**Programming 2 – Spring 2020** Student Name: \_\_\_Liam Martell\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**MP2 Scoresheet – 2 March 2021 (must be filled out prior to submission)**

**Source Code and Testing – to be turned in by END OF LAB 3/2/21** lab section: morning afternoon

1. Adherence to specifications (5 point **DEDUCTION** for each violation):

|  |  |
| --- | --- |
| **CODE REVIEW** | |
| **Method present?** | **Player class - method and exact signature** |
| **YES** | no | **int rollDie( DieLabel die1, DieLabel die2 )** |
| **YES** | no | **boolean isTurnScoreLost()** |
| **YES** | no | **boolean isGameScoreLost()** |
| **YES** | no | **boolean hasWon()** |
| **YES** | no | **boolean addTurnScoreToGameScore()** |
| **YES** | no | **int getTurnScore()** |
| **YES** | no | **int getGameScore()** |
| **YES** | no | **void reset()** |
| **JAVADOC GENERATION** | |
| **YES** | no | Javadoc for all public Player methods present and generated at time of testing |

**Other violations:**

**YES** | no - screenshots of application “in use” turned in with source code (-5 for each one missing)

**(Screenshots to be submitted will be outlined in the next section).**

1. Demonstration of application – to be submitted via screenshots of your code running. 5 points each unless otherwise noted. 50 points total.

* Program starts with one or more input dialogs asking for the names of the players Y | N score: \_\_\_\_\_
* Names of both players displayed on application along with their game score Y | N score: \_\_\_\_\_
* Player’s turn indicated by color changes between player labels Y | N score: \_\_\_\_\_
* Program successfully detects player change due to loss of turn Y | N score: \_\_\_\_\_
* Program successfully detects player change due to banking turn points Y | N score: \_\_\_\_\_
* Game ends when a player scores 100 or more points Y | N score: \_\_\_\_\_
* Correct score displayed when end of game is reached Y | N score: \_\_\_\_\_
* Confirm dialog used to ask whether or not to play another game Y | N score: \_\_\_\_\_

***Everything correct? Submitted correct?* YES (10 points) | no** score: \_\_\_\_\_

score: \_\_\_\_\_ /50 points

Comments:

**Testing/Validation Subtotal: \_\_\_\_\_\_\_\_\_ / 50**

**Readability Subtotal: \_\_\_\_\_\_\_\_\_ / 20**

**Documentation Subtotal: \_\_\_\_\_\_\_\_\_ / 30**

**SPECIFICATION DEDUCTIONS: < \_\_\_\_\_\_\_\_\_\_ >**

**TOTAL: \_\_\_\_\_\_\_\_\_ / 100**

**Programming 2 – Spring 2021**  Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**MP2 Scoresheet – 2 March 2021**

**Source Code Analysis Rubric** lab section: morning afternoon

**Readability – 20 points total**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Meets Expectations - 5** | **Needs Minor/Major Improvement – 4/3/2** | **Unacceptable/Missing - 0** |
| Organization  Score (x1): \_\_\_\_\_\_\_ | Code is broken down into clear, recognizable, well thought out sections of functional units; blank lines and comments used to establish visual structure. |  |  |
| Separation  Score (x1): \_\_\_\_\_\_\_ | Spaces used as appropriate to help differentiate distinct elements within each coding statement. |  |  |
| Consistency  Score (x1): \_\_\_\_\_\_\_ | Similar coding constructs regularly use the same format regarding indentation and alignment; similar or related variable names follow an established pattern. |  |  |
| Grammar  Score (x1): \_\_\_\_\_\_\_ | All comments employ proper sentence structure, capitalization, word choice, and punctuation. No spelling errors noted. |  |  |

**Documentation – 30 points total**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Meets Expectations - 5** | **Needs Minor/Major Improvement – 4/3/2** | **Unacceptable/Missing - 0** |
| Class Header Comments  Score (x1): \_\_\_\_\_\_ | Every class file starts with a header comment that contains the name of the file, the date of its writing, the full name of its author, and a description of what the class does. |  |  |
| Method Header Comments  Score (x2): \_\_\_\_\_\_\_ | Description clearly but succinctly explains purpose of the function.  Preconditions and postconditions as specified as appropriate. |  |  |
| Javadoc comments  Score (x1): \_\_\_\_\_\_ | Javadoc appropriately used with all public methods for the classes within the project. |  |  |
| Section Comments  Score (x1): \_\_\_\_\_\_\_ | Each functional section of code includes a comment describing the goal or purpose that that section is trying to accomplish without being either verbose or parroting. |  |  |
| Code Comments  Score (x1): \_\_\_\_\_\_\_ | Line-oriented comments are used to clarify meaning and/or provide elaboration as needed. |  |  |