Университет ИТМО

Факультет программной инженерии и компьютерной техники

# Лабораторная работа №2

по «Алгоритмам и структурам данных»

Выполнил:

Студент группы P3200

Шишкин Н.Д.

Преподаватели:

Косяков М.С.

Тараканов Д.С.

Санкт-Петербург

2019

## Задача №1322 «Шпион»

Пояснение к примененному алгоритму:

В данной задаче используется преобразование Барроуза-Уилера. Зная последний символ исходной строки, мы можем найти предыдущий следующим образом: мысленно циклически сдвинем вправо все строки. Тогда предпоследний символ станет последним, а последний - первым. Теперь нам надо понять, какой по порядку окажется новая строчка при лексиграфическом порядке. Во первых, она будет находиться точно ниже тех строк, которые имеют в начале символ меньший чем она. Во вторых, при одинаковом начальном символе, строчки сохранят тот же порядок что и их символы в последнем столбце, потому что при циклическом сдвиге порядок будет рассчитываться по суффиксам данных строк (первые символы одинаковые) - которые раньше были префиксам и определяли порядок до циклического сдвига.

Если произвести предподсчёт порядка одинаковых символов и для каждого уникального символа количества символов меньших него, то вычислять прошлый символ можно за O(1), в то время как сам предподсчёт занимает O(n) времени.

Сложность алгоритма - O(n)

## Задача №1604 «В стране дураков»

Пояснение к примененному алгоритму:

Заметим, что решение вообще без повторений знаков возможно всегда, кроме случаев, где максимальный по частоте порядок встречается большее число раз, чем все остальные вместе взятые (принцип Дирихле, где коробки - места между знаками, а кролики - знаки другого порядка). В противном случае распределение возможно, однако мы должны быть уверены, что после уменьшения некоторых порядков не появится ситуация, описанная ранее. Чтобы этого избежать, нам нужно постоянно поддерживать максимум среди порядков, а знаки уменьшать по одному - убирая максимальный порядок и любой другой. Тогда мы можем гарантировать следующее - выписанная нами пара содержит знаки разных порядков, а также, при выписывании данной пары мы уменьшим максимум и сумму всех кроме максимума на единицу - значит ситуации, где какой-то порядок встречается большее число раз чем все остальные не возникнет. Если же такой порядок был изначально, то просто выведем его остаток в конце.

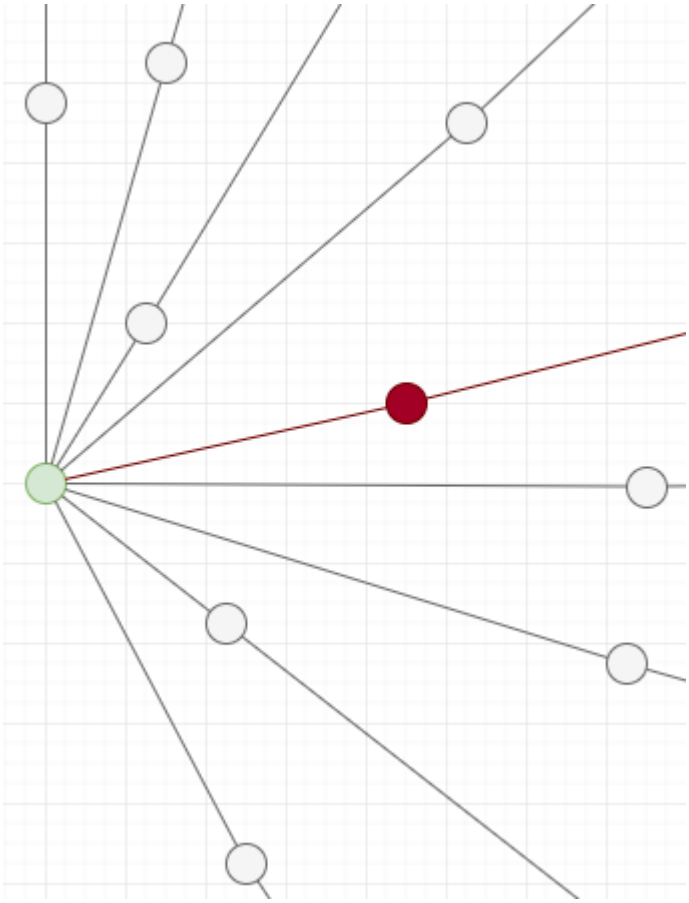
Сложность алгоритма - $(\sum_1^k n_i)*\log(k)$

## Задача №1207 «Медиана на плоскости»

Пояснение к примененному алгоритму:

Первым шагом найдем точку с минимальной абсциссой (если она одинаковая у двух точек, возьмем ту, чья ордината меньше.). Она будет первой точкой для нашего ответа.

