

Задача А. Стекланный забор

Имя входного файла: `swamp.in`
Имя выходного файла: `swamp.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

В известном городе Санкт-Тверь решили построить новый микрорайон, представляющий в плане прямоугольную область. Границы микрорайона и его улицы по проекту ориентированы строго по сторонам света, причем улицы разбивают микрорайон на кварталы размером $1 * 1$ км.

Во время привязки исходного проекта к местности выяснилось, что некоторые кварталы по проекту микрорайона оказываются полностью или частично расположенными на топком болоте. Область, занимаемая болотом, связна и со всех сторон окружена подлежащими застройке кварталами микрорайона (область связна, если из любой ее точки можно добраться в любую другую, не выходя за пределы области).

Для сохранения экологии местности и обеспечения безопасности жителей занятую болотом область решили оградить стекланным забором. Забор должен проходить только по границам кварталов проектируемого микрорайона, отделяя болото, и, возможно, некоторые кварталы проекта, не занятые болотом, от остальной части микрорайона.

Для экономии строительных материалов забор должен иметь минимальную длину. Среди всех заборов минимальной длины нужно выбрать тот, для которого площадь части микрорайона, попадающей внутрь забора, минимальна.

Требуется написать программу, которая спроектирует забор с заданными выше свойствами.

Формат входного файла

Входной файл содержит описание многоугольника — границы области, состоящей только из кварталов с заболоченными участками. Стороны многоугольника параллельны осям координат. В первой строке задано целое число n — количество вершин в многоугольнике ($4 \leq n \leq 100000, n$). В каждой из следующих n строк заданы два целых числа — координаты очередной вершины при обходе этого многоугольника против часовой стрелки. Все числа не превосходят 109 по абсолютной величине. Никакие три последовательные вершины границы не лежат на одной прямой. Граница многоугольника не содержит самопересечений и самокасаний.

Формат выходного файла

Выходной файл должен содержать описание многоугольника, определяющего

искомый забор. Формат описания многоугольника тот же, что и для входных данных. Никакие три последовательные вершины этого многоугольника не должны лежать на одной прямой.

Пример

<code>swamp.in</code>	<code>swamp.out</code>
8	6
0 0	0 0
9 0	9 0
9 9	9 9
6 9	6 9
6 3	6 6
3 3	0 6
3 6	
0 6	

Задача В. Треугольная реформа

Имя входного файла: `reform.in`
Имя выходного файла: `reform.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Король Полигонии Трианг IV помешан на реформах. Чтобы войти в мировую историю, он решил провести территориальную реформу в своей стране. Страна Полигония имеет форму простого многоугольника, то есть ее граница не имеет самопересечений и самокасаний. В Полигонии большую роль играют внутренние диагонали — отрезки, соединяющие вершины государственной границы и полностью проходящие по территории страны, не касаясь границы. При этом форма Полигонии такова, что никакие две внутренние диагонали не лежат на одной прямой.

Вместо традиционного деления государства на административные округа король Трианг IV решил разделить свою страну на административные треугольники внутренними диагоналями. Для сокращения управляющего аппарата король повелел подготовить такой план деления страны, в котором количество административных треугольников будет минимальным.

Требуется написать программу, выполняющую деление Полигонии внутренними диагоналями на минимально возможное число административных треугольников.

Формат входного файла

В первой строке входного файла записано число $N(3 \leq N \leq 20)$ — количество вершин многоугольника, образующего границу Полигонии. В следующих N строках находятся по 2 целых числа, по абсолютной величине не превосходящие 10 000 — координаты вершин в порядке обхода многоугольника против часовой стрелки. Гарантируется, что никакие три последовательные вершины многоугольника не лежат на одной прямой, и он не имеет самопересечений и самокасааний. Также гарантируется, что никакие две диагонали, содержащиеся внутри многоугольника, не лежат на одной прямой.

Формат выходного файла

В первую строку выходного файла выведите наименьшее количество административных треугольников, на которое можно разбить Полигонию.

Во вторую строку выведите количество различных внутренних диагоналей K , с помощью которых можно произвести данное разбиение.

В каждую из следующих K строк выведите 4 целых числа — координаты начала и конца соответствующей диагонали разбиения, полностью лежащей внутри многоугольника и не проходящей по его границе.

Если искомым разбиений несколько, выведите любое из них.

Пример

reform.in	reform.out
4	2
0 0	1
1 0	1 1 0 0
1 1	
0 1	
10	4
-6 0	3
0 2	2 4 -2 4
6 0	0 2 3 3
3 3	-3 3 0 2
6 4	
2 4	
0 6	
-2 4	
-6 4	
-3 3	

Задача С. Пересечение многоугольников

Имя входного файла: polint.in
Имя выходного файла: polint.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Два многоугольника на плоскости заданы координатами своих вершин. Требуется вычислить площадь пересечения этих многоугольников, то есть сумму площадей тех кусков, которые образуются при их пересечении и принадлежат каждому из них. При этом вы можете предполагать, что:

— Многоугольники выпуклые, а координаты их вершин даны в произвольном порядке.

