1 Objectives

1. Practice using class methods
2. Practice using class variables
3. Practice object-oriented programming

2 Tasks

1. Create a Java project called DiceRoll.
2. Create a class inside your project called Dice.
3. Create a class inside your project called Roll.
4. Create a class inside your project called Main.
5. Add the following two attributes to your Dice class:
   
   (a) private int value which will hold the current value of the dice (the top-side after being rolled).

   (b) private int sides which will hold the number of sides on the dice.

6. Modify the two attributes so that they are initialized to have value 1.
7. Add the following methods to your Dice class and implement them as described:

```java
/**
 * Assigns 's' to the 'sides' attribute, to denote the number of sides of the die.
 * Throw an exception if the parameter is not one of the numbers 4, 6, 8, 10, 12, 20
 */
public void setSides(int s){
    // your code here
}
```
* Returns the current value of the die
*/
public int getValue(){
    // your code here
}

/**
 * Assigns to the 'value' attribute a random integer
 * from 1 to 'sides'. Note that this does not return
 * the value of the roll, just re-generates the random
 * number.
 */
public void roll(){
    // your code here
}

8. In your Roll class, add an attributes to hold an array of Dice objects: private Dice[] diceArr;.

9. In the Roll class, create a default constructor that initializes the array to hold two Dice members that are six-sided dice.

   (a) **Hint:** you will have to use the setSides(int) function for the Dice since sides is a private attribute! This means that you’ll have to “propagate” the possible Exception upward with throws in the function definition, or use try..catch within the constructor. Be sure to document how your code will behave in this case, that is, is calling the Roll constructor safe because it will catch and handle exceptions, or is it volatile because it might just pass them upward to the calling function?

10. Add the following methods to the Roll class and implement them as the comments describe:

    /**
     * Re-randomizes each of the member Dice objects
     * by using their .roll() method
     */
    public void roll(){
        // implement the function here
        // for debugging, print out the resulting values after
        // rolling the dice.
    }

    /**
     * Returns the sum of the values of the Dice objects
     */
    public int getTotal(){
        // implement the function here
    }

11. In the Roll class, add a non-default constructor which takes two int arguments: one to denote how many dice are being rolled and another to set the number of sides of each Dice element
in the `diceArr` array. Throw an exception if the number of dice is not a positive number. Similarly, handle the possibly invalid number of sides as discussed above.

12. Add a `main` function to the `Main` class that

(a) Show an example of rolling two six-sided dice by using the default `Roll` constructor.

(b) Until the user decides to stop:

i. Ask the user for a number of dice and a number of sides

ii. Check for validity (using your exception handling)

iii. Create the `Roll` object using the parameters specified

iv. Roll the dice and report the total.

3 Turn In

1. Upload each “.java” file to the Canvas assignment for Lab 4. Be sure to comment your code!