Проведем из найденной точки лучи во все остальные, тогда на плоскости получим такую картину:



Теперь отсортируем относительно найденной точки остальные по увеличению угла (угол считаем от положительного направления оси ординат по часовой стрелки).

Тогда очевидно, что, соединив первую точку и некую отсортированную, мы разобьем плоскость на две полуплоскости, в одной из которых будут находиться все точки с углом меньшим, чем у выбранной, а в другой - больше (поскольку точки находятся только в правой полуплоскости относительно первой).

Для достижения равенства точек в каждой полуплоскости, возьмем точку в середине отсортированного массива. (Она будет определена однозначно, поскольку всего в сортированном массиве нечетное количество точек.

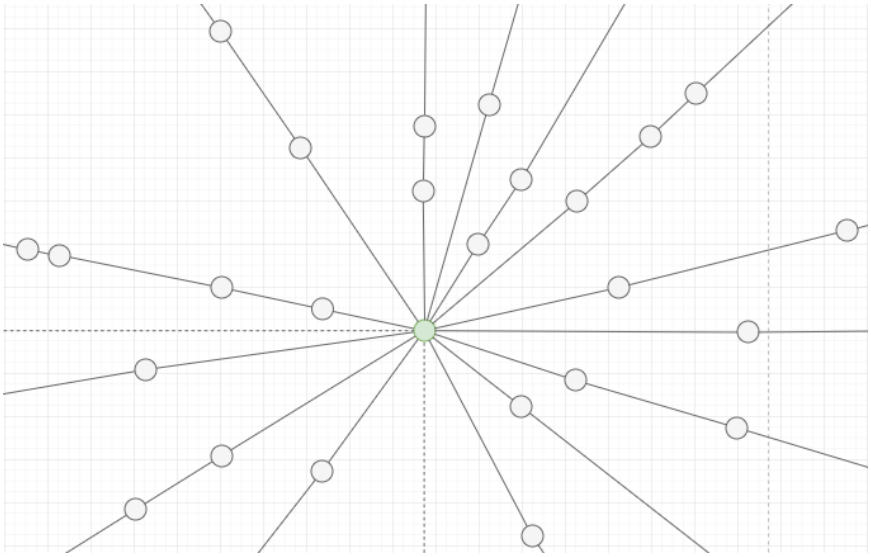
Полученные две точки являются ответом.

Сложность алгоритма: $n * log(n)$.

## Задача № 1444. «Накормить элефпотама»

Пояснение к примененному алгоритму:

Возьмем исходную точку как центр декартовой системы координат и проведем лучи во все оставшиеся точки (тыквы). В самом общем случае получим подобное изображение:

