**ГБОУ "Президентский ФМЛ № 239"**

**Рисование окружностей заданного радиуса, содержащих внутри себя точки**

Годовой проект по информатике

Работу выполнила

Ученица 10-3 класса

Грек Полина

Санкт-Петербург

2017

**1.Постановка задачи**

Из заданного множества точек на плоскости выбрать две различные точки так, чтобы окружности заданного радиуса с центрами в этих точках содержали внутри себя одинаковое количество заданных точек.

Необходимо найти подходящие по условию точки из заданного множества точек.

**2.Уточнение исходных и выходных данных**

Во входном файле содержатся диаметры окружностей и координаты точек. Для окружности ограничение 700.

**3.Анализ используемой структуры данных**

В задаче требуется хранить координаты заданных точек и диаметры окружностей. Координаты точек хранятся в массиве Dot[] arr в классе Set. Диаметры хранятся в переменных R1 и R2. Также нужно временно хранить количество точек, которые находятся на нужном расстоянии от выбранных, чтобы сравнивать их количество, для этого используем переменные cnt1 и cnt2.

**4.Описание метода решения**

Мы выбираем две любые точки из предложенных и проверяем, подходят ли они по условию, то есть, действительно ли внутри окружностей, центрами которых являются эти точки одинаковое количество точек. Если да, то сохраняем координаты точек внутри. Если нет, то выбираем другие.

**5.Составление алгоритма**

Нет

X1=a Y1=b X2=c Y2=d

Cnt1=Cnt2

Да

**6.Листинг программы**

**import** javax.swing.\*;  
**import** java.awt.\*;  
**import** java.awt.event.ActionEvent;  
**import** java.awt.event.ActionListener;  
**import** java.awt.event.MouseAdapter;  
**import** java.awt.event.MouseEvent;  
**import** java.awt.geom.Arc2D;  
**import** java.io.File;  
**import** java.io.FileNotFoundException;  
**import** java.io.PrintStream;  
**import** java.util.ArrayList;  
**import** java.util.Scanner;  
**import** java.util.StringTokenizer;  
  
**public class** Main {  
  
 **private static** ArrayList<Point> *points* = **new** ArrayList<Point>();  
 **private static** ArrayList<Circle> *circles* = **new** ArrayList<Circle>();  
 **public static void** createGUI() {  
 **final** JFrame frame = **new** JFrame(**"Testframe"**);  
 frame.setPreferredSize(**new** Dimension(700,700));  
 JPanel panel = **new** JPanel(**new** BorderLayout());  
 Panel butPanel = **new** Panel();  
 butPanel.setLayout(**null**);  
 butPanel.setPreferredSize(**new** Dimension(250,700));  
 **final** Panel pointpane = **new** Panel();  
 pointpane.setLayout(**null**);  
 *//pointpane.setPreferredSize(new Dimension(350,700));* JLabel addPointwithCoords = **new** JLabel(**"Добавить точку по координатам"**);  
 addPointwithCoords.setBounds(2,2,300,25);  
 butPanel.add(addPointwithCoords);  
 JLabel addRandomPoints = **new** JLabel(**"Добавить рандомное количество точек"**);  
 addRandomPoints.setBounds(2,50,300,25);  
 butPanel.add(addRandomPoints);  
 JLabel X = **new** JLabel(**"X:"**);  
 X.setBounds(2,25,15,25);  
 butPanel.add(X);  
 JLabel Y = **new** JLabel(**"Y:"**);  
 Y.setBounds(45,25,15,25);  
 butPanel.add(Y);  
 JLabel N = **new** JLabel(**"NUM:"**);  
 N.setBounds(2,70,30,25);  
 butPanel.add(N);  
 **final** JTextField x = **new** JTextField();  
 x.setBounds(17,25, 25,25);  
 butPanel.add(x);  
 **final** JTextField y = **new** JTextField();  
 y.setBounds(60,25, 25,25);  
 butPanel.add(y);  
 **final** JTextField n = **new** JTextField();  
 n.setBounds(35,70,25,25);  
 butPanel.add(n);  
  
  
  
 JLabel R1 = **new** JLabel(**"R1:"**);  
 R1.setBounds(50,350,15,25);  
 butPanel.add(R1);  
  
 **final** JTextField r1 = **new** JTextField();  
 r1.setBounds(75,350, 25,25);  
 butPanel.add(r1);  
  
 JLabel R2 = **new** JLabel(**"R2:"**);  
 R2.setBounds(50,400,15,25);  
 butPanel.add(R2);  
  
 **final** JTextField r2 = **new** JTextField();  
 r2.setBounds(75,400, 25,25);  
 butPanel.add(r2);  
  
  
 **final** JButton button1 = **new** JButton(**"Добавить точку"**);  
 button1.setBounds(2,100,160,40);  
 butPanel.add(button1);  
 button1.addActionListener(**new** ActionListener() {  
  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
  
