Физико-математический лицей № 239

**Проверка множества точек**

Годовой проект по информатике

Работу выполнила ученица 103 класса Мартыненко Анастасия

Санкт-Петербург

2017

1. **Постановка задачи**

Множество точек на плоскости назовем регулярным, если вместе с каждой точкой оно содержит также еще две, такие что эти три точки являются вершинами равностороннего треугольника. Определить, регулярно ли заданное множество точек.  
Нужно проверить, что каждая точка данного множества состоит хотя бы в одном равностороннем треугольнике.

1. **Ограничение исходных и выходных данных**

**2.1 Исходные данные**

Исходные данные могут находиться в файле input.txt, либо вбиваться вручную в специально отведенные поля в окне программы.

При вводе данных в файл, сначала должно идти число n, равное количеству точек. Далее в n строчках должны содержаться координаты точек x, y: 0 ≤ x ≤ 530; 0 ≤ y ≤ 770. Для ввода данных вручную ограничения те же.

**2.2 Выходные данные**

Необходимо вывести на экран все точки, все равносторонние треугольники, а также ответ на вопрос: «Является ли множество регулярным?»

**2.3 Математическая модель**

Нахождение длины отрезка по координатам его концов.

1. **Алгоритм решения задачи**

**3.1 Необходимые структуры данных**

Point – точка.  
Поля:  
int x, y – координаты точки.  
boolean rmt – переменная, показывающая что точка является вершиной равностороннего треугольника.  
Методы:  
1) getLength(Point a) – метод, возвращающий длину отрезка, соединяющего две заданные точки.  
2) paint() – метод, рисующий объект в окне программы.  
3) isRmt() – метод, возвращающий значение переменной rmt.  
4) setRmt() – метод, присваивающий новое значение переменной rmt.

Triangle – треугольник.  
Поля:  
1)Point a, b, c – вершины треугольника  
2)double aLength, double bLength, double cLength – длины сторон треугольника  
Методы:  
1) paint() – метод, рисующий объект в окне программы.  
2)isEquilateral() – метод, проверяющий является ли треугольник равносторонним.  
3)isSame() – метод, проверяющий что вершины треугольника совпадают.  
4)max(int a, int b, int c) – метод, возвращающий максимальное из трех чисел.  
5)getTrBounds – метод, выделяющий пространство под треугольник на панели.

**3.2 Построение алгоритма**

Вводим количество точек и их координаты самостоятельно или считываем эти данные из файла. Из данных точек создается массив. Изначально все точки имеют поле boolean rmt=false.  
После введения исходных данных запускаем программу. Для каждой точки осуществляется проверка на то, что она соответствует условию, с помощью цикла. Если у точки rmt=true, то просто переходим к следующей точке. Если у точки rmt=false, то проходим еще два цикла, внутри последнего проверяем, что треугольник с вершинами в данных точках является равносторонним. Если да, то меняем значение переменной rmt и рисуем на панели равносторонний треугольник, если нет, то переходим к следующему шагу цикла.  
После прохождения трех циклов вводим переменную cnt, отвечающую за количество точек, которые не подходят под условие. В цикле проверяем, что у каждой точки rmt=true, если нет, то увеличиваем cnt на один.  
Если количество точек, не подходящих под условие, равно нулю, то множество является регулярным, если нет, то множество является не регулярным.

1. **Составление алгоритма**

**4.1 Обобщенная блок-схема**

Начало

Считывание из файла. Задание координат точек

Проверка того, что точки состоят в равносторонних треугольниках

Нахождение среди точек тех, которые не состоят ни в одном равностороннем треугольнике

Вывод результата

Конец

**4.2 Блок-схема программы**

Начало

int i=0

Чтение из файла  
ArrayList<Point> *points* = **new** ArrayList<Point>

rmt==false

Нет

Point b=*points*.get(j);  
Point c=*points*.get(k);

Triangle tr=**new** Triangle(a,b,c);

Да

Нет

(tr.isEquilateral())&&(!tr.isSame())

Да

rmt=true

int i=0;

int cnt=0

rmt==false

Нет

cnt++

Вывод ответа

cnt==0

Конец

1. **Листинг программы**

Переменная, использующаяся в нескольких кнопках. Ее значения соответствуют ответу задачи:

**private static boolean** *ma* ;

Циклы, перебирающие все возможные комбинации из 3 точек из множества заданных точек:

**for**(**int** i=0;i<*points*.size();i++){  
 Point a=*points*.get(i);  
 **if** (a.isRmt()==**false**) {  
 **for** (**int** j=0;j<*points*.size();j++){  
 **for** (**int** k=0;k<*points*.size();k++){

3 точки из множества, выбранные в циклах:   
Point a = points.get(i);   
Point b = points.get(j);   
Point c = points.get(k); 

Массив введенных точек:

**private static** ArrayList<Point> *points* = **new** ArrayList<Point>();

Создаем треугольник из этих точек:   
Triangle tr=**new** Triangle(a,b,c);

Проверка на то, что треугольник равносторонний и что его вершины не совпадают:

**if** ((tr.isEquilateral())&&(!tr.isSame())){

Считаем количество точек, не входящих ни в какой равносторонний треугольник:

**for**(**int** i=0;i<*points*.size();i++){  
 Point a=*points*.get(i);  
 **if** (a.isRmt()==**false**){  
 cnt++;  
 }  
}

1. **Пример работы программы**

Входные данные: Выходные данные:  
4 Да  
10 10  
20 10  
15 1  
15 19