— Хотя бы один из многоугольников невыпуклый, но известно, что у каждого из многоугольников не более одного угла, большего 180 градусов, а координаты вершин даны в порядке обхода по часовой стрелке.

Ваша программа по входным данным должна сама определить, какой из этих двух случаев имеет место.

Формат входного файла

Первая строка входного файла содержит целое число N — количество вершин в первом многоугольнике ($3 \leq N \leq 50$). Во второй строке записаны координаты этих вершин. Третья и четвертая строки таким же образом задают второй многоугольник. Координаты всех вершин являются целыми числами из диапазона $[-32768, 32767]$.

Формат выходного файла

Выведите в выходной файл искомую площадь не менее чем с 6 верными значащими цифрами.

Пример

polint.in	polint.out
3	2.0
0 3 0 -3 -3 0	
5	
-1 1 2 1 1 0 2 -1 -1 -1	

Задача D. Пересечение окружностей

Имя входного файла: `cirint.in`
Имя выходного файла: `cirint.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

На плоскости провели N окружностей. Требуется определить площадь их пересечения.

Формат входного файла

В первой строке входного файла находится целое число N ($1 \leq N \leq 20$). В каждой из последующих N строк записана тройка вещественных чисел, описывающих очередную из окружностей. Первые два числа задают координаты ее центра, третье — радиус.

Формат выходного файла

Выведите в выходной файл искомую площадь не менее чем с 6 верными значащими цифрами.

Пример

<code>cirint.in</code>	<code>cirint.out</code>
2	0.570796
0 0 1	
1 1 1	

Задача E. Максимальное расстояние

Имя входного файла: `maxdist.in`
Имя выходного файла: `maxdist.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

На плоскости задано N точек. Найти расстояние между двумя наиболее удаленными друг от друга точками.

Формат входного файла

Первая строка входного файла содержит число N ($2 \leq N \leq 100000$). Далее в файле записано N пар целых чисел, задающих координаты точек. Все координаты по модулю не превышают 10^4 .

Формат выходного файла

В выходной файл выведите пару чисел — номера точек, для которых достигается максимум расстояния. Точки нумеруются начиная с 1. Если решений несколько, то следует вывести любое из них.

Примеры

<code>maxdist.in</code>	<code>maxdist.out</code>
4	4 3
0 1	
1 0	
0 0	
1 1	

Задача F. Коза

Имя входного файла: `koza.in`
Имя выходного файла: `koza.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

На поле, где растет трава, имеется N заборов. В центре поля вбит столбик, к которому привязана коза. Длина веревочки, которой она привязана, равна L . Коза может перемещаться по полю и есть траву. При этом она не может перепрыгивать через забор и веревочка, которой она привязана, тоже ни когда не пересекает забор (но может огибать его с одного из концов). Коза имеет сколь угодно малые, но ненулевые размеры. Чтобы выжить, козе требуется каждый день съесть ровно $1m^2$ травы. В последний день своей жизни коза может съесть и меньше травы. Таким образом, если площадь доступной для питания козы части поля составляет $1, 1m^2$, коза гарантированно проживет 2 дня, если $23, 78m^2$ — 24 дня...

Формат входного файла

Во входном файле записаны целые числа L и N — длина веревочки и количество заборов ($0 \leq L \leq 40, 0 \leq N \leq 250$). Далее идут описания заборов — четверки целых чисел $X1, Y1, X2, Y2$ ($|X_i|, |Y_i| \leq 100$). Все заборы представляют собой горизонтальные или вертикальные отрезки. Заборы могут пересекаться, накладываться, состоять из 1 точки, но ни один из них не проходит через точку $(0, 0)$ — место, где привязана коза.

Формат выходного файла

Выведите в выходной файл количество дней, которое проживет коза. Ваш ответ может отличаться от правильного на 1.

Примеры

koza.in	koza.out
4 5 0 1 0 3 0 1 1 1 1 1 1 -1 1 -1 -1 -1 -1 -1 -1 0 -1 0 -3 0	20

Задача G. Восьмиугольники

Имя входного файла: `octagons.in`
Имя выходного файла: `octagons.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

На плоскости расположены n правильных восьмиугольников. Ваша задача — сосчитать число целых точек (то есть, точек, у которых обе координаты целые), покрытых как минимум k из этих многоугольников.

Формат входного файла

В первой строке дано целое число $1 \leq n \leq 500$. Последующие n строк описывают восьмиугольники. Каждый восьмиугольник задан координатами окружности окружности (x_c, y_c) и ее радиусом r_c . Восьмиугольники повернуты таким образом, что они содержат только горизонтальные, вертикальные и диагональные отрезки. Координаты и радиусы — целые числа, не превосходящие 2000 по абсолютной величине.

Файл завершается строкой, содержащей число $1 \leq k \leq n$.

Формат выходного файла

Выведите число целых точек, покрытых одновременно как минимум k восьмиугольниками. Точка считается покрытой восьмиугольником, если она лежит внутри или на стороне, или является вершиной.

Пример

octagons.in	octagons.out
3 0 0 1 0 0 1 -1 0 1	1