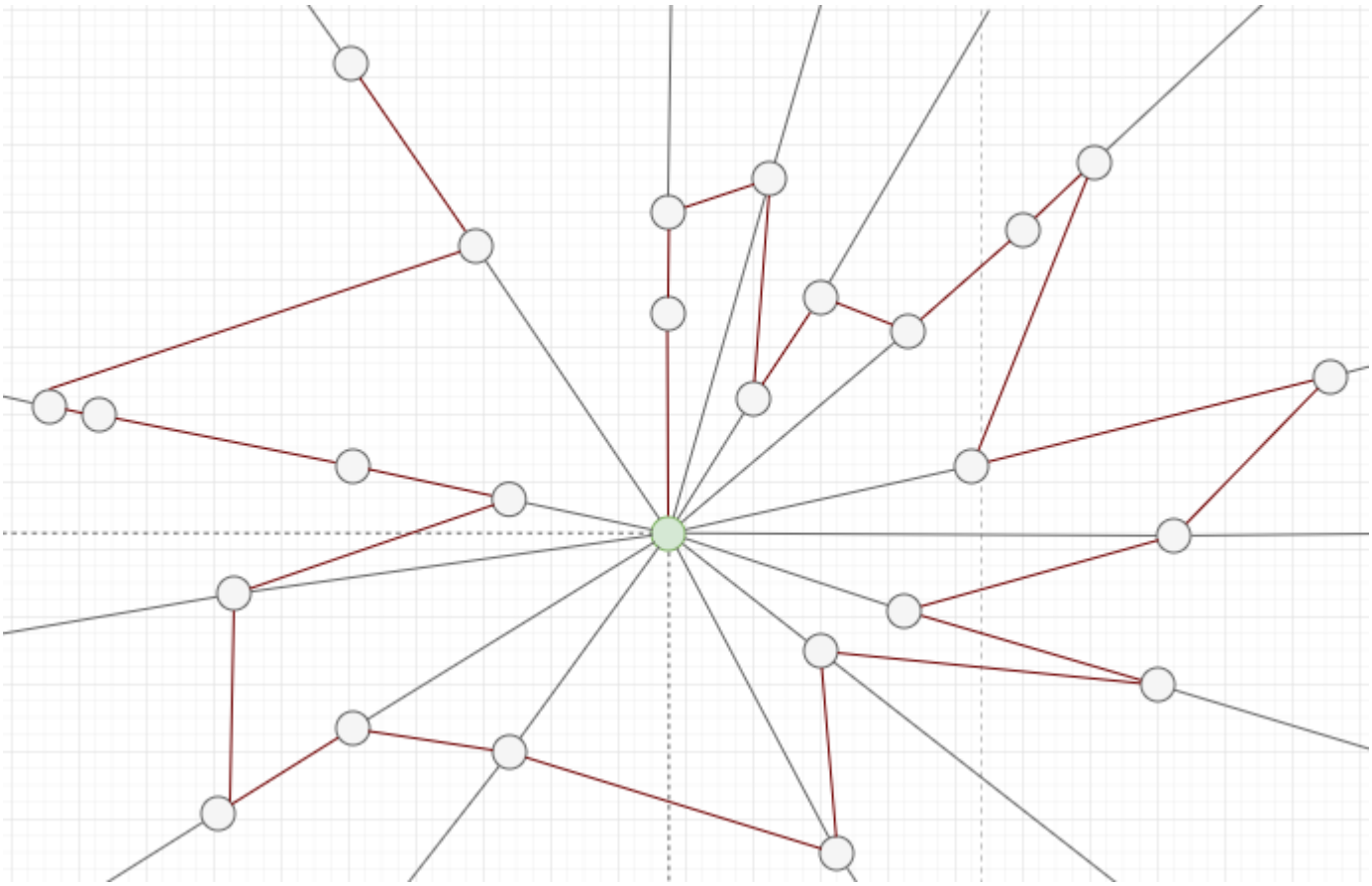


Как и в задаче с медианой отсортируем все остальные точки по углу (угол считаем от положительного направления оси ординат, если у двух точек углы равны, меньшей считается та, что ближе к центру координат).

Алгоритм обхода пути таков - последовательно идем по отсортированным точкам, в таком случае мы либо идем по одному из лучей, либо перескакиваем между двумя соседними (т. е. между ними лучей нет) лучами. Пересечений не будет, поскольку по каждому лучу мы пройдем один раз, при этом при переходе с луча на луч, линия пройдет через сектор,

ограниченный этими лучами. Внутри секций точек не может быть по условию сортировки, в то же время если угол сектора меньше 180° (случай при большем угле рассмотрен позднее), то линия между двумя радиусами лежит внутри сектора, а поскольку все сектора не пересекаются, то не будут пересекаться и линии внутри них.

Визуализация решения:



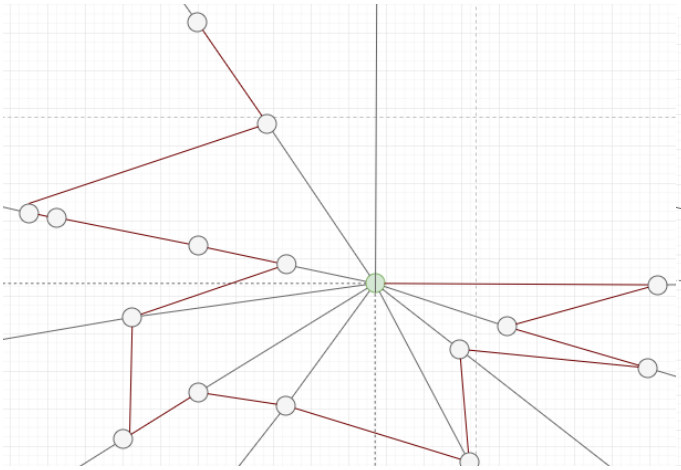
Теперь рассмотрим случай, когда угол сектора между двумя лучами больше или равен 180°

Такое возможно, когда между лучами будет находиться одна или пустых четвертей.

