Задача 1. Графический интерфейс

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

Компания, в которой вы работаете, занимается созданием нового оконного графического интерфейса. Это обусловлено тем, что все существующие реализации такого интерфейса обладают одним фатальным недостатком: их сделал кто-то другой. Окна, в которых отображаются данные, представляют собой прямоугольники, состоящие из пикселей. Стороны окон параллельны сторонам экрана. Вам поручена задача: написать модуль, который определяет взаимное расположение окон.

Согласно техническому заданию, модуль должен получать на вход положение двух окон A и B, и выдавать вердикт о том, как они расположены относительно друг друга. Возможны следующие вердикты взаимного расположения:

- `A in B` — все пиксели окна A являются пикселями окна B;
- `B in A` — все пиксели окна B являются пикселями окна A;
- `Separate` — никакой пиксель не принадлежит окнам A и B одновременно;
- `Intersect` — во всех остальных случаях.

Положение окна задаётся координатами двух пикселей: в левом верхнем его углу и в нижнем правом. Пиксель с координатами (0, 0) находится в левом верхнем углу экрана. Координаты возрастают при движении вправо и вниз.

Гарантируется, что положения окон A и B не совпадают.

Формат входных данных

В первой строке входного файла содержится одно целое число N — количество наборов входных данных для вашего модуля ($1 \leqslant N \leqslant 1000$). Затем следуют N пар строк.

Каждый набор входных данных описывается в двух строках: в первой из них указано положение окна A, а во второй — положение окна B. Для каждого окна это четыре целых числа: координаты X_l, Y_t левого верхнего и координаты X_r, Y_b правого нижнего пикселя ($0 \leqslant X_l \leqslant X_r \leqslant 10^5$, $0 \leqslant Y_t \leqslant Y_b \leqslant 10^5$).

Формат выходных данных

Для каждого набора входных данных в выходной файл нужно вывести в отдельную строку вердикт взаимного расположения окон.

Пример

input.txt	output.txt
3 0 0 3 3 1 1 2 2 0 0 9 9 10 0 19 9 1 0 3 2 0 1 2 3	B in A Separate Intersect

Иллюстрация

Окно A —

Окно B —

Случай 1

(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)
(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)
(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

Случай 3

(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)
(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)
(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

Задача 2. Поиск на кубе

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Два друга играют в придуманную настольную игру: поиск клада на кубе. Сначала ведущий загадывает размер куба, а также положение клада и начальное положение робота на этом кубе. Затем игрок пытается найти клад по принципу «теплее / холоднее», вслепую управляя роботом.

Игровое поле представляет собой куб размера $N \times N \times N$, каждая грань которого разбита на $N \times N$ клеток. Согласно правилам игры, ведущий может установить размер куба N в диапазоне от 3 до 300 включительно. В одной из $6N^2$ клеток на поверхности куба ведущий размещает клад, который остаётся в этой клетке в течение всей игры. Также изначально в одну из клеток ведущий ставит робота, повернув его в одном из четырёх направлений так, чтобы он смотрел параллельно одной из сторон клетки.

Игра разбивается на шаги, на каждом из которых игрок даёт роботу одну команду. Есть четыре типа команд:

- `left` — робот поворачивается на 90 градусов влево, оставаясь в той же клетке;
- `right` — робот поворачивается на 90 градусов вправо, оставаясь в той же клетке;
- `forward` — робот проезжает вперёд одну клетку;
- `dig` — робот копает клад в той клетке, в которой он находится.

Направление поворота влево/вправо определяются, если смотреть на робота извне куба.

При выполнении команды `forward` сначала определяется, на какую их четырёх сторон своей клетки сейчас смотрит робот. Затем робот перемещается в соседнюю по этой стороне клетку. Если при этом робот остаётся на той же грани куба, то его направление не изменяется. Если же он переезжает на другую грань куба, то вектор направления естественным образом поворачивается на 90 градусов вокруг ребра куба. При этом всегда действует простое правило возврата: если после выполнения команды `forward` повернуть робота на 180 градусов и снова выполнить команду `forward`, то робот вернётся обратно в предыдущую клетку.

Игрок не видит игрового поля: он вслепую говорит команды ведущему, а ведущий их выполняет. После выполнения каждой команды `forward` ведущий говорит игроку одно из трёх слов:

- `closer` — робот оказался ближе к кладу;
- `farther` — робот оказался дальше от клада;
- `same` — расстояние от робота до клада не изменилось.

