Государственное бюджетное общеобразовательное учреждение "Президентский физико-математической лицей № 239"

**Максимальная площадь пересечения широкого луча и окружности**

Годовой проект по информатике

**Работу выполнил**

**Ученик 10-7 класса**

**Нигматулин Даниил**

Cанкт-Петербург

2021

**1.Постановка задачи**

На плоскости задано множество "широких лучей" и множество окружностей. Найти такую пару "широкий луч"-окружность, что фигура, находящаяся внутри "широкого луча" и окружности, имеет максимальную площадь. В качестве ответа: выделить найденные "широкий луч" и окружность, выделить контур фигуры, которая ограничивает точки внутри найденного "широкого луча" и окружности, желательно выделить внутреннее пространство фигуры ("залить цветом").

Есть некое количество широких лучей и окружностей, которые как-то пересекаются. Нужно выделить пару, имеющую максимальную площадь пересечения.

**2. Уточнение исходных и выходных данных и ограничений на них**

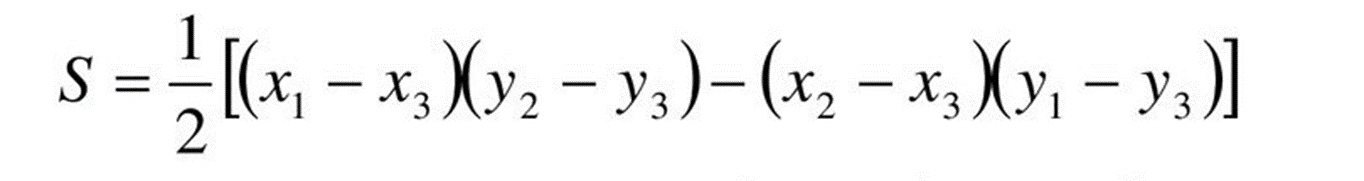
* **2.1. Входные данные:** количество случайных окружностей и лучей;
* **2.2. Выходные данные*:*** выделенная нужная пара «широкий луч – окружность»;

**3. Выбор метода решения**

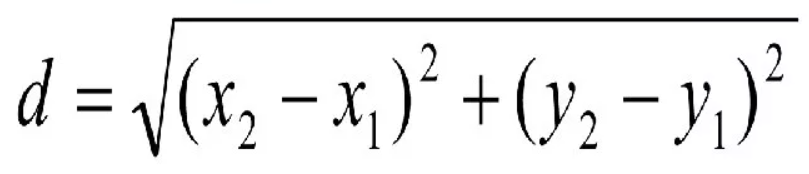
Для решения задачи нужно знать значение площадей пересечения всех окружностей и широких лучей. Не получалось решить задачу стандартными методами, так как требовался разбор большого количества случаев. Удобно использовать метод Монте Карло. Создаем большое число случайных точек, а затем выясняем, содержится ли данная точка в данном луче или данной окружности. Пара, имеющая максимальное число точек- искомая пара, которую нужно выделить.

**4. Математическая модель**

1. Площадь треугольника по координатам (нужна для проверки содержания случайной точки в широком луче). (Xi ; Yi)-координаты вершин треугольника.



1. Расстояние между двумя точками (нужна для проверки содержания случайной точки в окружности)



**5. Анализ используемой структуры данных**

Есть массив, хранящий случайные точки – points, хранящий случайные окружности – circles и хранящий случайные лучи – rays.

