**Lab 2 – ADT Implementation and Use**

**CSC 3302**

|  |  |
| --- | --- |
|  |  |
| **Introduction:** | The concept of a class as a representation of an ADT is central to object-based and object oriented computing. Java provides direct language support for ADTs through the class declaration, and object instantiation. In addition, these constructs provide the basis for reuse. |
|  |  |
| **Description:** | Design and implement an ADT (class) **Dice**. Each instance of this class should have as attributes the die’s current side (number of dots), the total number of times it has been rolled, and the total for all the sides it has displayed. These attributes are private and should only be modified through the class interface (methods).  There should be *observer* functions for all of these attributes. Two other functions, **roll** and **reset** are the *modifier* methods. You may choose to have other methods in the class. Develop the class for dice and test it in a simple tester program.  You should end up with at least 2 .java files: **p2.java** and **Dice.java** |
|  |  |
| **Program:** | Implement a game called *Aces in the Pot*. This is a dice game designed for 4-10 players. You need two dice, and two coins for each player. The object of the game is to have the last coin left, and then to avoid rolling a 6 during three throws in a row.  Decide who starts by rolling both dice; highest is the starting player and the turn order then proceeds clockwise.  In a turn, the player rolls both dice. If a 1 is rolled, the player puts one of the coins in the pot. (In American dice parlance a 1 is called an Ace, thus the name of the game.) If a 6 is rolled, the player gives one of the coins to the player on the left. Any other roll has no effect.  If two 1’s are rolled, both coins must go into the pot. If two 6’s are rolled, both coins are given to the player on the left (if there are two coins left; otherwise, give the one coin). The pot takes precedence. If a player has only one coin left and rolls 1-6, the one coin goes into the pot.  If a player has lost both of his coins, he may not roll the dice on his turn. The player is, however, still in the game – the player on his right may roll a 6 and have to give him a coin.  The game continues in this manner until there is only a single coin left which has not gone into the pot. The player with that coin takes the dice and rolls three times in a row. If a 6 is not rolled during these three rolls, he wins the pot. But if a 6 is rolled, the turn passes to the player on the left (who now has the remaining coin) and tries to roll three times without getting a 6.  The winner is the player who manages to have the coin and roll three times in a row, without getting a 6. He or she then sweeps the entire pot.  Design and implement a machine that allows the user to play *Aces in the Pot*. This **machine** should be defined as a client that uses the **Dice** class. The user interface should prompt for and accept the number of players (between 4 and 10, inclusive). It should then begin the game, displaying the results of each roll. At the end of the game, the program should display the winner and dice statistics. |
|  |  |
| **Input:** | Your program should prompt for and read in the number of players. |

|  |  |
| --- | --- |
| **Output:** | Produce the output as required by the program. All actions should provide a response. |
|  |  |
| **Example:** | Enter the number of players –  >>**4**  Player 1 rolls –  Die 1: 3  Due 2: 4  Player 2 rolls –  Die 1: 6  Due 2: 4  Player 3 rolls –  Die 1: 4  Due 2: 1  Player 4 rolls –  Die 1: 5  Due 2: 6  Player 4 goes first.  Player 4 rolls –  Die 1: 6  Due 2: 4  Player 4 gives 1 coin to Player 1.  Player 1 rolls –  Die 1: 1  Due 2: 1  Player 1 gives 2 coin to the Pot.  Player 2 rolls –  Die 1: 1  Due 2: 6  Player 2 gives 1 coin to Player 3.  Player 2 gives 1 coin the Pot.  Etc….  Game over.  Player 1 wins.  Dice statistics:  Die 1: rolled x times, average number = 2  Die 2: rolled y times, average number = 2.75 |
|  |  |
| **Hints:** | Use appropriate software design techniques, and implement the class and methods with the Java constructs for I/O, declarations, and manipulations. You should have at least one class with your Dice class declarations and methods and a file called p2.java containing the main routine (one page maximum). You may choose to implement another class, such as a Player class, if you so desire.  Build your program in steps (i.e., get the input and output working, then add the evaluation functions, then error checking, etc.). Emphasize functionality first, then add the advanced features. Work on this program over time, DO NOT wait until the last minute (i.e., the day before it is due) to start! |