Расстояние от робота до клада — это евклидово расстояние между центрами клеток, в которых они находятся, то есть длина соединяющего эти центры напрямую отрезка. Обратите внимание, что расстояние считается насквозь куба, а не вдоль его поверхности. Если эта величина уменьшилась, то ведущий говорит `closer`, а если увеличилась — то `farther`.

Вам предлагается написать программу, которая будет играть в эту игру за игрока. У вас нет права на ошибку: при выполнении команды `dig` робот должен обязательно стоять в клетке с кладом.

Протокол взаимодействия

Это интерактивная задача, и в ней вам предстоит работать не с файловым вводом-выводом, а со специальной программой — интерактором. Взаимодействие с ней осу-

ществляется через стандартные потоки ввода-вывода.

Ваша программа выводит на поток вывода команды управления роботом. Каждая команда задаётся словом `left`, `right`, `forward` или `dig` (см. описание команд выше). После каждой выведенной вашей программой команды `forward` на поток ввода приходит одно из слов: `closer`, `farther` или `same` (см. описание результатов выше).

После вывода команды `dig` игра заканчивается. Если при этом робот и клад находятся в разных клетках, то ваше решение получает вердикт Wrong Answer. Количество команд не должно превышать $100N$, иначе решение получит вердикт Wrong Answer.

Убедитесь, что вы выводите символ перевода строки и очищаете буфер потока вывода (команда `flush` языка) после каждой выведенной команды. Иначе решение может получить вердикт Timeout.

Пример

stdin	stdout
farther closer closer farther closer	forward left left forward forward forward right right forward dig

Задача 3. Зеркала

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	7 секунд
Ограничение по памяти:	512 мегабайт

Сбылась мечта идиота! Отныне ему не надо улепетывать от разъяренной вдовушки по коридору. Теперь мадам Грицацуева — главный персонаж игры «Отрази меня».

Согласно правилам этой игры, она движется по прямоугольному клеточному полю размера $N \times M$, постоянно отражаясь то от границ поля, то от двусторонних зеркал, которые либо вращаются, либо неподвижны. Каждое зеркало представляет собой отрезок единичной длины, центр которого находится в центре клетки. Сама мадам намного меньше и зеркала, и единичной клетки, поэтому её можно считать точкой, которая в любой целочисленный момент времени находится в центре какой-нибудь клетки.

За секунду мадам обязательно переходит в соседнюю клетку, если только она не отражается от границ поля. При этом, если она попадает в клетку, где находится зеркало, то она меняет направление движения по правилу «угол падения равен углу отражения». Более точно, если зеркало было наклонено под 45 градусов к её траектории, то мадам отразится на 90 градусов, а если зеркало перегораживало ей путь, то есть было перпендикулярно её движению, то она поменяет направление на 180 градусов. Если же зеркало не мешало её движению, то есть было параллельно направлению движения, то оно никак не помешает дальнейшему движению мадам. Если же вдовушка попадает в граничную клетку поля, то направление ее движения меняется на противоположное. При этом в этой граничной клетке мадам находится два соседних момента времени: можно считать, что полсекунды вдова движется от центра клетки до границы, там моментально отражается и ещё полсекунды возвращается обратно.

Мадам Грицацуева может стартовать в любой неотрицательный момент времени в произвольном направлении в одной из допустимых для старта точек. Её цель — как можно раньше оказаться в заветной точке с координатами (r_0, c_0).

Формат входных данных

В первой строке входного файла записано шесть целых чисел N, M, K, S, r_0 и c_0, где N, M — размеры поля по вертикали и горизонтали ($1 \leqslant N, M \leqslant 10^9$), K — количество возможных мест старта ($1 \leqslant K \leqslant 10^3$), S — количество зеркал ($0 \leqslant S \leqslant 10^3$), r_0, c_0 — номер строки и столбца, в которой находится точка выхода ($1 \leqslant r_0 \leqslant N$, $1 \leqslant c_0 \leqslant M$).

Далее в K строках приводятся координаты допустимых стартовых точек — в каждой строке по два целых числа r_j и c_j ($1 \leqslant r_j \leqslant N$, $1 \leqslant c_j \leqslant M$).

В остальных S строках приводятся описания зеркал: по пять целых чисел r_i, c_i, a_i, t_i, p_i в каждой. Здесь r_i, c_i — номера строки и столбца клетки, в которой расположено i-ое зеркало ($1 \leqslant r_i \leqslant N$, $1 \leqslant c_i \leqslant M$).