**6. Комментированный листинг**

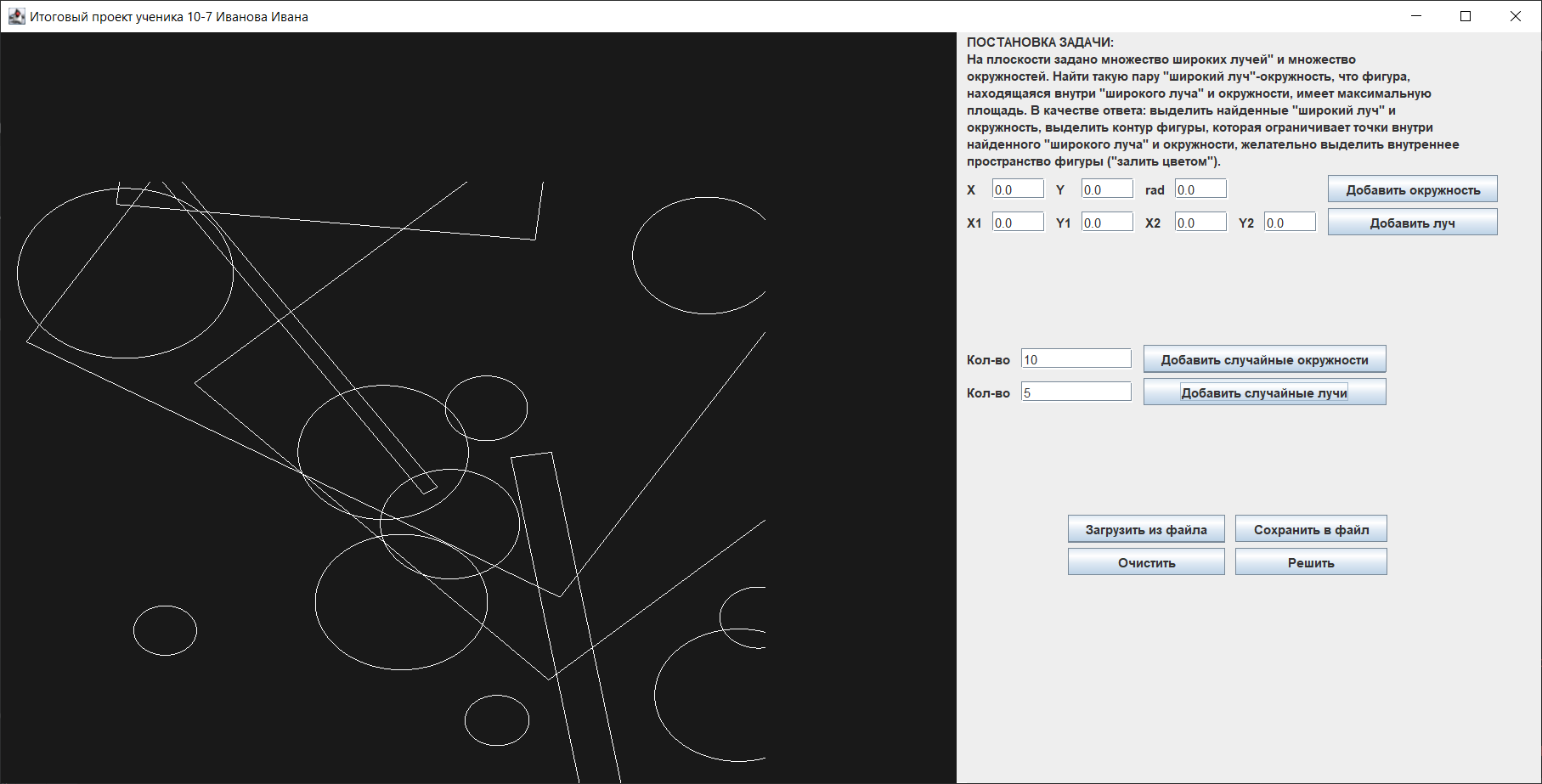
1. public boolean contains(Vector2 pos) {  
    return pos.minus(this.pos).len() <= r;  
   } //метод, проверяющий содержание случайной точки в окружности
2. public boolean contains(Vector2 pos) { //метод, проверяющий содержание случайной точки внутри широкого луча  
    double k = B.minus(A).len(); //длина одной из сторон широкого луча  
    double l = B.minus(C).len(); // длина другой стороны широкого луча  
    double S1 = l \* k; //площадь широкого луча  
     
    double sa = ((pos.x - B.x) \* (A.y - B.y) - (A.x - B.x) \* (pos.y - B.y)) / 2;  
    double sb = ((pos.x - B.x) \* (C.y - B.y) - (C.x - B.x) \* (pos.y - B.y)) / 2;  
    double sc = ((pos.x - D.x) \* (C.y - D.y) - (C.x - D.x) \* (pos.y - D.y)) / 2;  
    double sd = ((pos.x - D.x) \* (A.y - D.y) - (A.x - D.x) \* (pos.y - D.y)) / 2;  
     
    double S2 = Math.*abs*(sa) + Math.*abs*(sb) + Math.*abs*(sc) + Math.*abs*(sd);

// сумма площадей треугольников с вершинами в центре луча и в центре окружности  
 return Math.*abs*(S1 - S2) < 0.001;  
}