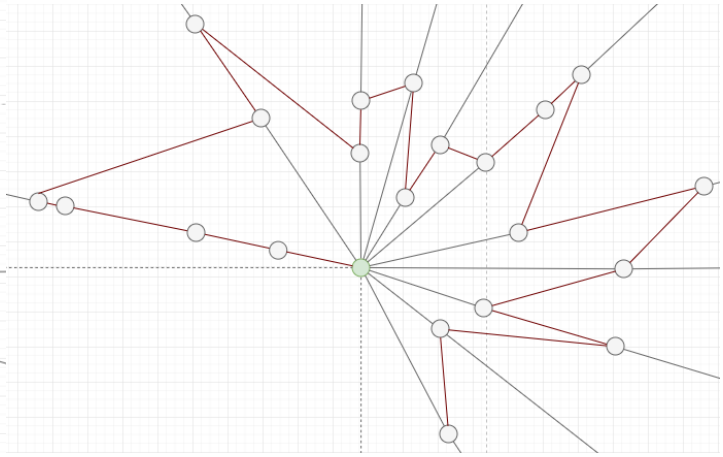
В случае одной:

Для того, чтобы избежать перемещения между такими лучами, мы будем начинать обход с четверти, после пустой. (обход четвертей считаем по часовой стрелке для определенности). В этом случае все оставшиеся точки будут лежать в трех последовательных четвертях, а значит ситуации с сектором с углом больше или равном 180° возникнуть не может. Причем чтобы начать такой обход пересортировывать массив не нужно, мы просто циклично обойдем его начиная с первой точки в выбранной четверти.

Примеры:



При пустой первой четверти

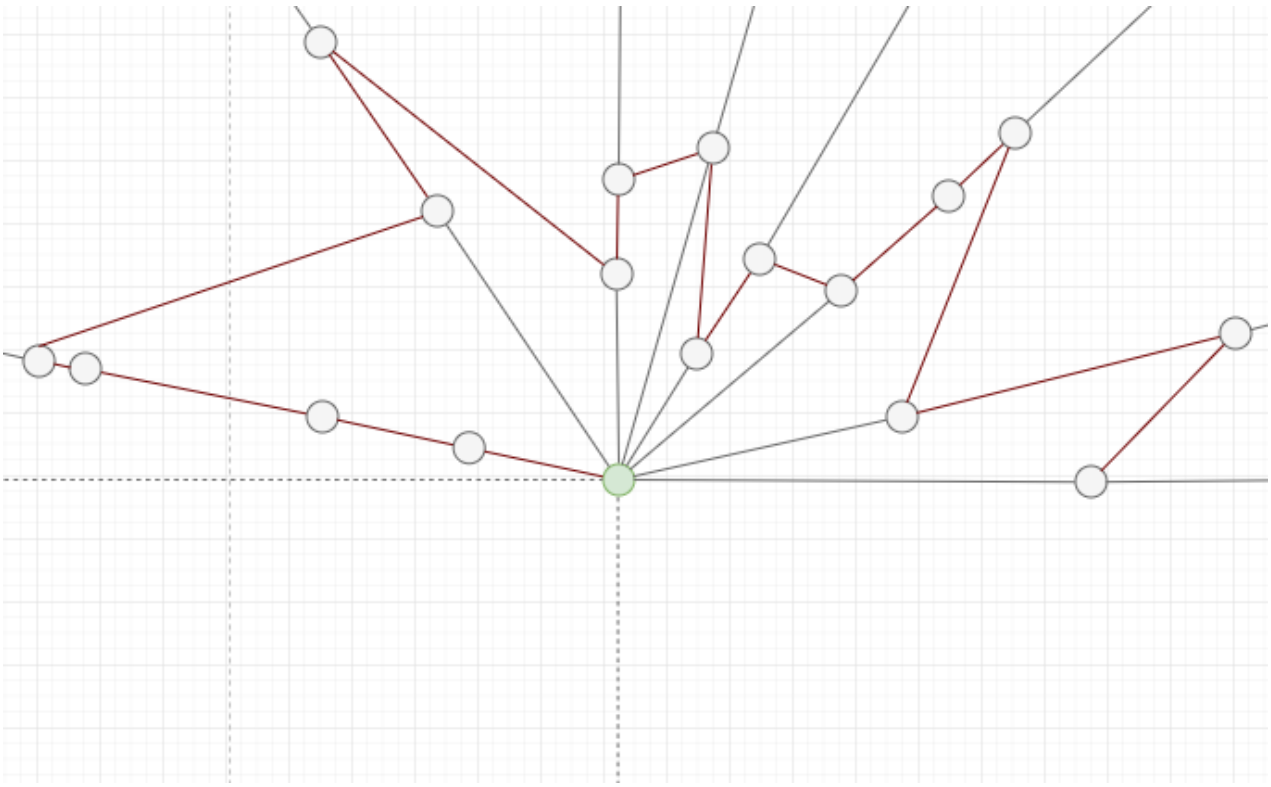


При пустой третьей четверти

В случае двух:

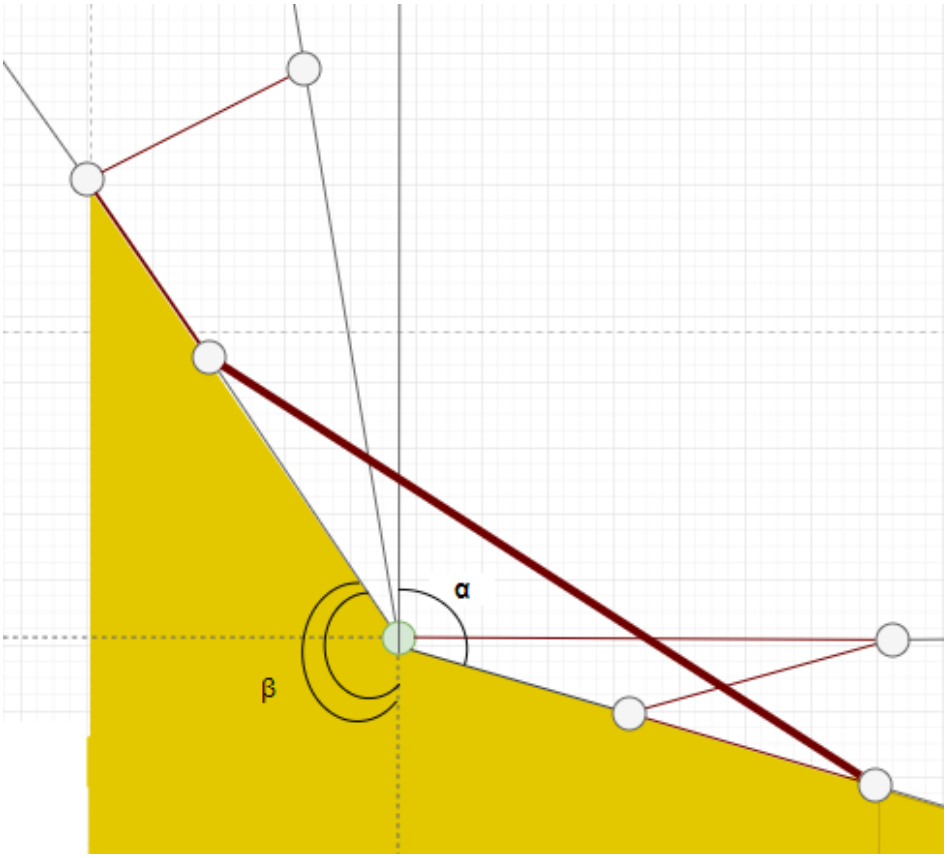
Здесь возможно два варианта. Две пустые четверти могут быть последовательными, в таком случае начнем обход с четверти после последней пустой, и аналогично прошлому случаю, мы будем обходить последовательные четверти, и сектора с углом больше или равном 180° не возникнет.

Пример:



При пустой третьей и четвертой четверти.

