Рассмотрим задачу 4 первого тура регионального этапа Всероссийской олимпиады школьников 2008/2009 года по информатике.

1. Треугольники

|  |  |
| --- | --- |
| Имя входного файла: | triangle.in |
| Имя выходного файла: | triangle.out |
| Максимальное время работы на одном тесте: | 3 секунды |
| Максимальный объем используемой памяти: | 64 мегабайта |

Роман достаточно давно занимается в математическом кружке, поэтому он уже успел узнать не только правила выполнения простейших операций, но и о таком достаточно сложном понятии как симметрия. Для того чтобы получше изучить симметрию, Роман решил начать с наиболее простых геометрических фигур – треугольников. Он скоро понял, что осевой симметрией обладают так называемые равнобедренные треугольники. Напомним, что треугольник называется *равнобедренным*, если его площадь положительна, и у него есть хотя бы две равные стороны.

Недавно Роман, зайдя в класс, увидел, что на доске нарисовано *n* точек. Разумеется, он сразу задумался, сколько существует троек из этих точек, которые являются вершинами равнобедренных треугольников.

**Требуется** написать программу, решающую указанную задачу.

Формат входных данных

Входной файл содержит в первой строке целое число *n* (3 ≤ *n* ≤ 1500). Каждая из последующих строк содержит по два разделенных пробелом целых числа – *xi* и *yi* , определяющих координаты *i*-ой точки. Все координаты точек не превосходят 109 по абсолютной величине. Среди заданных точек нет совпадающих.

Формат выходных данных

В выходной файл необходимо вывести ответ на вышеназванную задачу.

Пример входных и выходных данных

|  |  |
| --- | --- |
| triangle.in | triangle.out |
| 3  0 0  2 2  -2 2 | 1 |
| 4  0 0  1 1  1 0  0 1 | 4 |

Заметим сразу, что полный перебор всех точек приводит к очень медленному решению. Первую точку можно выбрать *n* способами, вторую — *n*-1 способом, третью *n*-2 способами. Всего вариантов. Даже если учесть, что обмен местами второй и третьей точек нового треугольника не дают, время работы программы, построенной по такому алгоритму, будет порядка *n*3. Это не позволит при больших *n* уложиться в отведённые 3 секунды. Поэтому будем применять другой алгоритм, рекомендованный авторами задачи в «Кратких методических рекомендациях по решению задач» (См. [1])

Перед описанием самого алгоритма обратим внимание на следующую теорему.

***Теорема 1.*** Пусть на плоскости задана прямоугольная декартова система координат с равными масштабами по обеим осям. Тогда не существует ни одного равностороннего треугольника с целочисленными координатами всех его вершин.

Доказательства этой теоремы см. в [2].

## Идея алгоритма

Чтобы получить ответ на вопрос задачи, достаточно для каждой из точек подсчитать количество равнобедренных треугольников с вершиной, противолежащей основанию, в этой точке, а потом полученные результаты сложить. В силу теоремы 1 при этом ни один из треугольников не будет учтён трижды.

## Подсчёт количества равнобедренных треугольников с вершиной в заданной точке

Пусть найдены *q* точек, находящихся на фиксированном расстоянии *d* от заданной точки A. Заметим, что все они лежат на окружности радиуса *q* c центром в точке A. Число способов выбрать по 2 точки, дополнительно к точке A для построения треугольника равно , т.к. первую точку можно выбрать *q* способами, а вторую *q*-1 способом. При этом порядок выбора точек значения не имеет. Однако некоторые из построенных таким способом треугольников могут оказаться вырожденными. Их вершины окажутся принадлежащими одной прямой. Это будет в том случае, если выбранная пара точек центрально-симметрична относительно точки A. Обозначим количество вырожденных треугольников через *s*. Тогда число равнобедренных треугольников с вершиной в точке A и боковой стороной *d* равно

Чтобы найти полное количество равнобедренных треугольников с вершиной в точке A, надо просуммировать это выражение по всем возможным *d*.

## Нахождение вырожденных треугольников

Чтобы найти центрально-симметричные относительно A точки C*i* окружности, рассмотрим углы наклона векторов к оси *x*. Центрально-симметричными будут те точки и , для которых (см. точки C и C\* на рис.). Однако вычисления углов лучше избежать, т.к. это связано с использованием тригонометрических функций, а значит и вещественных типов данных с неизбежной потерей точности вычислений.

C\*

*x*

C

A

B

α

*y*

Введём вектор длины *d,* направленный по оси *x*. Тогда вместо наклона к оси *x* можно вычислять его наклон к вектору . По определению векторного произведения , откуда

Проекция векторного произведения на ось *z* может быть вычислена через координаты векторов

Отсюда получаем формулу (1)

При этом для точек C, лежащих выше точки A (см. рис.), будет положительным, а ниже оси — отрицательным. По значению синуса его аргумент определён неоднозначно. Для установления вырожденности треугольника диаметрально противоположные относительно A точки C и C\* можно считать двумя совпадающими, потому что прямые (AC) и (AC\*) совпадают. Поэтому если , развернём в противоположную сторону, изменив знаки у его обеих координат. Вектор, направленный вниз, также развернём, направив его вверх. Рассмотрение сведётся к первой и четвёртой координатным четвертям, где функция является возрастающей, и, следовательно, обратная функция определена и тоже возрастает.

Если отсортировать все , например, по возрастанию, то значения для симметричных относительно точки A точек окажутся рядом. Чтобы найти количество вырожденных треугольников, достаточно будет один раз пройти по отсортированному массиву и подсчитать количество *s* рядом расположенных пар совпадающих значений.

Знаменатель формулы (1) вычислять не нужно, потому что он одинаков для всех . Задача сведётся к сортировке и просмотру целочисленного массива.

## Вычисление *q*

Следует сформировать массив расстояний от точки номер *i* до всех остальных. Отсортировать его, например, по возрастанию. Одинаковые расстояния окажутся рядом. За один проход по массиву определяем и *q* для различных расстояний *d*, и соответствующие .

Чтобы избежать работы с вещественными числами и потери времени на вычисление квадратных корней, лучше вместо массива расстояний использовать массив квадратов расстояний.

## О реализации

В папке triangle2009 находится реализация описанного алгоритма на Microsoft Visual C++ 2005. Для сортировки используется метод быстрой сортировки QuickSort. Функция сортировки реализована в виде шаблона, т.к. её приходится вызывать для сортировки элементов разных типов данных. Исполняемый файл находится в triangle2009\release. Время работы этой программы на моём компьютере менее 1 секунды на любом тесте. Набор тестов смотрите в [1]. При замене метода сортировки на квадратичный (прямую вставку) время работы заметно больше, но в отведённые 3 с укладывается. Полный перебор всех точек на том же компьютере осуществляется за время около 25 с.

## Литература

1. Методические рекомендации и задачи регионального этапа. <http://info.rusolymp.ru/images/informatics/methodics_informatics_3stage_2009.zip>
2. В. Вавилов, А. Устинов. Полуправильные многоугольники на решётках. //Журнал «Квант», №6, 2007 год.