Homework 8

**Name: Peter Dobbs**

**MU ID: 005803169**

**Date: 26 March, 2015**

**Homework Problem 1**

**Requirements**

Build a program that estimates pi using the Buffon’s needle technique.

**Design**

*main*

Create a random number generator for the landing site of the needle and the angle at which the needle lands, with respect to the horizontal. Place these random number generators in a ‘for loop’ and count how many times the end of the needle dropped crosses the boundary. Divide the total number of violations of the boundary into the number of times the ‘for loop’ went.

**Iterative Development Steps**

1. Import the random java utility into the program and create a new class.
2. Create a main method and initiate the Random class. Use the Random class to make new random doubles for the variables you desire.
3. Use a ‘for loop’ to find the ratio of needle drops to violations of the boundary, which should be pi

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | N/A | Pi | Pi | Good code |

**Homework Problem 2**

**Requirements**

Write a program that finds the distance between two points.

**Design**

class Point

instance variables

constructor(s)

*distanceFromOrigin*()

Uses the distance equation to find the distance from (0,0) to a given point (x,y).

*toString*()

Puts the x and y variables into String format.

*distance*(Point r)

Uses the distance equation to find the distance from (this.x, this.y) to (r.x, r.y).

class TestPoint

*main*

Creates variables and calls methods from Point class with variables as parameters.

**Iterative Development Steps**

1. Create class Point with constructors and methods distanceFromOrigin(), toString(), and distance(Point r). Distance methods calculate the distance between points and toString returns the String form of the input parameters.
2. Create class TestPoint with parameters either hard-coded or input through the command line. Call the methods desired for displaying the desired calculation and result.

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | (0, 2)  (4, 5) | (0, 2) is 5.0 far from (4, 5) | (0, 2) is 5.0 far from (4, 5) | Good Code |

**Homework Problem 3**

**Requirements**

Create a program that adds and multiplies rational numbers.

**Design**

class RationalNumber

instance variables

constructor

*add*(RationalNumber r)

adds the rational numbers it is passed and returns a new RationalNumber object

*multiply*(RationalNumber r)

multiplies the rational numbers it is passed and returns a new RationalNumber object

*reduce*()

reduces the rational number into least common terms

*toString*()

returns the passed parameters as String format.

class TestRationalNumber

*main*

Creates rational number objects and passes them to add() and multiply()

**Iterative Development Steps**

1. Create class RationalNumber with methods add(), multiply(), reduce(), and toString(). These methods should allow for the operations expected of them to occur.
2. Create class TestRationalNumber. In the main method of this class, test the RationalNumber class using input numbers as numerators and denominators.

**Tests**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description of test | Input | Expected result | Actual result | Cause |
| method1 normal input | 3  7  7  3 | Sum = 58/21  Product = 1 | Sum = 58/21  Product = 1 | Good Code |

**References**

An Introduction to Programming Using Java, Anthony J Dos Reis.