|  |  |
| --- | --- |
| **C:\Users\Far\Pictures\New Picture.jpg** | **Code Review Template** |

**This template is to be completed and submitted by the *Reviewer*.**

Names of the Reviewers: Lotfi Hasni

Name of the developer being reviewed: Ziad Chemali (Tic-tac-toe code)

|  |  |  |
| --- | --- | --- |
| **Category** | **Comments /questions about of the reviewing group about the design documents** | **Responses by the developer (if any)** |
| Spelling Mistakes | * The word ‘opponent’ is misspelled multiple times in classes BlockingPlayer and SmartPlayer. * The word ‘starting’ is misspelled in comments for runTheGame() method of Referee class. | I should double check my spelling mistakes |
| Naming issues | * The name of the method helper() is not a clear indication of its purpose. * Inconsistent naming conventions used: camel case mixed with snake case. | I should use descriptive method names and use same naming convention |
| SOLID Principle Violations | * Single Responsibility Violation: the makeMove() method in the HumanPlayer class accepts user input, checks for (potential) winner, and communicates results. Could these responsibilities be redistributed? * The testForBlocking() and testFoWinning() methods repeat code. Could this be reworked to avoid excess code? * The testforblocking() and testFoWinning() methods print intermediate decisions to user through screen. These may unnecessarily violate single responsibility principle. * Multiple Scanner objects created in loop and are not closed after use in makeMove() method. Could pass in Scanner object as argument to adhere more closely to dependency inversion principle and reduce coupling. * Open-close violation: row and column values are hard-coded into the program. Boards of different sizes could exist. It would require the modification of existing classes and addition of code to accommodate different board sizes. * Possible open-close issue: the player and space characters ae hard-coded into the program at multiple points. Could replace with the constants defined in Constants interface to allow for potential future changes. * Single responsibility violation: HumanPlayer makeMove() method takes user input and gives output, as well as making player’s move. These responsibilities could be redistributed. | I will make sure to implement SOLID principles when writing future codes |
| Lack of documentation | * Dates for class creation missing from files; this information could potentially be important. * Missing JavaDoc comments in BlockingPlayer, SmartPlayer, RandomPlayer, and HumanPlayer classes. | Add more documentation for future codes |
| Issues with organization and formatting | * Inconsistent indentation present in BlockingPlayer class. * Large, unnecessary spaces between code lines in multiple classes. * Auto-generated method stub comments have not been removed in a number of classes. | Make sure to organize code after compiling |
| Effective design decisions | * Efficient and brief RandomPlayer class. * Clever recursive strategy to deal with error in makeMove() methods. * Good use of inheritance to increase code reuse throughout program. * Helpful algorithmic comments included in SmartPlayer and BlockingPlayer classes. |  |
| Overly complicated code | * The testForBlocking() and testForWinning() methods are too large and convoluted, sometimes involving nested loops. Could be separated into multiple methods and simplified. | Having multiple methods could help in debugging my code |
| Error-checking issues | * No error checking (try-catch) present for users entering non-integers when choosing spot to place make on boad. | I should definitely add try-catch, to catch any user’s input mistake |