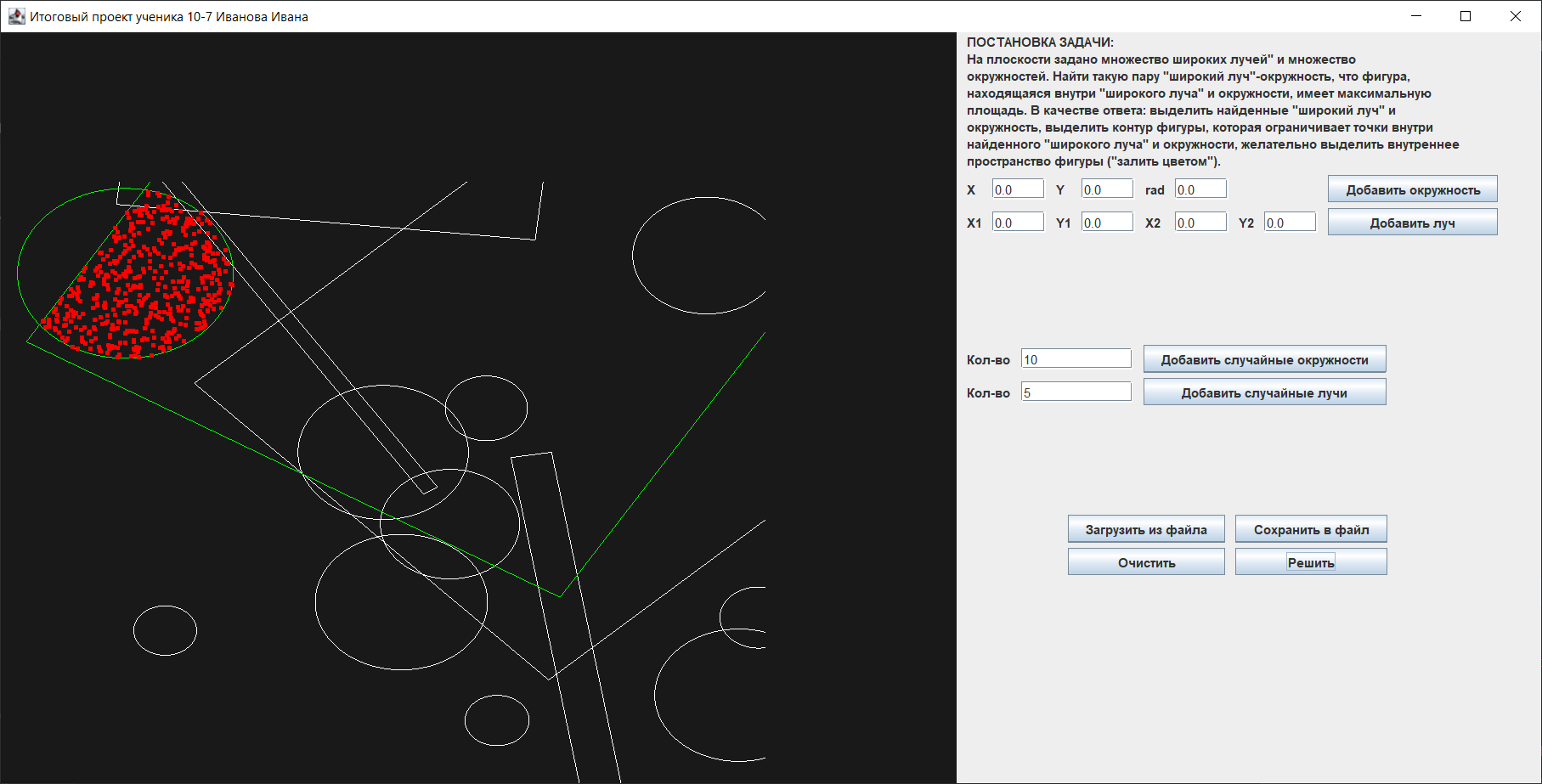
1. public void solve() { // решение  
    Random random = new Random();  
    ArrayList<Vector2> points = new ArrayList<>();  
    for (int i = 0; i < 10000; i++) {  
    points.add(new Vector2(random.nextDouble() \* 2 - 1, random.nextDouble() \* 2 - 1)); //создали 10000 случайных точек и добавили их в динамический массив  
    }  
    int maxPointCnt = 0; //создали переменную, которая будет хранить количество случайных точек, которые содержатся в максимальной площади пересечения  
    for (Ray rayloop : rays)  
    for (Circle circleLoop : circles) { //перебираем все лучи и все окружности  
    int localPointCnt = 0; //переменная, хранящая число точек, содержащихся в площади пересечения данной окружности и данного луча  
    ArrayList<Vector2> localPoints = new ArrayList<>(); //  
    for (Vector2 point : points) { //перебираем все точки if (rayloop.contains(point) && circleLoop.contains(point)) {  
    localPointCnt++;  
    localPoints.add(point);  
    }//если точка содержится и в широком луче, и в окружности, добавляем ее в массив   
    }  
    if (localPointCnt > maxPointCnt) {  
    squarePoints = localPoints;  
    maxPointCnt = localPointCnt;  
    ray = rayloop; //получаем нужный луч  
    circle = circleLoop; //получаем нужную окружность  
    }  
    }  
    if(maxPointCnt==0)  
    System.*out*.println("Не найдено пересечений");  
   }

**7. Пример работы программы**

1. Создаем 10 случайных окружностей и 5 случайных широких лучей



1. Нажимаем «Решить» и получаем нужную выделенную пару



**8. Анализ правильности решения**

Задача не требует точного вычисления площадей пересечения, нужно просто их сравнить. Именно поэтому можно сравнивать не значения площадей пересечения, а количество случайных точек, попавших в эти площади. (Чем большее количество случайных точек будет добавлено, тем точнее будет результат.)