Университет ИТМО

Кафедра вычислительной техники

**Отчёт по лабораторной работе №3  
по предмету «Программирование интернет-приложений»**

Выполнил:  
Студент гр. P3201  
Петров Д. С.

Принял:  
Гаврилов А. В.

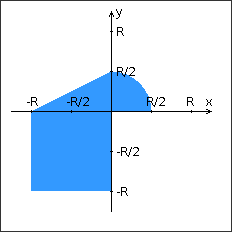
**Номер варианта**

Вариант выданный преподавателем: **20121**

**Цель работы**

Изучить:

1. Основные пакеты JDK 1.8.
2. Пакет java.lang.\*. Классы Object, Class, System.
3. Обработка исключительных ситуаций, три типа исключений.
4. Работа с массивами данных.
5. Класс String.
6. Классы InputStream, InputStreamReader.
7. Пакет java.nio. Класс Buffer. Каналы ввода-вывода.
8. Коллекции. Сортировка элементов коллекций.
9. Параметризованные типы.

**Задание**

На языке Java написать консольную программу, которая определяет, какие точки из множества {{0, 0}, {3, -5}, {3, 3}, {-2, 0}, {2, 1}, {-3, -5}, {-4, 5}, {-1, 1}, {-2, 0}} входят в заданную область S.

**Приложение должно содержать следующие классы:**

* Класс Spot, представляющий точку с координатами X и Y типа double.
* Класс Contour, представляющий область с заданным параметром R типа float, в котором должен быть реализован метод, возвращающий для заданной точки значение true, если точка входит в область, и false, если не входит. Попадание на границу области не считается попаданием в область.
* Основной класс Task3, в котором необходимо реализовать получение значения R со стандартного ввода по запросу пользователя. Преобразование из строки в число реализовать c помощью метода Float.parseFloat().

Точки хранятся в виде коллекции непараметризованного типа Vector.

Обход коллекции реализовать с помощью цикла for(element : Collection).

Приложение должно выводить на экран список всех точек с указанием попадания в область.

**Исходный код получившейся программы**

**Файл Task.java**

// Вариант 20121

import java.lang.\*;

import java.lang.Math.\*;

import java.util.\*;

public class Task3 {

public static void main(String[] args) {

Vector<Spot> points = new Vector<Spot>();

points.add(new Spot(0, 0));

points.add(new Spot(3, -5));

points.add(new Spot(3, 3));

points.add(new Spot(-2, 0));

points.add(new Spot(2, 1));

points.add(new Spot(-3, -5));

points.add(new Spot(-4, 5));

points.add(new Spot(-1, 1));

points.add(new Spot(-2, 0));

float r = getRadius();

Contour contour = new Contour(r);

for (Spot point : points) {

if (contour.isInContour(point)) {

System.out.println("Point (" + point.x + ", " + point.y + ") belongs to an open set");

}

else {

System.out.println("Point (" + point.x + ", " + point.y + ") doesn't belong to an open set");

}

}

}

private static float getRadius() {

float r;

Scanner sc = new Scanner(System.in);

System.out.println("Please, print R");

while (!sc.hasNextFloat()) {

System.out.println("Please, print real number!");

sc.nextLine();

}

return sc.nextFloat();

}

}

**Файл Contour.java**

import java.lang.\*;

import java.lang.Math.\*;

class Contour {

private float r;

public Contour(float r) {

this.r = r;

}

public boolean isInContour(Spot s) {

float x = (float) s.x;

float y = (float) s.y;

if (x < 0) {

if (y < 0) {

if(x > -r && y > -r) return true; // Square

}

if (y > 0) {

if (y < 0.5f \* (x + r)) return true; // Circle

}

if (y == 0) {

if (x > -r) return true; // Interval

}

}

if (x > 0) {

if (y > 0) {

if (x\*x + y\*y < r\*r) return true; // Triangle

}

}

if (x == 0) {

if (y > 0 && y < r/2) return true; // Interval

}

return false;

}

}

**Файл Spot.java**

import java.lang.\*;

import java.lang.Math.\*;

class Spot {

public double x, y;

public Spot(double x, double y) {

this.x = x;

this.y = y;

}

}

**Вывод**

Java NIO позволяет управлять несколькими каналами (сетевыми соединениями или файлами) используя минимальное число потоков выполнения. Однако ценой такого подхода является более сложный, чем при использовании блокирующих потоков, парсинг данных.

В языке java существуют параметризованные типы данных (т.н. generic), позволяющие писать не одноразовый код, а также стандартные методы сортировки коллекций, упрощающие жизнь программисту.