/\*\* =======================================================================

\* Class:Die Ex11.1 Pg.439 Author: Yin Linhai

\* Version:001Date:Feb 6, 2014

\*

\* A program that rolls a dice 6 times, and gets the average of the rolls

\*

\* Course:Computer Science 201Teacher:Mr Blakey

\* School:Sir Winston Churchill High School, Calgary, Alberta, Canada

\* Language: Java SE 7.0Target Operating System: Java Virtual Machine

\* System:Intel Celeron 3GHz running under Windows 7 IDE: Eclipse 4.2

\*========================================================================\*/

**Measurable interface**

**package** elevenPointOne;

/\*\*

Describes any class whose objects can be measured.

\*/

**public** **interface** Measurable {

/\*\*

Computes the measure of the object.

**@return** the measure

\*/

**double** getMeasure();

}

**Die Class**

**package** elevenPointOne;

**import** java.util.Random;

/\*\*

This class models a die that, when cast, lands on a random

face.

\*/

**public** **class** Die **implements** Measurable {

/\*\*

Constructs a die with a given number of sides.

**@param** s the number of sides, e.g. 6 for a normal die

\*/

**public** Die(**int** s) {

sides = s;

generator = **new** Random();

}

/\*\*

Simulates a throw of the die

**@return** the face of the die

\*/

**public** **void** cast()

{

*rolled* = (**double**) (1 + generator.nextInt(sides));

}

**public** **double** getMeasure() {

cast();

**return** *rolled*;

}

**private** Random generator;

**private** **int** sides;

**private** **static** **double** *rolled*;

}

**Dataset Class**

**package** elevenPointOne;

**public** **class** DataSet {

/\*\*

Constructs an empty data set.

\*/

**public** DataSet()

{

sum = 0;

count = 0;

maximum = **null**;

}

/\*\*

Adds a data value to the data set.

**@param** x a data value

\*/

**public** **void** add(Measurable x)

{

sum = sum + x.getMeasure();

**if** (count == 0 || maximum.getMeasure() < x.getMeasure())

maximum = x;

count++;

}

/\*\*

Gets the average of the added data.

**@return** the average or 0 if no data has been added

\*/

**public** **double** getAverage()

{

**if** (count == 0) **return** 0;

**else** **return** sum / count;

}

/\*\*

Gets the largest of the added data.

**@return** the maximum or 0 if no data has been added

\*/

**public** Measurable getMaximum()

{

**return** maximum;

}

**private** **double** sum;

**private** Measurable maximum;

**private** **int** count;

}

**Tester Class**

**package** elevenPointOne;

**public** **class** Tester {

**public** **static** **void** main(String[] args) {

DataSet data = **new** DataSet();

Die die = **new** Die(6);

**for**(**int** x=0; x<6; x++) {

data.add(die);

}

System.*out*.println(data.getAverage());

}

}

**Output:**

3.1666666666666665