Задача A. Гигантский квадратный коллайдер

Имя входного файла:	`collider2.in`
Имя выходного файла:	`collider2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

После того, как Бильбо и Фродо построили Большой огромный коллайдер, Гэндальф рассказал им о том, что на другом конце Средиземья строят *Гигантский квадратный надземный коллайдер*. Он представляет собой огромный квадрат, расположенный параллельно поверхности Средиземья на некотором расстоянии от земли.

Хоббиты, конечно же, захотели сразу построить у себя такой же. Как ни странно, основной проблемой для них стала постройка опор для этого коллайдера. Сами опоры должны обязательно располагаться в вершинах квадрата. Постройка опоры оказалось очень трудоемкой задачей, потому в качестве некоторых опор было предложено использовать высокие деревья.

Таким образом, задача свелась к тому, чтобы найти, какое наименьшее количество опор нужно построить так, чтобы среди этих опор и уже существующих в Хоббитании деревьев нашлись четыре, расположенные в вершинах некоторого квадрата.

Формат входного файла

Первая строка входного файла содержит n — число деревьев в Хоббитании. Следующие n строк содержат по два целых числа — координаты деревьев. Никакие два дерева не расположены в одной точке.

Ограничения

- Для числа n выполняется неравенство $1 \le n \le 2000$.
- Координаты вершин не превосходят по модулю 10^8.

Формат выходного файла

В первой строке выходного файла выведите k — минимальное количество опор, которые необходимо построить. В следующих k строках выведите по 2 целых числа, не превышающих по модулю 10^9 — координаты этих опор.

Примеры

collider2.in	collider2.out
2 0 1 1 0	2 0 0 1 1
3 0 2 2 0 2 2	1 0 0
4 -1 1 1 1 -1 -1 1 -1	0

Задача B. Палиндромность — 4

Имя входного файла:	`paly4.in`
Имя выходного файла:	`paly4.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Напомним, что *палиндромом* называется строка, которая читается одинаково как слева направо, так и справа налево. Например, палиндромами являются строки «abba» и «madam».

Для произвольной строки s введем операцию *деления пополам*, обозначаемую $half(s)$. Значение $half(s)$ определяется следующими правилами:

- Если s не является палиндромом, то значение $half(s)$ не определено;
- Если s имеет длину 1, то значение $half(s)$ также не определено;
- Если s является палиндромом четной длины $2m$, то $half(s)$ — это строка, состоящая из первых m символов строки s;
- Если s является палиндромом нечетной длины $2m+1$, большей 1, то $half(s)$ — это строка, состоящая из первых m символов строки s.

Например, значения $half(\text{informatics})$ и $half(\text{i})$ не определены, $half(\text{abba}) = \text{ab}$, $half(\text{madam}) = \text{ma}$.

Палиндромностью строки s будем называть максимальное число раз, которое можно применить к строке s операцию деления пополам, чтобы результат был определен.

Например, палиндромность строк «informatics» и «i» равна 0, так как к ним нельзя применить операцию деления пополам даже один раз. Палиндромность строк «abba» и «madam» равна 1, а палиндромность строки «totottotot» равна 3, поскольку операция деления пополам применима к ней три раза: «totottotot» → «totot» → «to».

Вам задан набор строчных букв латинского алфавита L. Необходимо составить из этих букв строку, обладающую наибольшей палиндромностью.

Формат входного файла

Первая строка входного файла содержит 26 целых чисел: C_a, C_b, C_c, … , C_y, C_z — количество вхождений каждой из строчных букв латинского алфавита в набор L.

Ограничения

- Для любой строчной буквы латинского алфавита ch справедливо неравенство $0 \le C_{ch} \le 10^6$;
- Для чисел C_a, … , C_z справедливо неравенство $1 \le \sum\limits_{ch=a}^{z} C_{ch} \le 10^6$.

Формат выходного файла

В выходной файл выведите искомую строку. Если возможных вариантов ответа несколько, выведите любой из них.

Примеры

paly4.in
`1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0`
paly4.out
`ab`

paly4.in
1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
paly4.out
bab

paly4.in
2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
paly4.out
abba

paly4.in
2 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
paly4.out
aczca

Задача C. Обратная задача для RMQ

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

Обратные задачи представляют собой быстро развивающуюся область информатики. В отличии от классической постановки задачи, где по заданным исходным данным D требуется решить некоторую оптимизационную задачу P, в обратной задаче по заданной задаче P и результату вычисления R требуется подобрать исходные данные D, на которых достигается этот результат. В этой задаче вам предлагается решить обратную задачу к задаче о минимуме на отрезке (range minimum query, RMQ).

