Физико-математический лицей № 239

**Нахождение окружности с максимальной плотностью точек**

Отчёт о домашней работе по информатике

Работу выполнил

Ученица 10-7 класса

Малышкина Марина

Санкт-Петербург

2021

# Постановка задачи

На плоскости задано множество точек. Найти окружность, содержащую внутри себя хотя бы две точки множества, имеющую наибольшую плотность точек внутри себя (количество точек на единицу площади). В качестве ответа нарисовать найденную окружность и выделить все точки, находящиеся внутри нее.

# Уточнение исходных и выходных данных и ограничений на них

## Исходные данные

Во входном файле input.txt содержатся координаты точек в формате: Vector (x, y). Например, Vector1(0.3 0.5) Количество точек заранее неизвестно. Координаты точки задаются через запятую в одну строку. Первый символ каждой из координат может принимать одно из значений: 1 или 0. В оставшейся части строки содержится вещественное число, в котором дробная часть (если она есть) отделена точкой от целой части. Дробная часть может отсутствовать. Число может быть отрицательным (знак минус перед числом). Порядок указания координат в файле определен: первое число является абсциссой, а второе ординатой. Оба числа лежат в пределах от -1 до +1.

## Выходные данные

Необходимо вывести на экран окружность с максимальной плотностью точек. Возможно, такая окружность будет только одна.

В файле будут напечатаны координаты точек, на основе которых построена окружность.

# Выбор метода решения

## Анализ исходных данных и выбор используемой структуры данных

В задаче требуется использовать информацию о конкретном взаиморасположении точек. Анализируется расстояние между двумя конкретными точками. Размер массива с координатами точек не ограничен.

## Выбор метода решения

Считываем координаты точек по очереди. Перебираем пары и тройки точек, сдвигая индекс каждой следующей выбранной точки на один, чтобы в парах и тройках точек все точки были различными.

После чтения исходных данных приведём накопленные данные к более удобному для дальнейшей работы виду. По теореме Пифагора считаем расстояние между точками попарно и берём модуль полученного значения. Может оказаться две ситуации: две точки с минимальным расстоянием между ними, однако рядом с окружностью на бесконечно малом расстоянии от неё, тогда построение описанной окружности на треугольнике с вершинами в этих точках ведёт к бесконечно малому изменению площади и увеличению количества точек на единицу площади на 1; а вторая ситуация – та, в которой нет этой третьей точки.

# Составление алгоритма

## Обобщённая блок-схема алгоритма

Визуализация решения.

Нахождение окружности с максимальной плотностью точек

Начало

Загрузка исходных координат точек из файла.

Визуализация заданных точек

конец

## Блок-схема алгоритма

i=0, I < n

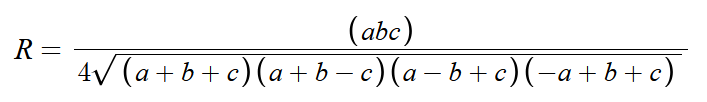
Начало

j=i+1, j < n

k=j+1, k < n

Да k >= n Нет

CircleOnRadius(pi, pj) CircleThreePoints(pi, pj, pk)

Center (a, b) = midpoint(a, b) Center (a, b, c)

(смотреть в коде систему уравнений)

Построение окружности

Конец

# Листинг программы

for (int i, i=0, i<n) {выбор первой точки}

for (int j, j=i+1, j<n) {выбор первой точки}

for (int k, k=j+1, k<n) {выбор первой точки}

if (k >= n) { на диаметре, специальный случай k=n (k выходящее за границы индекса 0..n-1 мы используем как флаг использовать пару точек на диаметре, а не тройку точек}

getCircleOnRadius(pi, pj) {окружность на радиусе}

getCircleThreePoints(pi, pj, pk) {окружность на трех точках}

# Пример работы программы

6.1. Исходные данные 6.2. Выходные данные

-0.9566662677950062 -0.6759670884621705  
-0.9279960204568036 - 0.6695216948460128

-0.9566662677950062 -0.6759670884621705  
-0.9419315084225217 -0.010643238206000216  
0.9548023209362495 -0.37703029558426593  
-0.6699665134592132 -0.44509853147294764  
-0.9279960204568036 -0.6695216948460128

# Вычислительная сложность

# В программе перебираются пары и тройки точек. Сложность по расчету пар существенно меньше (N2) по сравнению с тройками (N3). Для каждой пары и тройки точек нужно посмотреть количество попавших внутрь окружности точек (сложноть N). Это дает суммарную сложность N4. В программу был вставлен замер времени, потраченного на решение. Данные представлены в таблице.

|  |  |
| --- | --- |
| Число точек (шт.) | Время (миллисекунды)[на ноутбуке Asus] |
| 10 | 4 |
| 20 | 9 |
| 40 | 17 |
| 80 | 111 |
| 160 | 1582 |
| 320 | 22260 |
| 640 | 371165 |

# Представленная зависимость представлена в двойных логарифмических осях. При малых N перебор точек является не основной составляющей сложности вычисления, из-за чего происходит отклонение от N4. Для больших N увеличение N в 2 раза ведёт увеличению времени работы программы в 16 раз.

# 