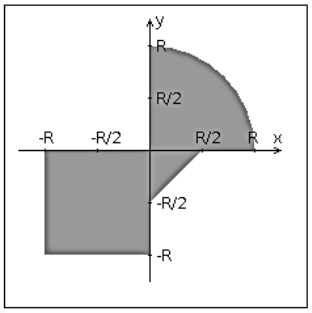
ИТМО Кафедра Вычислительной техники

Отчет по лабораторной работе №3
Программирование интернет-приложений
Вариант 17183

Выполнил: студент группы P3217
Плюхин Дмитрий
Преподаватель: Гаврилов А. В.

2016 год

1. Задание к лабораторной работе



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

Приложение должно содержать следующие классы:

- Класс Ponto, представляющий точку с координатами X и Y типа float.
- Класс Silhouette, представляющий область с заданным параметром R типа double, 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 Lab3, в котором необходимо реализовать получение значения R в качестве аргумента командной строки. Преобразование из строки в число реализовать с помощью метода Double.parseDouble().

Точки хранятся в виде коллекции непараметризованного типа HashSet.

Обход коллекции реализовать с помощью цикла for с итератором.

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

2. Исходный код

```
//Файл Ponto.java
public class Ponto {
    private float x = 0;
    private float y = 0;

    public Ponto(float x, float y){
        this.x = x;
        this.y = y;
    }

    public void setX(float x){
        this.x = x;
    }

    public float getX(){
        return x;
    }

    public void setY(float x){
        this.x = x;
    }

    public float getY(){
        return y;
    }

    public String toString(){
        return ("Ponto { "+x+" ; "+y+" }");
    }
}

//Файл Silhouette
public class Silhouette {
    private double r = 0;

    public Silhouette(double r){
        setR(r);
    }

    public void setR(double r){
        if (r <= 0){
            throw new IllegalArgumentException();
        }
        this.r = r;
    }

    public double getR(){
        return r;
    }

    public boolean checkPonto(Ponto p){
        float x = p.getX();
        float y = p.getY();

        return ((x*x + y*y < r*r) && (x > 0) && (y > 0)) || //part of circle
                (((x > -r) && (x < 0)) && ((y > -r) && (y < 0))) || //square
```

```
                ((y > x - r/2) && (x >= 0) && (y <= 0) && !((x == 0) && (y == 0)));
                                                                        //triangle
    }
}


//Файл Lab3.java
public class Lab3 {
    public static void main (String[] args){
        HashSet pontos = new HashSet();
        Silhouette slh;
        try {
            slh = new Silhouette(Double.parseDouble(args[0]));
        } catch (IndexOutOfBoundsException e){
            System.out.println("There are no arguments");
            return;
        } catch (NumberFormatException e){
            System.out.println("There is not number in argument");
            return;
        }

        pontos.add(new Ponto(-3,4));
        pontos.add(new Ponto(0,3));
        pontos.add(new Ponto(0,0));
        pontos.add(new Ponto(3,2));
        pontos.add(new Ponto(0,3));
        pontos.add(new Ponto(4,-4));
        pontos.add(new Ponto(-4,-5));
        pontos.add(new Ponto(2,-3));
        pontos.add(new Ponto(3,3));

        Ponto curPonto;

        for (Iterator i = pontos.iterator(); i.hasNext();){
            curPonto = (Ponto)i.next();
            if (!slh.checkPonto(curPonto)){
                System.out.println(curPonto);
            }
        }
    }
}
```

3. Вывод

В результате лабораторной работы было написано простое консольное приложение, с помощью которого были изучены несколько новых возможностей Java, а именно: исследованы способы работы с коллекцией непараметризованного типа HashSet, рассмотрены возможности по добавлению элементов в эту коллекцию. Был исследован способ обхода коллекции с помощью итератора и сделан вывод о том, что этот способ хотя и сложнее использования foreach, но предоставляет большие возможности по работе с коллекцией. Кроме того, были исследованы основы обработки исключений в Java и сделан вывод о том, что этот процесс мало отличается от уже известных методов работы с исключениями в C#.