Пусть задан массив $a[1..n]$. Ответ на запрос о минимуме на отрезке $Q(i, j)$ — это минимальное среди значений $a[i], \ldots, a[j]$. Вам дано n и последовательность запросов о минимуме на отрезке с ответами. Восстановите исходный массив a.

Формат входного файла

Первая строка входного файла содержит n — размер массива, и m — количество запросов ($1 \le n, m \le 100\,000$). Следующие m строк содержат по три целых числа: числа i, j и q означают, что $Q(i, j) = q$ ($1 \le i \le j \le n$, $-2^{31} \le q \le 2^{31} - 1$).

Формат выходного файла

Если входные данные несовместны, то есть искомого массива a не существует, выведите “`inconsistent`” на первой строке выходного файла.

В противном случае выведите “`consistent`” на первой строке выходного файла. Вторая строка должна содержать сам массив. Элементы массива должны быть целыми числами между -2^{31} и $2^{31} - 1$. Если решений несколько, выведите любое.

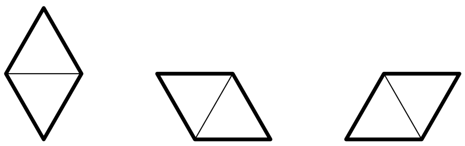
Примеры

rmq.in	rmq.out
3 2 1 2 1 2 3 2	consistent 1 2 3
3 3 1 2 1 1 1 2 2 3 2	inconsistent

Задача D. Треугольное замощение

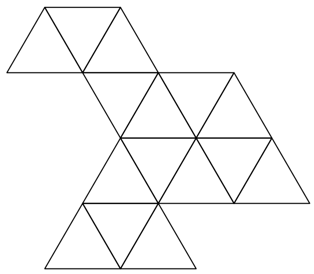
Имя входного файла:	`tiling.in`
Имя выходного файла:	`tiling.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

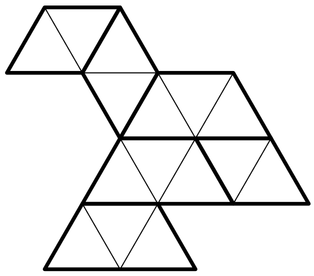
Треугольное домино представляет собой два равносторонних треугольника, имеющих одну общую сторону.



Аналогично, треугольное тримино прсдставляет собой трапецию, составленную из трех равносторонних треугольника (на рисунке показаны три из шести возможных треугольных тримино).

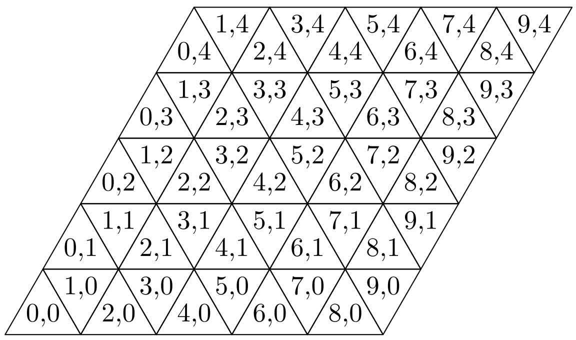
Рассмотрим фигуру на треугольной сетке, составленную из единичных треугольников. Требуется замостить ее треугольными домино и тримино, либо выяснить, что это сделать невозможно. Каждый треугольник фигуры должен быть покрыт ровно одним домино или тримино.





Формат входного файла

Первая строка входного файла содержит число n — количество треугольников в фигуре ($1 \le n \le 500$). Следующие n строк содержит координаты треугольников. Координаты введены на треугольной сетке способом, показанным на рисунке. Координаты не превышают 500 по модулю.



Формат выходного файла

Если решения не существует, выведите в выходной файл фразу “`No solution`”. В противном случае на первой строке выходного файла выведите k — количество использованных фигурок домино и тримино. Занумеруем фигурки от 1 до k в произвольном порядке. Вторая строка должна содержать n чисел от 1 до k — для каждого треугольника исходной фигуры выведите номер фигурки, которым он покрыт.

Пример

tiling.in	tiling.out
15 0 0 1 0 2 0 0 1 1 1 2 1 3 1 4 1 -1 2 0 2 1 2 2 2 -2 3 -3 3 -4 3	6 1 1 1 2 2 2 3 3 4 5 5 5 4 6 6

Пример выходного файла соответствует иллюстрации к условию.