**CMSC 2133**

**Object Oriented Programming**

**Project 1 – Dice Roller**

**Points: 50 pts**

**Assignment**

Create software that meets the requirement of the story, passes all the acceptance criteria and meets the technical requirements listed.

**Version 1.1 - 20 points**

* Problem Domain Objects
* AC 1-3 for Problem Domain services
* Review in Lab to Turn in
  + Source Code
  + Object Model
  + Demo Passing Tests

**Version 1.2 - 30 points**

* Problem Domain Objects
* AC 4-8 for Problem Domain services
* Review in Lab to Turn in
  + Source Code
  + Java Docs
  + Object Model
  + Demo Passing Tests

**Objectives:**

**The student will demonstrate the use of these CS Principles by the successful completion of the software for this project.**

* Encapsulation
* Abstraction
* Polymorphism
* Re-use
* Separation of Concerns

**The student will demonstrate the use of these CS Skills by the successful completion of the software and documentation for this project.**

* UML OO Modeling in Visual Paradigm
* OOP Programming in Java in Eclipse
* Incremental Development

**The student will demonstrate the application of this CS Knowledge by the successful completion of the software and documentation for this project**

* UML
  + Class diagrams
* OOP Concepts/Modeling
  + Classes - Chapter 5 – Classes
  + Attributes
  + Methods (messages)
  + Separation of Concerns
* Java
  + Arrays of Objects
  + Iterators - Chapter 16
  + Basic Swing - Chapter 6 – UI
  + Exception Handling - Chapter 10
  + Packages
* Eclipse
  + Use
  + Packages
  + Debugging
  + Java Docs
* Problem Specification
  + Stories
  + Acceptance Criteria
* Iterative Development
  + Small stories
  + Add on stories
* Book
  + Chapter 5 – Classes
  + Chapter 6 – UI
  + Chapter 7 – Arrays of Objects
  + Chapter 10 – Exceptions
  + Chapter 16 – Iterators

**Main Story**

As a game player, when I need a dice roll, I want to view total of the faces on the specified number of dice using the specified number of faces so that I can use the total in my game playing.

**Acceptance Criteria**

**AC #1**

**Given**: I am ready to roll

**When**: I enter the number of dice and the number of faces

**Then**: I can view the value of each face rolled and the total of all faces rolled.

**AC #2**

**Given**: I have specified the number of faces on the die

**When**: I roll the die face

**Then**: the value for result is between 1 and the number of faces specified.

**AC #3**

**Given**: I have rolled the dice 100 times

**When**: I calculate the frequency of each face value

**Then**: the the frequency of each possible face value is the same

**AC #4**

**Given**: I am ready to roll

**When**: I start the program

**Then**: I can view a screen with fields for entry of number of faces and number of dice and a button to click to roll.

**AC #5**

**Given**: I am ready to roll

**When**: I enter the number of dice and the number of faces and click on roll

**Then**: I can view the value of each face rolled and the total of all faces rolled.

**AC #6**

**Given**: I have specified the number of faces on the dice and the number of dice to roll and rolled the dice

**When**: I view the total

**Then**: The total is equal to sum of the faces displayed for each dice

**AC #7**

**Given**: I have specified the number of faces on the dice and rolled the dice

**When**: The number specified is not greater than 0

**Then**: An error message is displayed and the total is set to 0.

**AC #8**

**Given**: I have specified the number of dice and rolled the dice

**When**: The number specified is not greater than 0

**Then**: An error message is displayed and the total is set to 0.

**Technical Requirements:**

1. Write program in Java using Eclipse IDE.
2. Do incremental development.
3. Create UML Object Model
4. Create Java Docs for the software.
5. Apply Abstraction by creating objects that represent the real world items.
6. Apply Encapsulation by creating objects in a PD model.
7. Apply Reuse by using the same PD objects with different UIs.
8. Apply Separation of Concerns by putting HI Classes and PD Class in separate packages.
9. Apply Separation of Concerns for error checking by not doing I/O in PD classes.
10. Apply Polymorphism by using the same method name for the dice and collection of dice for rolling the dice.
11. Out put the results of tests.