Число a_i описывает положение зеркала в начальный момент времени ($0 \leqslant a_i \leqslant 3$). Если $a_i = 0$, то зеркало расположено вертикально, а с каждым увеличением значения a_i на 1 оно поворачивается на 45 градусов по часовой стрелке. Таким образом, если $a_i = 1$, то отрезок зеркала направлен из левого нижнего угла в правый верхний, если $a_i = 2$, то оно горизонтально и т.д.

Значения t_i и p_i задают соответственно время первого поворота зеркала на 45 градусов по часовой стрелке и период, за который оно совершает каждый последующий такой поворот ($1 \leqslant p_i \leqslant 7$, $1 \leqslant t_i \leqslant p_i$). Таким образом, зеркало поворачивается в моменты времени: t_i —

первый поворот, $t_i + p_i$ — второй поворот, $t_i + 2p_i$ — третий поворот и т.д. Заметим, что в моменты времени t_i, (t_i+p_i) зеркало уже находится в новом положении, а в моменты времени $(t_i - 1)$, $(t_i + p_i - 1)$ — ещё в предыдущем. Если же зеркало неподвижно, то два последних числа t_i и p_i в описании равны -1.

Гарантируется, что все позиции стартовых точек, зеркал и точки выхода попарно различны, в том числе, что ни одно зеркало не находится в стартовой точке. Ориентация поля и нумерация строк и столбцов таковы, что при движении вниз увеличивается номер строки, а при движении вправо увеличивается номер столбца.

Формат выходных данных

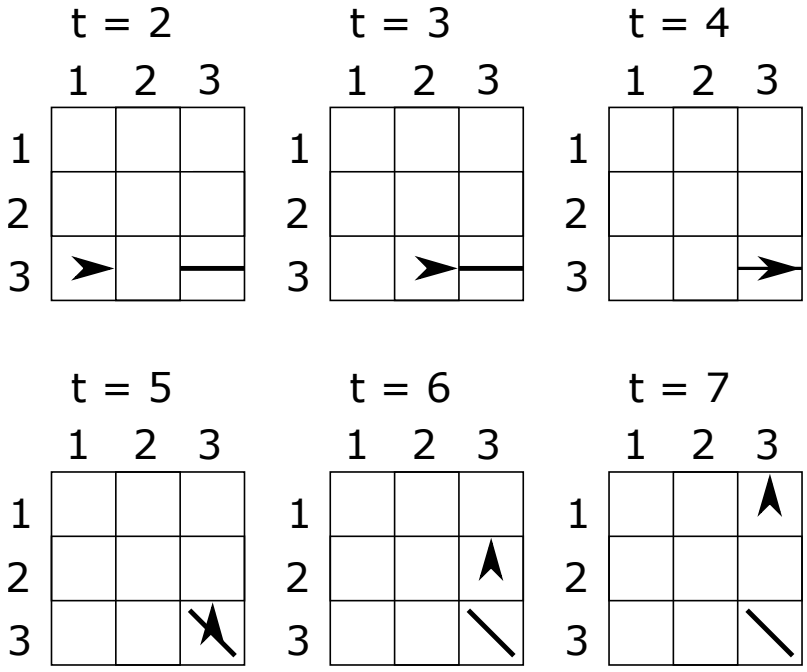
В выходной файл требуется вывести одно целое число: самое раннее время, когда мадам может оказаться в точке r_0, c_0, или -1, если это невозможно.

Пример

input.txt	output.txt
3 3 1 1 1 3 3 1 3 3 2 5 7	7
3 3 1 2 3 2 1 1 1 3 2 5 7 3 3 2 -1 -1	-1

Пояснение к примеру

На рисунке ниже изображён первый тест из условия. Стрелка соответствует положению и направлению движения мадам, отрезок в клетке $(3, 3)$ обозначает зеркало. Мадам стартует в момент времени $t = 2$ вправо. В момент времени $t = 4$ она пролетает мимо зеркала и в момент времени $t = 4.5$ отражается от стены, после чего отражается от повернувшегося зеркала в момент времени $t = 5$. В момент времени $t = 7$ мадам достигает точки выхода.



Задача 4. Дороги к синематографу

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

После фиаско в Санта-Каролине, мистер Фёст и его друзья открыли новый храм синематографа. Новое место популяризации синематографа выбрали неподалёку от большой железной дороги. Чёрный Джек предложил построить дороги до близких станций, на которых останавливаются пассажирские поезда.

