

Задача 1. Поедатель пиццы

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

На плоском, бесконечно простирающемся во все стороны столе лежат несколько пицц. Каждая пицца имеет форму круга, при этом пиццы друг на друга не накладываются. Поедатель пиццы, представляющий собой отрезок, способен двигаться только вдоль перпендикулярной ему прямой. Двигаться он может в обе стороны, меняя направление сколько угодно раз.

Поедатель может съесть части пицц, находящиеся в точках, которые он может посетить. Требуется найти суммарную площадь частей пицц, которые могут быть съедены поедателем.

Формат входного файла

Первая строка входного файла содержит четыре целых числа x_1, y_1, x_2, y_2 — координаты концов поедателя пиццы в его начальном положении. Во второй строке записано целое число N — количество пицц ($1 \leq N \leq 100000$). Затем следуют N строк, содержащих целые числа x_{ci}, y_{ci} и r_i , задающие координаты центра и радиус каждой из пицц. Все координаты, заданные в файле, не превосходят 10000 по абсолютной величине.

Формат выходного файла

В выходной файл необходимо вывести одно число — суммарную площадь съеденных частей пицц с точностью не менее пяти знаков после десятичной точки.

Примеры

input.txt	output.txt
0 0 0 10 1 1 1 2	10.109631
2 0 0 2 3 -4 0 1 0 0 1 4 0 1	3.141593

Задача 2. Фломастеры

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Пете подарили набор из N разноцветных фломастеров. Их цвет, так же как и цвет бумаги, можно представить тройкой чисел (R, G, B) , соответствующих интенсивности красной, зелёной и синей цветовых компонент. Будем считать, что эти значения являются целыми положительными числами, не превосходящими 255.

При рисовании фломастер изменяет цвет бумаги следующим образом: если бумага имела цвет (R_0, G_0, B_0) , то после того, как на ней нарисовали фломастером цвета (R, G, B) , цвет её закрашенных областей стал $(R_0 \& R, G_0 \& G, B_0 \& B)$, где $A \& B$ — результат применения побитовой логической операции «и».

Логическая операция «и» для битов определяется в соответствии со следующей таблицей:

A	B	$A \& B$
0	0	0
0	1	0
1	0	0
1	1	1

Результатом побитовой логической операции, применённой к двум целым числам A и B , является целое число, i -ый бит которого равен результату применения одноимённой логической операции произведённой над i -ым битом числа A и i -ым битом числа B .

Петя хочет узнать, какие цвета могут получиться, если рисовать фломастерами из его набора на белой (то есть имеющей цвет $(255, 255, 255)$) бумаге. Помогите Пете!

Рисовать разными фломастеарми по одной и той же области бумаги можно бесконечно много раз.

Формат входного файла

Первая строка входного файла содержит количество фломастеров ($1 \leq N \leq 18$). Следующие N строк содержат тройки чисел R_i, G_i и B_i — компоненты цвета фломастеров.

Формат выходного файла

В первую строку выходного файла необходимо вывести целое число C — количество возможных цветов. Затем выведите C строк, содержащих значения цветов в том же формате, что используется во входном файле. Цвета можно выводить в любом порядке.

Примеры

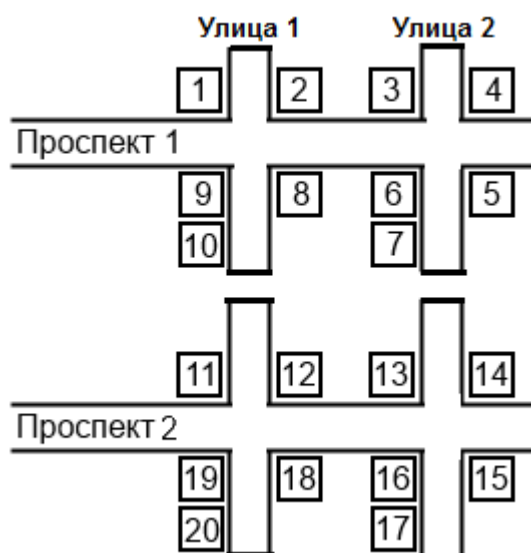
input.txt	output.txt
2 64 64 0 192 128 0	4 64 0 0 64 64 0 192 128 0 255 255 255
3 255 255 0 255 0 255 0 255 255	8 0 0 0 0 0 255 0 255 0 0 255 255 255 0 0 255 0 255 255 255 0 255 255 255

Задача 3. Дачная нумерация

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Планировка дачного посёлка выглядит следующим образом. Основу поселка составляют несколько больших параллельных проспектов, каждый проспект пересекает одинаковое количество улиц, заканчивающихся слева и справа от проспекта тупиками. Дачи строятся в тупиках. В каждом тупике с левой стороны от проспекта находится по K_1 домов, а с правой стороны от него построено по K_2 домов. Улицы нумеруются с единицы, начиная от начала каждого проспекта.

В этом дачном поселке каждый дом имеет свой уникальный номер. Нумерация домов начинается на первом проспекте с единицы, идёт по возрастанию от начала проспекта по его левой стороне, захватывая все дома в тупиках, которые выходят на эту сторону. Дойдя до конца проспекта, нумерация домов продолжается по его противоположной стороне и идёт по возрастанию к его началу. Таким же образом идет нумерация домов на всех остальных проспектах. Первые дома на втором и следующих проспектах имеют номер на единицу больше номера последнего дома на предыдущем проспекте.



