UDACITY

Project Evaluation

STUDENT

Gregory Rermgosakul

COURSE

Intro to Java Programming

Gregory,

Your project meets specifications! You fixed the diagonal issue, and that is clearly visible in your agent's win percentages. I've left a few notes regarding efficiency and encapsulation for you to think about. One thing all programmers learn is that there is always room to refactor and improve their code. I encourage you to look back on this project and all the code you wrote in this course, and I think you'll be pleased to find how readily you can think of improvements given how much you have learned. Congratulations on a successful project, and I hope you enjoyed the course!

-Louis

How helpful was this feedback? Click here to tell us about it.

Code Functionality — Meets Specifications

- The code compiles and runs without throwing any Exceptions or Errors. These include but are not limited to NullPointerExceptions, Compile errors, Runtime errors, etc.
- The agent has a functional move method.
- The agent recognizes and responds appropriately to available moves that would cause either player to win the game.
  - Your agent now finds diagonal 4-in-a-row's! Good job.
- The agent can reliably win the game (The agent beats the Random Agent at least 18 times out of 20 and the Beginner Agent at least 12 times out of 20). Your agent wins:
  - against Random Agent about 96% of the time. (Great!)
  - against Beginner Agent about 81% of the time. (Awesome!)
  - against Brilliant Agent about 50% of the time, which is still pretty good!

Use of Control Flow Statements — Meets Specifications

- Appropriate control statements (for, while, if, else) are used in each relevant situation.
- Iterative control statements (for, while) are used effectively to avoid repetitive code.
- Conditional control statements (if, else) effectively used to guide code flow.

Definition and Use of Methods

Meets Specifications

- Repeated blocks of code are encapsulated in methods.
  - Good use of one method to implement both iCanWin and theyCanWin, and good use of the checkIfEqual helper method.
  - Checking every direction individually in the anyonesGame method makes it quite long, and I think you could have either used some clever loops or encapsulated some of those checks into other methods, e.g. a checkDiagonal method that takes a direction as a parameter and checks for 4-in-a-rows along that axis. This would shorten your code and also make it more readable, but your method totally works fine!
- Methods defined in the program are called where appropriate.
- Parameters and return values are appropriate for each method's purpose.
- Use static and instance members of a class properly.

Object Oriented Programming

Meets Specifications

- Code correctly uses accessor methods to get game data out of the board.
- Code correctly uses mutators to modify the game board.
- Code efficiently and effectively leverages the object-oriented class structure of the game.

Code Readability

Meets Specifications

- Naming conventions from the Java Language Coding Guidelines are used:
  - Variable and method names are lowercase, with occasional upperCase characters in the middle.
  - Class names start with an Uppercase letter.
  - Constant names are UPPERCASE, with an occasional UNDER_SCORE, and are declared as final.
- Method documentation is clear and concise and is included for each method.
  - Javadocs present for all methods! Awesome.
- Names of methods and parameters are descriptive and meaningful.
- Magic numbers are avoided. final variables are defined for numeric constants used in the program.

PROJECT EVALUATION

Meets Specifications