Report

Zhifang Wang the University of AucklandAuckland, New Zealand  
zwan686@aucklanduni.ac.nz

# Assignment 1 review

**Worked well**:

In Assignment 1, I decided to encapsulate some of the methods in order to make them smaller. At the same time, I adopted a full English naming strategy for the methods and added comments to some of the problematic methods. This helped me a lot in completing A2. When I reviewed the code for A1, I could see exactly what each method did, which allowed me to quickly build how to complete the two new features of A2. I can reuse these already encapsulated methods in new features, which also makes me more productive.

The playGame() method in A1, which was long but clear to me through the comment and variable names, was a big help to me in implementing the 'bmf' function, I only had to focus on how the robot was selecting the houses and it didn't require me to rethink the logic of the game.

**Did not work well**:

However, through the weeks of lecture, I have found that some methods are not suitable. For example, the isGetOppositeSeed() method in A1 contains five parameters, and methods with too many parameters cause me to review my code slower and reduce the overall maintainability of the code. Also, some of my methods are not 'small,' such as the playGame() method in A1, this method has about 100 lines, which can be difficult for developers to understand. I had to go back through the code.

Also, my Board class is not good because a good class tends to be realistic. In A1, I have done all the game logic for the whole code in the Board class, which does not comply with the class naming conditions and tends to cause problems for the developers.

# Assignment 2 implementaion

In A2, I have split the Board class into PlayLogic and Board classes. To be more in line with the Java language and object-oriented programming, the PlayLogic class is only responsible for implementing the entire game logic. In contrast, the Board class is only responsible for printing the board and other functions. In A2, I needed to implement two new functions: print the board vertically and add bots. For printing the board vertically, I added a variable isVertical to the PlayLogic class and initialized the constructor's board class. Also, in the Board class, there is a printBoard() method that can determine, with parameters, if the board is to be printed vertically. For the second 'bmf' function, I think the bot added is also a kind of player, so I created the playWithRobot() method in PlayLogic, which will perform the "best first move" on request. In order to solve the problem of too many parameters in some methods in A1, I changed some variables to global variables, which effectively solved the problem of too many parameters in some methods. For example, the isGetOppositeSeed() method in the A1, Board class has 5 arguments, but after refactoring, it has been reduced to 3. In addition, for some, if statements with only one line, such as the gameOverMessage() method, I removed {}, which can more clearly meet the requirements of the lecturer. I have done this assignment more mathematically, making a judgment on each of the possible scenarios. For example, in the sowSeed() method, I have commented on each of the four scenarios, explaining what the method does.