Exercise 10 Answers

Exceptions

1. Here is the `Student.java` source file after all modifications have been made:

```
/* Need to import java.io package to use the BufferedReader and
 InputStreamReader.
*/
import java.io.*;

class InvalidAgeException extends Exception {
   public InvalidAgeException () {
      super();
   }
   public InvalidAgeException (String m) {
      super(m);
   }
}

public class Student {

   private static BufferedReader stdIn =
      new BufferedReader(new InputStreamReader(System.in));

   private String name;
   private int age;

   public Student () {
      name = "";
      age = 0;
   }

   public void readName () throws IOException {
      System.out.print("Input your name: ");
      name = stdIn.readLine();
   }

   public void printName () {
      System.out.println("Name: " + name);
   }

   public void readAge () {
      boolean ok = false;

      while (!ok) {
         System.out.print("Input your age: ");
         try {
            age = Integer.parseInt(stdIn.readLine());
            if (!(ok = ((age >= 0) && (age <=150))))
               System.out.println("Try again! (range 0 to 150.)");
         }
         catch (IOException e) {
            System.out.println("Something BAD happened!");
            System.exit(0);
         }
         catch (NumberFormatException e) {
            age = -1;
            System.out.println("Try again!");
         }
      }
   }

   public void printAge () {
      System.out.println("Age: " + age);
   }


   public static void main (String[] args) throws IOException {
      Student me = new Student();
      me.readName();
      me.readAge();

      me.printName();
      me.printAge();

   }
}
```

2. Here is the `BRTest.java` source file after all modifications have been made:

```
public class BRTest {
   private final static int ARITHMETICEXCEPTION = 0;
   private final static int NULLPOINTEREXCEPTION = 1;
   private final static int ARRAYINDEXOUTOFBOUNDSEXCEPTION = 2;
   private final static int CLASSCASTEXCEPTION = 3;
   private final static int NEGATIVEARRAYSIZEEXCEPTION = 4;

   private int[] excepCounts = new int[5];
   private int totalReturned = 0;
   private int calls = 0;
   private int successfulCalls = 0;

   public void resetCounts () {
      calls = 0;
      totalReturned = 0;
      successfulCalls = 0;
      for (int i = 0; i <= NEGATIVEARRAYSIZEEXCEPTION; i++)
```

```
         excepCounts[i] = 0;
   }

   public void callIt () {
      calls++;
      try {
         totalReturned += BadRandom.randVal();
         successfulCalls++;
      }
      catch (ArithmeticException e) {
         System.out.println(e.getMessage());
         excepCounts[ARITHMETICEXCEPTION]++;
      }
      catch (NullPointerException e) {
         System.out.println(e.getMessage());
         excepCounts[NULLPOINTEREXCEPTION]++;
      }
      catch (ArrayIndexOutOfBoundsException e) {
         System.out.println(e.getMessage());
         excepCounts[ARRAYINDEXOUTOFBOUNDSEXCEPTION]++;
      }
      catch (ClassCastException e) {
         System.out.println(e.getMessage());
         excepCounts[CLASSCASTEXCEPTION]++;
      }
      catch (NegativeArraySizeException e) {
         System.out.println(e.getMessage());
         excepCounts[NEGATIVEARRAYSIZEEXCEPTION]++;
      }
      catch (Exception e) {
      }
   }

   public void nRandInts (int n) {
      for (;successfulCalls < n;) {
         callIt();
      }
   }

   public void writeData () {
      System.out.println("\n\n=========================================");
      System.out.println("Number of calls:  " + calls);
      System.out.println("Successful calls:  " + successfulCalls);
      System.out.println("Total returned:  " + totalReturned);
      System.out.println("Percentage Arithmetic Exceptions:  " +
                         ((float) excepCounts[ARITHMETICEXCEPTION]/calls*100));
      System.out.println("Percentage Null Pointer Exceptions:  " +
                         ((float) excepCounts[NULLPOINTEREXCEPTION]/calls*100));
      System.out.println("Percentage Array Index Exceptions:  " +
                         ((float) excepCounts[ARRAYINDEXOUTOFBOUNDSEXCEPTION]/calls*100));
      System.out.println("Percentage Class Cast Exceptions:  " +
                         ((float) excepCounts[CLASSCASTEXCEPTION]/calls*100));
      System.out.println("Percentage Negative Array Exceptions:  " +
                         ((float) excepCounts[NEGATIVEARRAYSIZEEXCEPTION]/calls*100));
      System.out.println("Percentage of successful calls:  " +
                         ((float) successfulCalls/calls*100));
      System.out.println("==========================================\n\n");
   }


   public static void main (String[] args) {
      BRTest me = new BRTest();
      me.resetCounts();
      me.nRandInts(30);
      me.writeData();
   }
}
```