 **int** X = (!x.getText().equals(**""**) ? Integer.*parseInt*(x.getText()) : 0);  
 **int** Y = (!y.getText().equals(**""**) ? Integer.*parseInt*(y.getText()) : 0);  
 **int** N = (!n.getText().equals(**""**) ? Integer.*parseInt*(n.getText()) : 0);  
 **if** ((X > 0) && (Y > 0)) {  
 Point b = **new** Point(X, Y);  
 *points*.add(b);  
 b.setBounds(b.**x**, b.**y**, b.**x** + 3, b.**y** + 3);  
 pointpane.add(b);  
 pointpane.revalidate();  
 pointpane.repaint();  
  
  
 } **else** {  
 **if** (N > 0) {  
 **for** (**int** i = 0; i < N; i++) {  
 Point b = **new** Point((**int**) (Math.*random*() \* (frame.getWidth() - 250)), (**int**) (Math.*random*() \* frame.getHeight()));  
 *points*.add(b);  
 b.setBounds(b.**x**, b.**y**, b.**x** + 3, b.**y** + 3);  
 pointpane.add(b);  
 pointpane.revalidate();  
 pointpane.repaint();  
 }  
 }  
 }  
 }  
 }  
  
 );  
  
  
  
 JButton button2 = **new** JButton(**"очистить"**);  
 button2.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
 **for** (**int** i=0;i<*points*.size();i++){  
 **while**(*points*.size() > 0) {  
 **int** index = *points*.size() - 1;  
 Point point = *points*.remove(index);  
 pointpane.remove(point);  
 pointpane.repaint();  
 pointpane.revalidate();  
 }  
 }  
  
 **for** (**int** i=0;i<*circles*.size();i++){  
 **while**(*circles*.size() > 0) {  
 **int** index = *circles*.size() - 1;  
 Circle circle = *circles*.remove(index);  
 pointpane.remove(circle);  
 pointpane.repaint();  
 pointpane.revalidate();  
 }  
 }  
 }  
 });  
 button2.setBounds(2,150,160,40);  
 butPanel.add(button2);  
 panel.add(pointpane,BorderLayout.***CENTER***);  
 panel.add(butPanel,BorderLayout.***EAST***);  
 frame.getContentPane().add(panel);  
 frame.pack();  
 frame.setLocationRelativeTo(**null**);  
 frame.setVisible(**true**);  
  
  
 **final** JButton button3 = **new** JButton(**"Добавить окружность"**);  
 butPanel.add(button3);  
 button3.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
  
 **int** R1 = (!r1.getText().equals(**""**) ? Integer.*parseInt*(r1.getText()) : 0);  
 **int** R2 = (!r2.getText().equals(**""**) ? Integer.*parseInt*(r2.getText()) : 0);  
  
 **int**[] cnt1 = **new int**[*points*.size()];  
 **int**[] cnt2 = **new int**[*points*.size()];  
 **for** (**int** i = 0; i < *points*.size(); i++) {  
 **for** (**int** j = i + 1; j < *points*.size(); j++) {  
  
 *// Если i-й точки до j-й точки расстояние меньше первого радиуса, то j-я точка входит в  
 // окружность первого радиуса вокруг первой точки.* **double** dist = Math.*sqrt*( Math.*pow*(*points*.get(i).**x** - *points*.get(j).**x**, 2) +  
 Math.*pow*(*points*.get(i).**y** - *points*.get(j).**y**, 2));  
 **if** (dist < R1 / 2) {  
  
 cnt1[i]++;  
 cnt1[j]++;  
 }  
  
 **if** (dist < R2 / 2) {  
  
 cnt2[i]++;  
 cnt2[j]++;  
 }  
 }  
 }  
  
 **for** (**int** i = 0; i < *points*.size(); i++) {  
 **for** (**int** j = i + 1; j < *points*.size(); j++) {  
 **if** (cnt1[i] == cnt2[j]) {  
  
 Circle b = **new** Circle(*points*.get(i).**x** + R1 / 2, *points*.get(i).**y** + R1 / 2, R1);  
 *circles*.add(b);  
 b.setBounds(1, 1, 700, 700);  
 pointpane.add(b);  
  
  
 Circle c = **new** Circle(*points*.get(j).**x** + R2 / 2, *points*.get(j).**y** + R2 / 2, R2);  
 *circles*.add(c);  
 c.setBounds(1, 1, 700, 700);  
 pointpane.add(c);  
  
 pointpane.revalidate();  
 pointpane.repaint();  
  
 **return**;  
 }  
 }  
 }  
 }  
 });  
 button3.setBounds(2,200,160,40);  
  
 JButton button4 = **new** JButton(**"Добавить в файл"**);  
 butPanel.add(button4);  
 button4.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
  
 **int** R1 = (!r1.getText().equals(**""**) ? Integer.*parseInt*(r1.getText()) : 0);  
 **int** R2 = (!r2.getText().equals(**""**) ? Integer.*parseInt*(r2.getText()) : 0);  
  
 **int**[] cnt1 = **new int**[*points*.size()];  
 **int**[] cnt2 = **new int**[*points*.size()];  
 **for** (**int** i = 0; i < *points*.size(); i++) {  
 **for** (**int** j = i + 1; j < *points*.size(); j++) {  
  
