Using the Coin Class

The Coin class in the text is in the file *Coin.java*. Copy it to your directory, then write a program to find the length of the longest run of heads in 100 flips of the coin. A skeleton of the program is in the file *Runs.java*. To use the Coin class you need to do the following in the program:

* Include the Coin class in your Java/BlueJ project.
* Create a coin object.
* Inside the loop, you should use the *flip* method to flip the coin, the *toString* method (used implicitly) to print the results of the flip, and the *isHeads* method to see if the result was HEADS. Keeping track of the current run length (the number of times in a row that the coin was HEADS) and the maximum run length is an exercise in loop techniques!
* Print the result after the loop (enable BlueJ's unlimited buffering to show the entire run). Also print the Runs class code (this is the one you changed).

```
//********************************************************************
//  Coin.java       Author: Lewis/Loftus/Cocking
//  Represents a coin with two sides that can be flipped.
//********************************************************************

public class Coin
{
   private final int HEADS = 0;
   private final int TAILS = 1;

   private int face;

   //-----------------------------------------------------------------
   //  Sets up the coin by flipping it initially.
   //-----------------------------------------------------------------
   public Coin ()
   {
      flip();
   }

   //-----------------------------------------------------------------
   //  Flips the coin by randomly choosing a face value.
   //-----------------------------------------------------------------
   public void flip ()
   {
      face = (int) (Math.random() * 2);
   }

   //-----------------------------------------------------------------
   //  Returns true if the current face of the coin is heads.
   //-----------------------------------------------------------------
   public boolean isHeads ()
   {
      return (face == HEADS);
   }

   //-----------------------------------------------------------------
   //  Returns the current face of the coin as a string.
   //-----------------------------------------------------------------
   public String toString()
   {
      String faceName;
      if (face == HEADS)
         faceName = "Heads";
      else
         faceName = "Tails";
```

```
      return faceName;
   }
}

//********************************************************************
// Runs.java
//
// Finds the length of the longest run of heads in 100 flips of a coin
//********************************************************************

public class Runs
{
      public static void main (String[] args)
      {
            final int FLIPS = 100;          // number of coin flips
            int currentRun = 0;             // length of the current run of HEADS
            int maxRun = 0;                 // length of the maximum run so far

            //  Create a coin object

            //  Flip the coin FLIPS times
            for (int i = 0; i < FLIPS; i++)
            {
                  //  Flip the coin & print the results

                  //  Update the run information

            }

            //  Print the results
      }
}
```

Biased Coin - Modifying the Coin Class

1. Create a new class named *BiasedCoin* that models a biased coin (heads and tails are not equally likely outcomes of a flip). Create the new BiasedCoin class similar to the Coin class as follows:
   * Add a private data member *bias* of type double. This data member will be a number between 0 and 1 (inclusive) that represents the probability the coin will be HEADS when flipped. So, if bias is 0.5, the coin is an ordinary fair coin. If bias is 0.6, the coin has probability 0.6 of coming up heads (on average, it comes up heads 60% of the time).
   * Modify the default constructor by assigning the value 0.5 to bias *before* the call to flip. This will make the default coin a fair one.
   * Modify *flip* so that it generates a random number then assigns *face* a value of HEADS if the number is less than the bias; otherwise it assigns a value of TAILS.
   * Add a second constructor with a single double parameter -- that parameter will be the bias. If the parameter is valid (a number between 0 and 1 inclusive) the constructor should assign the *bias* data member the value of the parameter; otherwise it should assign *bias* a value of 0.5. Call flip (as the other constructor does) to initialize the value of face.
2. Create a BiasedRuns class similar to the Runs program from the last exercise that uses a BiasedCoin.
3. Print the results after the loop showing the bias and all of the heads and tails (enable BlueJ's unlimited buffering to show the entire run). Also print the code for both classes.