Week 7 Assignment - OOP Review and Best Practices

Overview

This week we reviewed Object Oriented Programming concepts, as well as went over best practices associated with OOP. For this week's assignment, you will be implementing a simple game called Pig. Details about the game can be found here. Your job will be to code a functional application that implements this game for two players. Next week, we will expand on this assignment and allow the computer to play, using a certain strategy.

Make sure to create a github repository for this assignment named **IS211_Assignment7**. All development should be done in this repository.

Useful Reminders

1. Read the assignment over a few times. At least twice. It always helps to have a clear picture of the overall assignment when understanding how to build a solution.
2. Think about the problem for a while, and even try writing or drawing a solution using pencil and paper or a whiteboard.
3. Before submitting the assignment, review the "Functional Requirements" section and make sure you hit all the points. This will not guarantee a perfect score, however.

Rules of The Game

The rules of Pig are simple. The game features two players, whose goal is to reach 100 points first. Each turn, a player repeatedly rolls a die until either a 1 is rolled or the player holds and scores the sum of the rolls (i.e. the *turn total*). At any time during a player's turn, the player is faced with two decisions:

- **roll** - If the player rolls a
    - **1**: the player scores nothing and it becomes the opponent's turn.
    - **2 - 6**: the number is added to the player's turn total and the player's turn continues.
- **hold** - The turn total is added to the player's score and it becomes the opponent's turn.

The key to the game is this: on every roll, a player faces a decision about how large a turn total should be risked to possibly get an even larger total. You can play this online (requires java) to get a feel for the game.

Design

Now it is time to stop, and think about the design of the program. You could certainly write this assignment using no classes, but would clearly defeat the purpose of the assignment. How would you represent the game as a collection of objects? What entities do you think are in the game that should be modeled as an object? The design for this program will be up to you, but your code must implement the game fully as described above and in the functional requirements.

To help guide your design, you should think about the following entities:

- Players: in this case, we only need two (human) players per game, but how would we model a single player? What properties should it have and what actions should it have to implement?
- The Die: for this variant of Pig, we are using a single die. How can we model this as an object?

- How do we model the state of the game itself? For example, we need to track the players, their scores, and the die being used for the game.

Testability

To make your code easier to test for correctness, please remember:

- When using any random features in your code, make sure to use a random seed. To keep everyone consistent, use a seed of 0 at the beginning of your code.
- When asking the human for a decision, you should accept the letter ‘r’ to indicate a roll, or a ‘h’ to indicate the player wants to hold
- When the game is over, the program should stop. To run another game, you need to run the program again.

Functional Requirements

- The game should continue until one of the players has a score that is greater than or equal to 100
- At every step, the program should output to the player what was rolled, the current score for this turn so far, and their total score in the game before asking for a decision.
- Every rule listed in the “Rules Of The Game” section should work (i.e., rolling a one ends a player’s turn with no points added to their score)

Extra Credit

For extra credit, you can:

- Implement the same game as above but that allows for a configurable number of players (via a command line parameter called `--numPlayers`)
- Allow for more than one game at at time. Remember, at the end of a game, the scores and the game state need to be reset