СПб НИУ ИТМО

кафедра ИПМ

Основы программной инженерии

Лабораторная работа № 3

Вариант 2041

Работу выполнил:

Студент II курса

Группы № 2120

Журавлев Виталий

Санкт-Петербург

2013 г.

**Цель работы:**

С помощью пакета JUnit провести модульное тестирование программы, реализующей задание к лабораторной работе #3 по дисциплине "Программирование интернет-приложений". Тестировое покрытие должно быть реализовано для всех классов приложения, тестовые сценарии должны учитывать все возможные варианты попадания или непопадания точек в область на координатной плоскости, а также возможность ввода пользователем некорректных данных.

**Требования к тестируемой программе:**

На языке Java написать консольную программу, которая определяет, какие точки из набора A входят в заданную область S.

Приложение должно содержать следующие классы:

* Класс Mark, представляющий точку с координатами X и Y типа float.
* Класс Silhouette, представляющий область с заданным параметром R, в котором должен быть реализован метод, возвращающий для заданной точки значение 1, если точка входит в область, и 0, если не входит. Попадание на границу области считается попаданием в область.
* Класс Lab2, который получает параметр R типа int со стандартного ввода по запросу пользователя. Получение числа из строки реализовать c помощью метода Scanner.nextInt().

Точки хранятся в виде коллекции параметризованного типа PriorityQueue.

Обход коллекции реализовать с помощью цикла for(a:b).

Приложение должно выводить на экран список точек, не попадающих в область.



**Тестовое покрытие:**

* Точки:
	+ В центр области
	+ По краям области
	+ В пустую область
	+ Чуть больше/меньше границ
	+ Случайные числа
* Радиус:
	+ Положительное число
	+ Отрицетельное число
	+ Равный нулю
	+ Большой
	+ Текстовый

**Текст тестовых программ:**

1. **Класс JavaLab3**

package javalab3;

import org.junit.Test;

import static org.junit.Assert.\*;

public class JavaLab3Test {

 Object R;

 Silhouette S;

@Test

 public void testInt()

 {

 R = 13564;

 try

 {

 S = new Silhouette((int)R);

 }

 catch (Exception e)

 {

 fail("Unhandled exception:" + e.toString());

 }

 }

 @Test

 public void testNegativeInt()

 {

 R = (int)-84651;

 try

 {

 S = new Silhouette((int)R);

 }

 catch (Exception e)

 {

 fail("Unhandled exception:" + e.toString());

 }

 }

 @Test

 public void testZero()

 {

 R = (int)0;

 try

 {

 S = new Silhouette((int)R);

 }

 catch (Exception e)

 {

 fail("Unhandled exception:" + e.toString());

 }

 }

 @Test

 public void testLong()

{

 R = (long) Long.MAX\_VALUE-1000;

 try

 {

 S = new Silhouette((int)R);

 }

 catch (Exception e)

 {

 fail("Unhandled exception:" + e.toString());

 }

 }

 @Test

 public void testString()

 {

 R = "ftkdrytkufg15648";

 try

 {

 S = new Silhouette((int)R);

 }

 catch (Exception e)

 {

 fail("Unhandled exception:" + e.toString());

 }

 }

}

1. **Класс Silhouette**

package javalab3;

import java.util.Random;

import org.junit.Test;

import static org.junit.Assert.\*;

import org.junit.BeforeClass;

public class SilhouetteTest

 {

 Mark mark1,mark2,mark3,mark4;

 static int r;

 static Random RandomR = new Random();

 static Silhouette s;

 @BeforeClass

 public static void CreateArea()

 {

 r = Math.abs(RandomR.nextInt());

 s = new Silhouette(r);

 }

 @Test

 public void TestCentreArea()

 {

 mark1 = new Mark(-r/4,r/6);

 mark2 = new Mark(-r/4,-r/3);

 mark3 = new Mark(2\*r/3,-3\*r/5);

 assertTrue(s.IsHit(mark1)==1&&s.IsHit(mark2)==1&&s.IsHit(mark3)==1);

 }

 @Test

 public void TestBorderArea()

 {

 mark1 = new Mark((float)(-r/Math.sqrt(2)),(float)(-r/Math.sqrt(2)));

 mark2 = new Mark(-r/4,r/4);

 mark3 = new Mark(r,-r);

 mark4 = new Mark(0,0);

 assertTrue(s.IsHit(mark1)==1&&s.IsHit(mark2)==1&&s.IsHit(mark3)==1&&s.IsHit(mark4)==1);

 }

 @Test

 public void TestEmptyArea()

 {

 mark1 = new Mark(4,6);

 mark2 = new Mark(r/4,r/3);

 mark3 = new Mark(12\*r,24\*r/5);

 assertTrue(s.IsHit(mark1)==0&&s.IsHit(mark2)==0&&s.IsHit(mark3)==0);

 }

 @Test

 public void TestAroundBorderArea()

 {

 mark1 = new Mark((float)(-r/Math.sqrt(2.00001)),(float)(-r/Math.sqrt(2)));

 mark2 = new Mark(-48652/194604,r/4);

 mark3 = new Mark(2578\*r/2577,-87215\*r/87216);

 mark4 = new Mark((float)Math.sin(Math.PI+0.00000001),0);

 assertTrue(s.IsHit(mark1)==1&&s.IsHit(mark2)==0&&s.IsHit(mark3)==1&&s.IsHit(mark4)==0);

 }

 @Test

 public void TestRandomMarks()

 {

 Random RandomX;

 Random RandomY;

 float x;

 float y;

 short check;

 for(int k = 0; k < 100; k++)

 {

 RandomX = new Random();

 RandomY = new Random();

 x = RandomX.nextInt()/100000;

 y = RandomY.nextInt()/100000;

 mark1 = new Mark(x,y);

 boolean Shoot = (y>=0&&x<=0&&y<=x+r/2||y<=0&&x>=0&&x<=r&&y>=-r||y<=0&&x<=0&&Math.pow(x, 2)+Math.pow(y, 2)<=Math.pow(r/2, 2));

 if (Shoot) check = 1; else check = 0;

 if (s.IsHit(mark1)!= check) fail("Error on random Marks Tests");

 }

 assertTrue(s.IsHit(mark1)== 1);

 }

 }

**Результат:**

**Вывод:**

В ходе выполнения лабораторной работы я научился работать с модульным тестированием в Java посредством пакета JUnit, а так же разобрался с написанием самих тестов и определением тестовых покрытий.