 *// Если i-й точки до j-й точки расстояние меньше первого радиуса, то j-я точка входит в  
 // окружность первого радиуса вокруг первой точки.* **if** (Math.*sqrt*( Math.*pow*(*points*.get(i).**x** - *points*.get(j).**x**, 2) +  
 Math.*pow*(*points*.get(i).**y** - *points*.get(j).**y**, 2)) < R1 / 2) {  
  
 cnt1[i]++;  
 cnt1[j]++;  
 }  
  
 **if** (Math.*sqrt*( Math.*pow*(*points*.get(i).**x** - *points*.get(j).**x**, 2) +  
 Math.*pow*(*points*.get(i).**y** - *points*.get(j).**y**, 2)) < R2 / 2) {  
  
 cnt2[i]++;  
 cnt2[j]++;  
 }  
 }  
 }  
  
 **for** (**int** i = 0; i < *points*.size(); i++) {  
 **for** (**int** j = i + 1; j < *points*.size(); j++) {  
 **if** (cnt1[i] == cnt2[j]) {  
  
 **try** {  
 PrintStream fout = **new** PrintStream(**new** File(**"out.txt"**));  
 fout.println(**"R1 = "** + R1);  
  
 **for** (**int** ii = 0; ii < *points*.size(); ii++) {  
 **if** (i == ii) {  
 **continue**;  
 }  
 *// Если i-й точки до j-й точки расстояние меньше первого радиуса, то j-я точка входит в  
 // окружность первого радиуса вокруг первой точки.* **if** (Math.*sqrt*( Math.*pow*(*points*.get(i).**x** - *points*.get(ii).**x**, 2) +  
 Math.*pow*(*points*.get(i).**y** - *points*.get(ii).**y**, 2)) < R1) {  
  
  
  
 fout.println(**""** + *points*.get(ii).**x** + **" "** + *points*.get(ii).**y**);  
 **break**;  
 }  
 }  
  
 fout.println(**"R2 = "** + R2);  
  
 **for** (**int** ii = 0; ii < *points*.size(); ii++) {  
 **if** (j == ii) {  
 **continue**;  
 }  
 *// Если i-й точки до j-й точки расстояние меньше первого радиуса, то j-я точка входит в  
 // окружность первого радиуса вокруг первой точки.* **if** (Math.*sqrt*( Math.*pow*(*points*.get(j).**x** - *points*.get(ii).**x**, 2) +  
 Math.*pow*(*points*.get(j).**y** - *points*.get(ii).**y**, 2)) < R2) {  
  
  
 fout.println(**""** + *points*.get(ii).**x** + **" "** + *points*.get(ii).**y**);  
 **break**;  
 }  
 }  
  
 }  
 **catch** (Exception ex) {  
  
 }  
  
 **return**;  
 }  
 }  
 }  
 }  
 });  
 button4.setBounds(2,250,160,40);  
  
 JButton button5 = **new** JButton(**"Прочитать из файла"**);  
 butPanel.add(button5);  
 button5.addActionListener(**new** ActionListener() {  
 @Override  
 **public void** actionPerformed(ActionEvent e) {  
  
 Scanner fin = **null**;  
 PrintStream fout = **null**;  
  
 **try** {  
  
 fin = **new** Scanner(**new** File(**"In.txt"**));  
  
 **if** (fin.hasNextInt()) {  
 r1.setText(**""** + fin.nextInt());  
 }  
  
 **if** (fin.hasNextInt()) {  
 r2.setText(**""** + fin.nextInt());  
 }  
  
 **while** (fin.hasNextInt()) {  
 **int** x = fin.nextInt();  
 **int** y = 0;  
  
 **if** (fin.hasNextInt()) {  
 y = fin.nextInt();  
 }  
 **else** {  
 *// Информация об ошибке. Количество координат нечётное* **return**;  
 }  
  
 Point b = **new** Point(x, y);  
 *points*.add(b);  
 b.setBounds(b.**x**, b.**y**, b.**x** + 3, b.**y** + 3);  
 pointpane.add(b);  
 pointpane.revalidate();  
 pointpane.repaint();  
 }  
 }  
 **catch** (FileNotFoundException ex) {  
 *// Сообщение об ошибке - нет файла* }  
 }  
 });  
 button5.setBounds(2,300,160,40);  
  
  
 }  
  
  
 **public static void** main(String[] args) {  
 javax.swing.SwingUtilities.*invokeLater*(**new** Runnable() {  
 **public void** run() {  
 JFrame.*setDefaultLookAndFeelDecorated*(**true**);  
 *createGUI*();  
 }  
 });  
 }  
}

7.Пример работы программы

Исходные данные:

80 //R1

100 //R2

100 150 //Координаты точек

200 200

67 76

564 876

500 300

23 345

123 123

89 456

320 100

200 500

Выходящие данные:

R1 = 80

123 123

R2 = 100

100 150