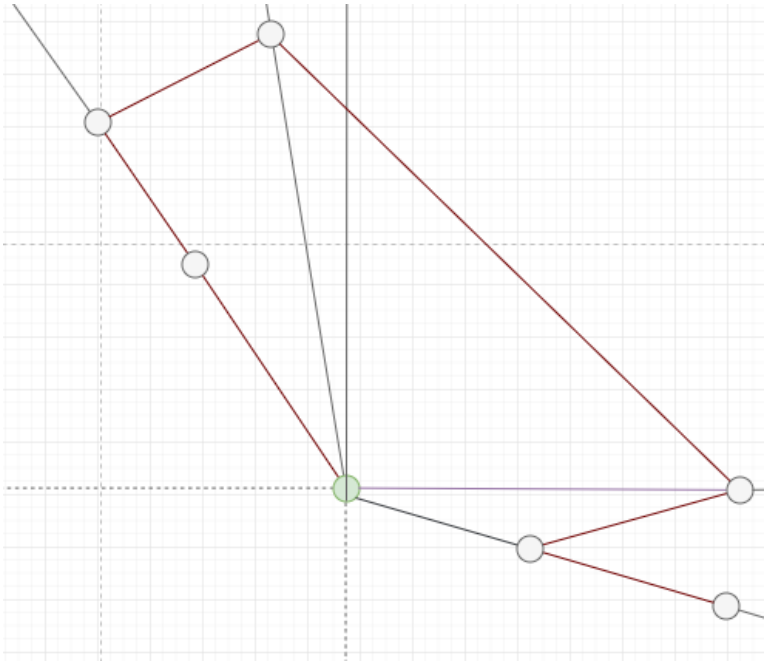
Второй вариант - пустые четверти лежат не последовательно, а через одну. В таком случае мы можем получить сектор с углом больше или равном 180° при переходе с луча из одной четверти в другую. Поэтому если просто будем последовательно идти по отсортированным точкам, мы рискуем наткнуться на следующую ситуацию:




Угол сектора (выделен желтым) больше 180°

Заметим, что данный случай возникает, только когда $\alpha \leq \beta$, потому что угол сектора $= \beta + (180° - \alpha) = 180° + (\beta - \alpha)$. В таком случае, чтобы избежать пересечения нам нужно начать обход с другой четверти, потому что в таком случае сектора с углом больше или равном 180° возникнуть не может, поскольку весь оставшийся угол кроме сектора гарантированно не больше 180° (если он равен 180°, то все равно пересечения не будет, так как внутри нового сектора лежит хотя бы один луч, который гарантированной разобьет его на секторы с меньшим углом. Луча не может не быть, потому что в таком случае мы имеем всего два луча из центра координат, угол между которыми ровно 180°. Но это значит, что все точки лежат на одной прямой, что недопустимо по условию).

Визуализация решения:



В случае трех пустых четвертей:

Менять алгоритм не нужно, поскольку внутри одной четверти сектора с углом больше или равном 180° возникнуть не может.

Сложность алгоритма - $n * log(n)$.

## Задача № 1726. «Кто ходит в гости»

Пояснение к примененному алгоритму:

Поскольку передвигаться мы можем только по дорогам параллельным осям координат, то кратчайший путь между домами с координатами $(x_1;\ y_1)$ и $(x_2;\ y_2)$ равен:

$$S = |x_2 - x_1| + |y_2 - y_1|$$

Из этой формулы видно, что суммарный путь складывается независимо из путей параллельных оси x и путей, параллельных оси y. Значит мы можем отдельно посчитать эти части общего пути и после сложить их.

Рассмотрим пути вдоль дорог, параллельных оси x. Для этого мысленно спроецируем все дома на ось x. Теперь отсортируем дома по возрастанию координаты x и последовательно будем рассматривать промежутки между соседними точками.

Допустим мы имеем путь (назовем его дорогой) между домами $x_i$ и $x_{i+1}$ и суммарное число ученых n. Тогда ниже этой дороги живут i ученых, а выше - n-I. Каждый ходит к каждому, это значит, что при походах в гости, данная дорога будет пройдена i*(n-i)*2 раз. (Всего комбинаций встреч ученых по разные стороны дороги i*(n-i) штук, и каждая встреча произойдет два раза). Последовательно перебрав все возможные пути между последовательными домами и просуммировав суммарный путь, мы получим расстояние, которые ученые преодолеют суммарно по дорогам, параллельным оси x.

Аналогично высчитаем пути по дорогам параллельным оси y, и сложим полученные результаты, получив суммарный путь, который пройдут все ученые, ходя в гости. Всего n*(n-1) раз ученые сходят в гости, значит поделив сумму путей на эту величину получим искомое расстояние.

Сложность алгоритма - $n * log(n)$.