DAY 2: INTRODUCTION TO INHERITANCE, OVERRIDING, AND OVERLOADING

The Randomizer interface:

```java
public interface Randomizer {
  // Getters
  public int getPossibleOutcomes();
  public int getCurrentValue();
  public String getCurrentFace();

  // Setters (or Mutators)
  public void randomize();
}
```

The Coin Randomizer Class:

```java
// Class Name and Interface(s) Implemented by the Class
public class Coin implements Randomizer {
// Instance Variables
  private boolean isHeads;

// Constructors
  public Coin() {
    isHeads=Math.random()<.5;
  }
// Getters
  public int getPossibleOutcomes() {
    return 2;
  }
  public int getCurrentValue() {
    if (isHeads==true) return 1;
    return 0;
  }
  public String getCurrentFace() {
    if (isHeads==true) return "Heads";
    return "Tails";
  }
  // Overriding an inherited method from the Object Class
  public String toString() {
    return "The coin is showing "+getCurrentFace();
  }
// Setter(s) or Mutator
  public void randomize() {
    isHeads=Math.random()<.5;
  }
}
```

The Coin Runner Class:

```java
public class CoinRunner {
  public static void main(String[] args) {
    Coin myCoin = new Coin(); // Instantiate a Coin Object
    System.out.println("Initial Value="+myCoin.getCurrentFace());
    int countHeads=0;
    for (int i=0; i<10; i++) {
      myCoin.randomize();
      if (myCoin.getCurrentValue()==1) {
        countHeads++;
      }
      System.out.println(myCoin);
    }
    System.out.println("Heads="+countHeads);
  }
}
```

The D6 Class:

```java
public class D6 implements Randomizer {
  // Instance Variables
  private int sideUp;
  // Constructor
  public D6() {// Default (a.k.a. No Parameters Constructor)
    randomize();
  }
  // Getters
  public int getPossibleOutcomes() {
    return 6;
  }
  public int getCurrentValue() {
    return sideUp;
  }
  public String getCurrentFace() {
    return ""+sideUp;
  }
  // Overriding Inherited Method from the Object Class
  public String toString() {
    return "d6="+getCurrentFace();
  }
  // Setters (or Mutators)
  public void randomize() {
    // Cast double into an int
    sideUp=(int)(Math.random()*6)+1;
  }
}
```

The DiceRunner class (Day 2, Parts I & II):

```java
public class DiceRunner {
  public static void main(String[] args) {
    D6 die=new D6();
    System.out.println("Initial Value="+die.getCurrentFace());
    /* Roll the die 10 times */
    for (int i=0; i<10; i++) {
      die.randomize();
      System.out.println("Roll #"+(i+1)+", "+die);
    }
  }
}
```

The DiceBagRunner class (Day 2, Part III):

```java
public class DiceBagRunner {
  public static void main(String[] args) {
    D6[] dice=new D6[3];
    dice[0]=new D6();
    dice[1]=new D6();
    dice[2]=new D6();
    for (int rollNum=1; rollNum<=10; rollNum++) {
      System.out.print("Roll #"+rollNum+": ");
      int diceTotal=0;
      for (int dieIndex=0; dieIndex<dice.length; dieIndex++) {
        D6 die=dice[dieIndex];
        die.randomize();
        diceTotal=diceTotal+die.getCurrentValue();
        System.out.print(die+", ");
      }
      System.out.println(" total="+diceTotal);
    }
  }
}
```

The PolyhedralDie class:

```java
public class PolyhedralDie implements Randomizer {
   // Instance Variables
   private int numberOfSides;
   private int sideUp;

   // Constructor Methods
   public PolyhedralDie() { // Default Constructor
      numberOfSides=6;
      randomize();
   }
   // Overloaded Constructor
   public PolyhedralDie(int setNumberOfSides) {
      numberOfSides=setNumberOfSides;
      randomize();
   }
   // Getter Methods
   public int getPossibleOutcomes() {
      return numberOfSides;
   }
   public int getCurrentValue() {
      return sideUp;
   }
   public String getCurrentFace() {
      return ""+sideUp;
   }
   // Overridden Inherited toString() method from the Object class
   public String toString() {
      return "d"+getPossibleOutcomes()+"="+getCurrentFace();
   }
   // Setter Methods (or Mutator)
   public void randomize() {
      sideUp=(int)(Math.random()*numberOfSides)+1;
   }
}
```

The DiceBagRunner class (Day 2, Part IV):

```java
public class DiceBagRunner {
   public static void main(String[] args) {
      // Creating the dice array filled with null values
      PolyhedralDie[] dice=new PolyhedralDie[3];
      // Instantiating a Die in the dice array with the Default Constructor
      dice[0]=new PolyhedralDie();
      // Instantiating Dice in the dice array with the Overloaded Constructor
      dice[1]=new PolyhedralDie(12);
      dice[2]=new PolyhedralDie(20);
      for (int rollNum=1; rollNum<=10; rollNum++) {
         System.out.print("Roll #"+rollNum+": ");
         int diceTotal=0;
         for (int dieIndex=0; dieIndex<dice.length; dieIndex++) {
            PolyhedralDie die=dice[dieIndex];
            die.randomize();
            diceTotal=diceTotal+die.getCurrentValue();
            System.out.print(die+", ");
         }
         System.out.println(" total="+diceTotal);
      }
   }
}
```