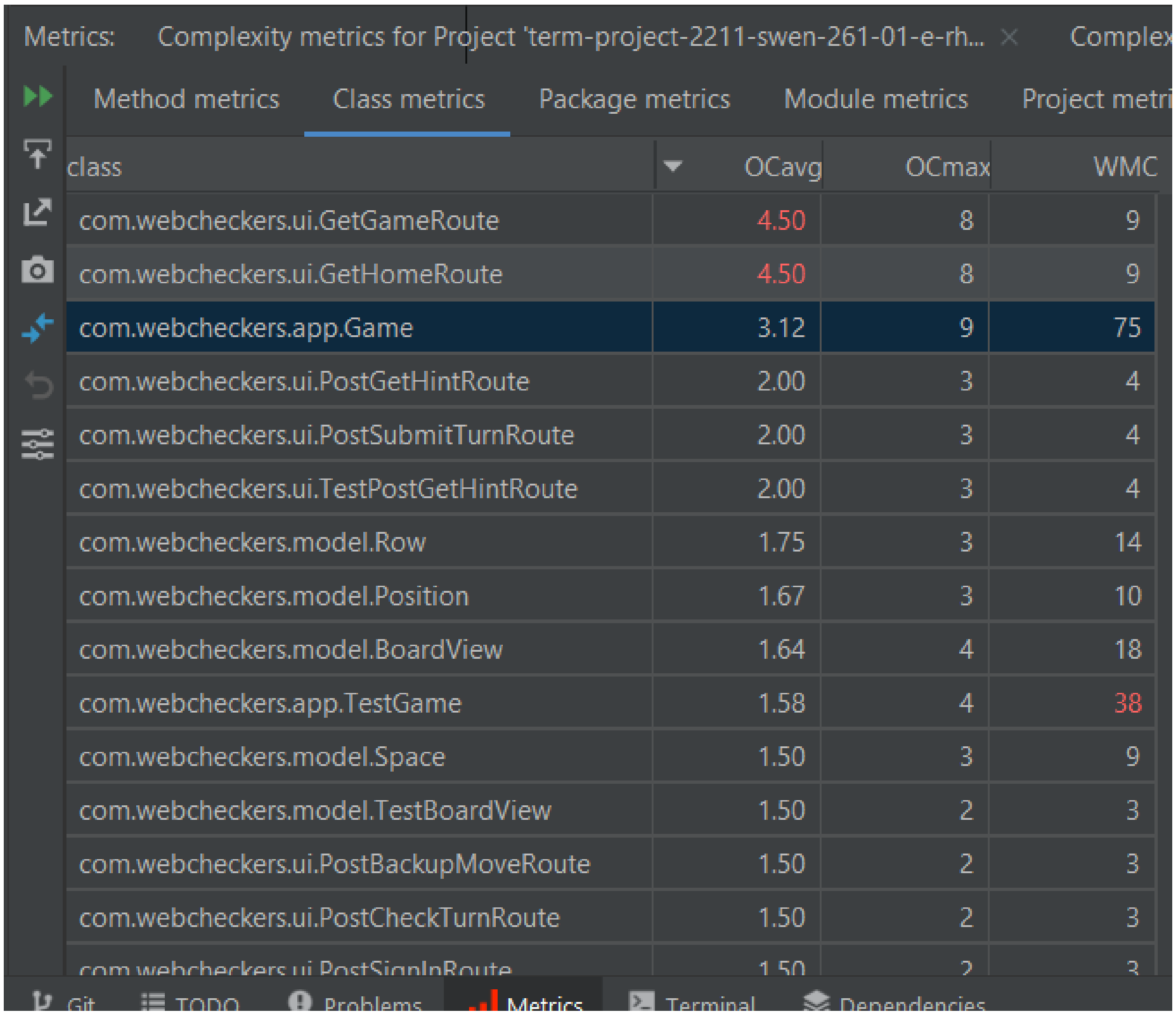
With that information, we can deduce that the only reason those 2 methods failed was that we have a big amount of comments adding to our line count, so it's different from our results. Other than that, a similar problem is in our Game class. This class has a code metric error due to the complexity, but I feel as if the error is in the metric on this one as well, and We have the proof to back it up. We took one section of the Game class and split it up into an old commented-out code metric and a new Code Metric section.

