**Grading Rubric for Lab 2, Pig Game v1**

|  |  |
| --- | --- |
| Textbook Exercises | **10/10 Points** |
| *Criteria* |  |
| Exercise 4-1: Are there screen shots of the Tip Calcualtor running? | 5/5 |
| Exercise 5-1: Are there screen shots of the Tip Calcualtor running? | 5/5 |

|  |  |
| --- | --- |
| **Release Version** | **30/30 Points** |
| *Criteria* |  |
| Can both players enter their names, and is the correct name displayed when it is that player’s turn? | 5/5 |
| Does the roll dice button cause random die images (1 through 6) to be displayed? | 5/5 |
| Does the die value get added to the score for the turn? | 2/2 |
| If a player rolls a 1 is the score for the turn set to zero and the turn ended? | 2/2 |
| Can a player press the end turn button to end their turn (before getting a zero)? | 2/2 |
| Are the points for the turn added to each player’s score at the end of the turn? | 2/2 |
| Is a winner correctly determined after one or the other player reaches 100? | 2/2 |
| Does the player who starts second get to complete their turn if the first player reaches 100 before they do? (then the winner will be the one with the most points) | 3/3 |
| Are all scores reset when the New Game button is pressed? | 2/2 |
| Are there two layouts, one for landscape and one for portrait? | 5/5 |
| <u>Optional, extra credit</u>: is there a frame layout with a background image? | 3/3 |

|  |  |
| --- | --- |
| **Code Quality** | **10/10 Points** |
| *Criteria* |  |
| Does the app run without throwing exceptions or crashing? | 2/2 |
| Is all the game-play logic in a separate class? | 5/5 |
| Is the game-play class written with good programming practices: private instance variables modified by getters and setters, constructor for initializing state, no dependencies on UI code, single return per method, no duplicated code blocks, etc. (one point deducted for each “no”) | 2/2 |
| Is proper coding style used? | 1/1 |

**Criteria for coding style**

* Proper indentation is used.
* UI elements are named according to convention (example: incrementButton).
* Java variables, constants, methods, and classes are named using descriptive names (Example: *int count*, not *int i*).
* Unnecessary lines of code or files been removed (no commented out code or left-over boiler-plate).
* Explanatory comments are in the code (Coments that explain why you did what you did).
* Variable names and methods use camel case (Example: camelCase).
* Constants are used where needed and named using ALL\_CAPS.
* Classes are named using PascalCase (aka TitleCase).