```
//********************************************************************
//  Represents one die (singular of dice) with faces showing

//  values between 1 and 6.
//********************************************************************

public class Die
{
   private final int MAX = 6;  // maximum face value

   private int faceValue;  // current value showing on the die

   //-----------------------------------------------------------------
   //  Constructor: Sets the initial face value of this die.
   //-----------------------------------------------------------------
   public Die()
   {
      faceValue = 1;
   }

   //-----------------------------------------------------------------
   //  Computes a new face value for this die and returns the result.
   //-----------------------------------------------------------------
   public int roll()
   {
     faceValue = (int)(Math.random() * MAX) + 1;

      return faceValue;
   }

   //-----------------------------------------------------------------
   //  Face value mutator. The face value is not modified if the
   //  specified value is not valid.
   //-----------------------------------------------------------------
   public void setFaceValue (int value)
   {
      if (value > 0 && value <= MAX)
         faceValue = value;
   }

   //-----------------------------------------------------------------
   //  Face value accessor.
   //-----------------------------------------------------------------
   public int getFaceValue()
   {
      return faceValue;
   }

   //-----------------------------------------------------------------
   //  Returns a string representation of this die.
```

```
   //-----------------------------------------------------------------
   public String toString()
  {
      String result = Integer.toString(faceValue);

      return result;
  }
}


//********************************************************************
//  Demonstrates the use of a programmer-defined class.
//********************************************************************

public class SnakeEyes
{
   //-----------------------------------------------------------------
   //  Creates two Die objects and rolls them several times, counting
   //  the number of snake eyes that occur.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      final int ROLLS = 500;
      int num1, num2, count = 0;

      Die die1 = new Die();
      Die die2 = new Die();

      for (int roll=1; roll <= ROLLS; roll++)
      {
         num1 = die1.roll();
         num2 = die2.roll();

         if (num1 == 1 && num2 == 1)     // check for snake eyes
            count++;
      }

      System.out.println ("Number of rolls: " + ROLLS);
      System.out.println ("Number of snake eyes: " + count);
      System.out.println ("Ratio: " + (float)count / ROLLS);
   }
}
```

```
//********************************************************************
//  Represents a bank account with basic services such as // // //  deposit
and withdraw.
//********************************************************************

import java.text.NumberFormat;

public class Account
{
   private final double RATE = 0.035;  // interest rate of 3.5%

   private String name;
   private long acctNumber;
   private double balance;

   //-----------------------------------------------------------------
   //  Sets up this account with the specified owner, account number,
   //  and initial balance.
   //-----------------------------------------------------------------
   public Account (String owner, long account, double initial)
   {
      name = owner;
      acctNumber = account;
      balance = initial;
   }

   //-----------------------------------------------------------------
   //  Deposits the specified amount into this account and returns
   //  the new balance. The balance is not modified if the deposit
   //  amount is invalid.
   //-----------------------------------------------------------------
   public double deposit (double amount)
   {
      if (amount > 0)
         balance = balance + amount;

      return balance;
  }

   //-----------------------------------------------------------------
   //  Withdraws the specified amount and fee from this account and
   //  returns the new balance. The balance is not modified if the
   //  withdraw amount is invalid or the balance is insufficient.
   //-----------------------------------------------------------------
   public double withdraw (double amount, double fee)
   {
      if (amount+fee > 0 && amount+fee < balance)
         balance = balance - amount - fee;
```

```
      return balance;
   }

   //-----------------------------------------------------------------
   //  Adds interest to this account and returns the new balance.
   //-----------------------------------------------------------------
   public double addInterest ()
   {
      balance += (balance * RATE);
      return balance;
   }

   //-----------------------------------------------------------------
   //  Returns the current balance of this account.
   //-----------------------------------------------------------------
   public double getBalance ()
   {
      return balance;
   }

   //-----------------------------------------------------------------
   //  Returns a one-line description of this account as a string.
   //-----------------------------------------------------------------
   public String toString ()
   {
      NumberFormat fmt = NumberFormat.getCurrencyInstance();

      return (acctNumber + "\t" + name + "\t" + fmt.format(balance));
   }
}
```

```
//********************************************************************
//  Demonstrates the creation and use of multiple Account

//  objects.
//********************************************************************

public class Transactions
{
   //-----------------------------------------------------------------
   //  Creates some bank accounts and requests various services.
   //-----------------------------------------------------------------
   public static void main (String[] args)
```

```
   {
      Account acct1 = new Account ("Ted Murphy", 72354, 25.59);
      Account acct2 = new Account ("Angelica Adams", 69713, 500.00);
      Account acct3 = new Account ("Edward Demsey", 93757, 769.32);

      acct1.deposit (44.10);  // return value ignored

      double adamsBalance = acct2.deposit (75.25);
      System.out.println ("Adams balance after deposit: " +
                          adamsBalance);

      System.out.println ("Adams balance after withdrawal: " +
                          acct2.withdraw (480, 1.50));

      acct3.withdraw (-100.00, 1.50);  // invalid transaction

      acct1.addInterest();
      acct2.addInterest();
      acct3.addInterest();

      System.out.println ();
      System.out.println (acct1);
      System.out.println (acct2);
      System.out.println (acct3);
   }
}
```