With these divisions, two helper methods were added, one for making the end positions from a given start position on a board, and making an Array List of positions surrounding each corner of the current position square. The other Helper Method would be the fact that the position a checker piece might move to is in bounds. It takes the Start and End as parameters, and makes a move class based on it, then applies the isInBounds method inside it to verify the move works and returns true if so.

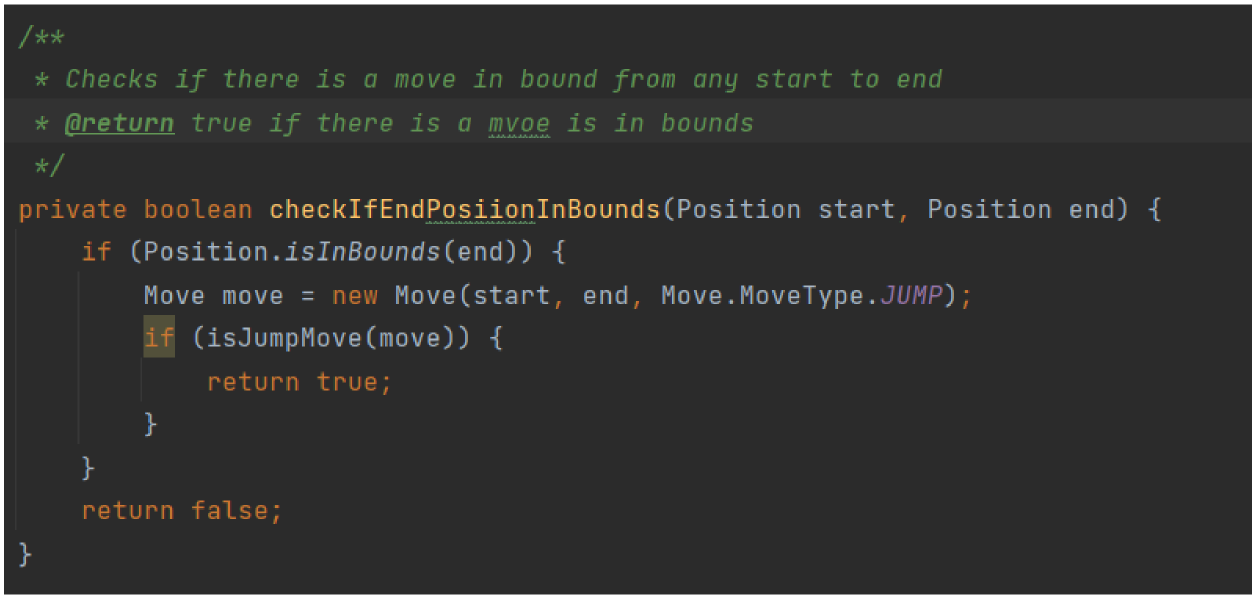
The aforementioned Helper Methods were applied, and a smaller Code Metrics score was achieved(3.22 to 3.12), which is good, but the new code made was much harder for a user to read, because it brings the user in so many directions, that we felt if it was all simplified, and it used for loops such used in the old method, it would be fine. Although the Java Compiler understands the code better, we as humans do not. I felt the current code would be sufficient, instead of making the code much larger, and less easy to follow. The time Complexity or Space Complexity has not changed either.

Another prominent error shown by the Code Metrics, is the warning sign that our gameMap in the Game class should be finalized, as that is not the case, we are changing the method, and removing data out of this Hashmap. We even have the code for changing this hashmap in the Game class itself, so there is no reason we should be getting this error whatsoever.

