**ECCS 1621 – Programming 2**

**Spring Semester 2021**

**MP2 – Revisiting Roll’em Pigs!**

**Part A Due Monday 22 February 2021 (i.e., Pre-Lab 06)**

**Part B Due by Tuesday 2 March 2021**

The game of Pig is a simple two player dice game in which the first player to reach 100 or more points wins. Players take turns. On each turn, a player rolls a pair of six-sided dice. After each roll, the die values are examined:

* If neither die shows a 1, the sum of the dice is added to the current turn total. The player can now elect to roll again or to stop. If the player chooses to stop, then the sum of all of the rolls for the current term (i.e. the current turn total) is added to the player’s game total; this is the only time that points are added to the game total. If the player rolls again, the both the current turn and game totals are at risk of being lost.
* If a single 1 is rolled, the player’s score for the current turn is lost (i.e., it is set to zero) and the turn ends.
* If two 1’s are rolled, the player’s score for the entire game is lost (i.e., it is set to zero) and the turn ends.

You are to write a Java application that plays the game of Pig for two human players. Use input dialogs to get the names of the two players; each name should be displayed on a label along with additional labels displaying each player’s (total) game scores. Provide two buttons to control the turn: “roll” and “stop”. Use color changes between player labels to indicate whose turn it is. Provide a label that can be employed to provide messages to the players, such as what the current roller’s turn score is. Use what was learned in Lab 5 Part B to create a Panel with a GridLayout (Swing) for displaying the dice. When a game is won, update the score display and then present a confirm dialog (Swing) that congratulates the winner and asks if the contestants wish to play another game. Please see the next page for an example layout of the application. As part of your application, you are required to write a separate Player class (i.e., separate file) that contains **at least** the following methods:

* int rollDice( JLabel die1, JLabel die2 )

The rollDice method will perform the rolling of the individual die being displayed on each of the two JLabel objects; (the returned value are to be used to update the game and turn scores, as well as the status variables for the boolean-returning methods below). The int value being returned in the sum of the die.

* boolean isTurnScoreLost()

Returns true if the result of the roll causes the player to lose the turn, false otherwise.

* boolean isGameScoreLost()

Returns true if the result of the roll causes the player to lose both the turn and all points accumulated in the game, false otherwise.

* boolean hasWon()

Returns true if the player has scored at least 100 points, false otherwise.

* boolean addTurnScoreToGameScore()

This method is to be called ONLY when a player finishes his/her turn. Returns true if the value of the game score changes as a result of the operation.

* int getTurnScore()

Returns the current turn score.

* int getGameScore()

Returns the current game score.

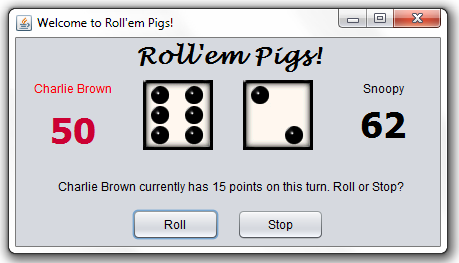
* void reset()

Resets the player’s turn and game scores to zero, primarily for use in starting a new game.

**Notes:**

* You’ll need to make two instantiations of the Player class – one for each player.
* Think about how you would play this, or a similar dice game, with several of your friends. Do each of you have his or her own pair of dice, or do you share one pair of dice? The rollDice() method allows each player to share the dice displayed on the GUI; you should NOT be instantiating JLabels objects as part of a Player instantiation.
* You might want to consider making changes to the DieGame class (you created this in lab 05) so that the programmer can turn off the ability of the player to click and hold the value of one or both dice. Another approach is to always pass the value false to the holdDie() method just prior to invoking rollDie() as this will clear any holds that a cheating player might attempt to try.

**Example Application Layout – please note that you should not have to copy this verbatim!**



**Deliverables:**

Monday, 22 February: PART A / Pre-Lab 06

* Screenshot of GUI (but functionality not required)
* The Player Class completed

Tuesday, 2 March: PART B

* All code files – make sure you have all the required Player methods
  + Source code for Player class
  + Score code for the GUI (i.e., both the .java file and the .form file)
  + Any other classes you created to solve the MP #2
* Javadoc generation that includes the public interface for the Player class
* Screenshots of “in use” application, similar to that shown above (see scoresheet for details)