Найти расположение определенного дома при такой нумерации очень сложно. Вам известен номер дома, напишите программу, вычисляющую номер проспекта и номер улицы, на которой находится этот дом.

Формат входного файла

В первой строке записано через пробел пять целых чисел N , M , K_1 , K_2 и T , где N — количество проспектов, M — количество улиц на каждом проспекте, K_1 — количество домов в каждом тупике, примыкающем к левой стороне проспекта, K_2 — количество домов в каждом тупике, примыкающем к его правой стороне, T — номер дома, для которого нужно определить его расположение ($1 \leq N, M, K_1, K_2 \leq 10^{18}$, $1 \leq T \leq 10^{18}$).

Формат выходного файла

В выходной файл необходимо вывести два целых числа — номер проспекта и номер улицы, на которых находится дом.

Пример

input.txt	output.txt
2 2 2 3 16	2 2

Комментарий

На рисунке изображена нумерация, соответствующая примеру. Искомый дом находится на второй улице проспекта номер 2.

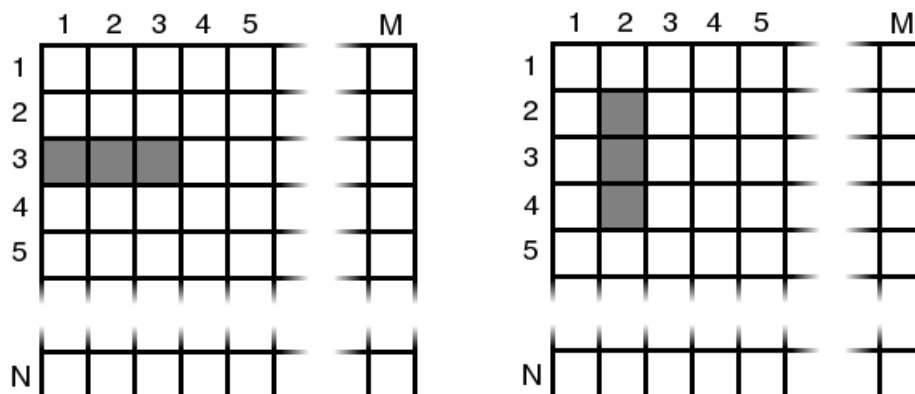
Задача 4. Широкий робот

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Робот Curiosity, уже почти два года исследующий поверхность Марса и пытающийся найти там воду, оказался в нелёгком положении. В поисках наилучшего места для съёмки очередной панорамы из марсианского кратера он заблудился среди сложнопроходимых марсианских холмов, и теперь не может найти выход из кратера.

Поверхность кратера на Марсе можно отобразить на карту, представленную в виде прямоугольной сетки, состоящей из квадратных клеток единичного размера, образующих N строк и M столбцов. Левая верхняя клетка на карте имеет координаты $(1, 1)$, ось OY направлена вниз, ось OX — вправо. Каждая клетка может быть либо непроходимым холмом, либо проходимой равниной.

Задача поиска выхода из кратера осложняется тем, что Curiosity — достаточно широкий робот длиной в одну клетку, а шириной в целых три! Центром робота называется клетка, расположенная между двумя крайними. Однако, он достаточно манёвренный: робот может ехать как назад, так и вперёд, также он может поворачиваться на 90° , но не может двигаться боком вправо или влево. Робот преодолевает путь в одну клетку за одну минуту, один поворот на 90° осуществляется за три минуты, более того, Curiosity способен совершать несколько поворотов подряд за 3 минуты каждый. Повороты совершаются относительно центра, на картинке показан пример поворота робота, центр которого находится в клетке с координатами $(2, 3)$.



Стоит не забывать, что передвижению робота мешают марсианские холмы: для движения вперёд или назад необходимо, чтобы все 3 клетки, на которые передвигается робот, не были холмами, а для поворота необходимо, чтобы центр робота и 8 соседних клеток, имеющих с центром хотя бы одну общую точку, так же не являлись холмами.

Срок работы робота ограничен, а потому требуется вывести его из кратера как можно скорее. Ваша задача состоит в том, чтобы помочь сотням учёных, и найти минимальное время, за которое Curiosity может доехать до границы кратера. Центр робота должен оказаться на одной из клеток с координатами $(x, 1)$, (x, N) , $(1, y)$ или (M, y) , где $1 \leq x \leq M$, $1 \leq y \leq N$. Известно, что выход из кратера существует, иначе робот просто не смог бы в него попасть.

Формат входного файла

В первой строке входного файла записано через пробел два целых числа N и M — размеры карты по вертикали и горизонтали ($3 \leq N, M \leq 1000$). В следующих N строках записана сама карта. Каждая строка карты содержит ровно M символов. Символ '#' означает непроходимый холм, '.' — проходимую равнину, '*' — проходимую равнину, на которой находится робот. Гарантируется, что входной файл содержит ровно три символа '*', идущих подряд в одной строке или в одном столбце.

Формат выходного файла

В выходной файл необходимо вывести одно целое число — минимальное время в минутах, которое потребуется роботу, чтобы выбраться из кратера.

Примеры

input.txt	output.txt
4 5 #...# #### #...# #####	1
4 5 #.*.# #.*.# #.*.# #####	4
6 6 ##...# #....# #....# ####.# #....# ##..##	10