DAY 5: MAKING THE GAME OF BOGGLE!

The Randomizer interface (from Day 1): The Interface For All Randomizers

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

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

The AbstractRandomizer class (from Day 4, Part I) – Superclass to PolyhedralDie

```java
public abstract class AbstractRandomizer implements Randomizer {
   private int sideUp;

   // Getters
   abstract public int getPossibleOutcomes();
   public int getCurrentValue() {
      return sideUp;
   }
   abstract public String getCurrentFace();

   // Setters (or Mutators)
   public void randomize() {
      sideUp=(int)(Math.random()*getPossibleOutcomes())+1;
   }
}
```

The PolyhedralDie class (from Day 4, Part III) – Superclass to LabeledPolyhedralDie

```java
public class PolyhedralDie extends AbstractRandomizer {
  // Instance Variables
  private int numSides;

  // Constructor Methods
  public PolyhedralDie() {
    numSides=6;
    randomize();
  }
  public PolyhedralDie(int setNumSides) {
    numSides=setNumSides;
    randomize();
  }
  // Getter Methods
  public int getPossibleOutcomes() {
    return numSides;
  }
  public String getCurrentFace() {
    return Integer.toString(getCurrentValue());
  }
  public String toString() {
    return "d"+numSides+"="+getCurrentFace();
  }
}
```

The LabeledPolyhedralDie class (from Day 4, Part III) – Superclass to BoggleDie

```java
public class LabeledPolyhedralDie extends PolyhedralDie {
  // Instance Variables
  private String[] sideLabels;

  // Constructor Methods
  public LabeledPolyhedralDie() {
    super();
    sideLabels=new String[getPossibleOutcomes()];
    for (int i=0; i<sideLabels.length; i++) {
      sideLabels[i]=Integer.toString(getCurrentValue());
    }
  }
  public LabeledPolyhedralDie(String[] setSideLabels) {
    super(setSideLabels.length);
    sideLabels=new String[getPossibleOutcomes()];
    for (int i=0; i<sideLabels.length; i++) {
      sideLabels[i]=setSideLabels[i];
    }
  }

  // Getter Methods
  public String getCurrentFace() {
    return sideLabels[getCurrentValue()-1];
  }
  public String toString() {
    return super.toString()+"("+getCurrentValue()+")";
  }
}
```

The BoggleDie class (Day 5, Part II) – Making A Randomizer For Boggle

```java
public class BoggleDie extends LabeledPolyhedralDie {
  public static String[] lettersToArray(String letters) {
    String[] arr={"?","?","?","?","?","?"};
    int labels=letters.length();
    if (labels>arr.length) {
      labels=arr.length;
    }
    for (int i=0; i<labels; i++) {
      arr[i]=letters.substring(i, i+1).toUpperCase();
    }
    return arr;
  }
  public BoggleDie(String letters) {
    super(BoggleDie.lettersToArray(letters));
  }
  public String toString() {
    String s=getCurrentFace();
    if (s.equals("Q")) {
      s=s+"u";
    }
    return s;
  }
}
```

The BoggleBoard class (Day 5, Part III) – Making the Boggle Device

```java
public class BoggleBoard {
   // Instance Variables
   private BoggleDie[][] boggleDice;
   private int rows, cols;

   // Constructor Methods
   public BoggleBoard() {
      rows=4;
      cols=4;
      int dieIndex=0;
      String[] bLettersArr={"AAEEGN","ELRTTY","AOOTTW","ABBJOO",
            "EHRTVN","CIMOTU","DISTTY","EIOSST","DELRVY",
            "ACHOPS","HIMNQU","EEINSU","EEGHNW",
            "AFFKPS","HLNNRZ","DEILRX"};
      boggleDice=new BoggleDie[rows][cols];
      for (int row=0; row<rows; row++) {
         for (int col=0; col<cols; col++) {
            boggleDice[row][col]=new BoggleDie(bLettersArr[dieIndex++]);
         }
      }
   }
   // Getter Methods
   public int getRows() {
      return rows;
   }
   public int getCols() {
      return cols;
   }
   public BoggleDie getBoggleDie(int row, int col) {
      if (row>=0 && row<getRows() && col>=0 && col<getCols()) {
         return boggleDice[row][col];
      }
      return null;
   }
   public String toString() {
      String s="";
      for (int r=0; r<getRows(); r++) {
         for (int c=0; c<getCols(); c++) {
            s+=getBoggleDie(r,c).getCurrentFace();
         }
         s+="\n";
      }
      return s;
   }
```

Continued from previous page:

```java
  // Setter Methods
  public void randomize() {
    for (int i=0; i<100; i++) {
      int r1=(int)(Math.random()*getRows());
      int c1=(int)(Math.random()*getCols());
      int r2=(int)(Math.random()*getRows());
      int c2=(int)(Math.random()*getCols());
      BoggleDie temp=boggleDice[r1][c1];
      boggleDice[r1][c1]=boggleDice[r2][c2];
      boggleDice[r2][c2]=temp;
      boggleDice[r1][c1].randomize();
      boggleDice[r2][c2].randomize();
    }
  }
}
```

The BoggleRunner class (Day 5, Part III) – Using the BoggleBoard class

```java
public class BoggleBoardRunner {
  public static void main(String[] args) {
    BoggleBoard bb=new BoggleBoard();
    for (int turns=1; turns<=3; turns++) {
      bb.randomize();
      System.out.println("Turn #"+turns);
      System.out.println(bb);
    }
  }
}
```