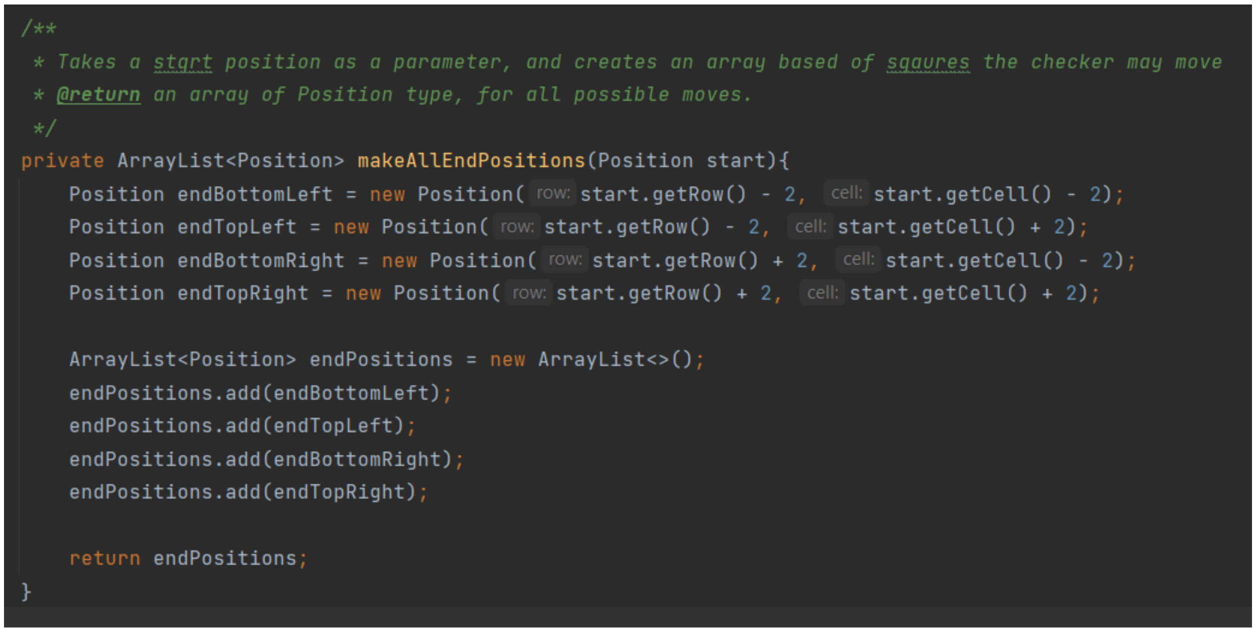
Our final Code Metric that did not meet Intellij's expectations would be the TestGame class. The Metrics that are applied to the Model tier & the test should not be the same. They are fundamentally different and do not achieve nor measure the same thing. The Model tier is supposed to be simplified to make the code perform the same but have better complexity and be easier to follow. While the somewhat opposite is true for the Testing classes. The Code is supposed to be extensive enough to measure all cases and scenarios that can apply to our code. The error we got from Game in Code Metrics can apply to its test as well. So all in all the testing module should take as many measures and does not violate any principles we learned in class, we feel as if there is an error in the metric for this one as well.



class	OCavg	OCmax	WMC
com.webcheckers.ui.GetGameRoute	4.50	8	9
com.webcheckers.ui.GetHomeRoute	4.50	8	9
com.webcheckers.app.Game	3.12	9	75
com.webcheckers.ui.PostGetHintRoute	2.00	3	4
com.webcheckers.ui.PostSubmitTurnRoute	2.00	3	4
com.webcheckers.ui.TestPostGetHintRoute	2.00	3	4
com.webcheckers.model.Row	1.75	3	14
com.webcheckers.model.Position	1.67	3	10
com.webcheckers.model.BoardView	1.64	4	18
com.webcheckers.app.TestGame	1.58	4	38
com.webcheckers.model.Space	1.50	3	9
com.webcheckers.model.TestBoardView	1.50	2	3
com.webcheckers.ui.PostBackupMoveRoute	1.50	2	3
com.webcheckers.ui.PostCheckTurnRoute	1.50	2	3
com.webcheckers.ui.PostSignInRoute	1.50	2	3



```java
/**
 * Checks if there is a move in bound from any start to end
 * @return true if there is a mvoe is in bounds
 */
private boolean checkIfEndPosiionInBounds(Position start, Position end) {
    if (Position.isInBounds(end)) {
        Move move = new Move(start, end, Move.MoveType.JUMP);
        if (isJumpMove(move)) {
            return true;
        }
    }
    return false;
}
```



```java
/**
 * Takes a stgrt position as a parameter, and creates an array based of sqaures the checker may move
 * @return an array of Position type, for all possible moves.
 */
private ArrayList<Position> makeAllEndPositions(Position start){
    Position endBottomLeft = new Position( row: start.getRow() - 2,  cell: start.getCell() - 2);
    Position endTopLeft = new Position( row: start.getRow() - 2,  cell: start.getCell() + 2);
    Position endBottomRight = new Position( row: start.getRow() + 2,  cell: start.getCell() - 2);
    Position endTopRight = new Position( row: start.getRow() + 2,  cell: start.getCell() + 2);

    ArrayList<Position> endPositions = new ArrayList<>();
    endPositions.add(endBottomLeft);
    endPositions.add(endTopLeft);
    endPositions.add(endBottomRight);
    endPositions.add(endTopRight);

    return endPositions;
}
```