Мистер Фёст нарисовал на плане систему координат. Началом системы координат выбрано то место, в котором расположились миссионеры синематографа, а оси X и Y направлены на восток и север соответственно. Станции отмечены на плане точками с координатами (x_i, y_i) при $i = 1 \ldots n$.

Железная дорога идёт в направлении на юго-восток, поэтому выполняются неравенства:

$$\begin{cases} 0 \leqslant x_1 \leqslant x_2 \leqslant x_3 \leqslant \ldots \leqslant x_{n-1} \leqslant x_n \\ 0 \leqslant y_n \leqslant y_{n-1} \leqslant y_{n-2} \leqslant \ldots \leqslant y_2 \leqslant y_1 \end{cases}$$

Диана настаивает на том, чтобы каждая дорога шла в точности в какую-нибудь сторону света, то есть чтобы соответствующий ей отрезок на плане был параллелен какой-нибудь оси координат. Джек же говорит, что дороги нужно построить таким образом, чтобы можно было доехать по дорогам от каждой станции до синематографа, не используя железную дорогу и проехав при этом минимально возможное расстояние. Разумеется, расстояние должно быть минимальным при условии параллельности дорог осям координат.

Дороги стоят денег, поэтому мистер Фёст хочет, чтобы суммарная длина дорог была минимальна при условии выполнения требований Дианы и Джека. А сэкономленные деньги можно потратить на покупку новых кинолент.

Формат входных данных

В первой строке входного файла задано целое число n — количество станций ($1 \leqslant n \leqslant 500$). В каждой из оставшихся n строк записано по два целых числа x_i, y_i — координаты одной из станций ($0 \leqslant x_i, y_i \leqslant 10^6$).

Гарантируется, что последовательность (x_i) нестрого возрастает, а последовательность (y_i) — нестрого убывает. Никакие две станции не находятся в одной точке. Никакая станция не расположена в начале координат.

Формат выходных данных

В первую строку выходного файла нужно вывести два целых числа: k — сколько дорог предлагается построить ($1 \leqslant k \leqslant 2\,000$) и A — суммарную длину всех этих дорог. В каждую из следующих k строк требуется выдать описание одной дороги.

Описание дороги должно содержать четыре целых числа x_1, y_1, x_2, y_2 — координаты концевых точек дороги ($x_1 \geqslant x_2$, $y_1 \geqslant y_2$). При этом должно быть верно ровно одно из двух утверждений: либо $x_1 = x_2$, либо $y_1 = y_2$.

Гарантируется, что существует оптимальный план, удовлетворяющий ограничениям формата выходных данных. Если возможно несколько вариантов ответа, выведите любой из них.

Пример

input.txt	output.txt
3 0 9 2 4 5 1	4 16 0 9 0 4 2 4 0 4 5 1 0 1 0 4 0 0

Задача 5. Геометрический решатель

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

Мальчик Кедас участвует в разработке нового геометрического решателя. Ему поручили реализовать устранение избыточных ограничений на двумерном чертеже. Помогите Кедасу справиться с задачей.

В рамках данной задачи будем полагать, что на чертеже есть только N отрезков $L_1, L_2, \ldots, L_n$ и набор связывающих их ограничений. Каждое *ограничение* — это утверждение об объектах чертежа. Ограничения бывают четырёх типов:

- `hor` i — отрезок L_i горизонтальный (Y-координаты его концов совпадают);
- `vert` i — отрезок L_i вертикальный (X-координаты его концов совпадают);
- `par` i j — отрезки L_i и L_j параллельны;
- `perp` i j — отрезки L_i и L_j перпендикулярны.

Решением чертежа называется размещение всех $2N$ концов всех отрезков чертежа на координатной плоскости, при котором:

1. все ограничения чертежа удовлетворены, т.е. соответствующие утверждения верны;
2. каждый отрезок имеет ненулевую длину, то есть его концевые точки **не** совпадают.

Считается, что направление отрезка значения не имеет.

Избыточные ограничения могут плохо повлиять на работу численных алгоритмов геометрического решателя, поэтому их по возможности стараются удалить. Подмножество ограничений чертежа называется *избыточным*, если при его удалении множество решений чертежа не изменяется. Если множество решений чертежа пусто, то чертёж называется *несовместным*, а в противном случае — *совместным*.

В данной задаче требуется определить, является ли заданный чертёж совместным, и если он является, то также найти на нём избыточное подмножество ограничений максимального размера.

Формат входных данных

В первой строке входного файла записано два целых числа n и m, где n — количество отрезков на чертеже ($1 \leqslant n \leqslant 10^5$), а m — количество ограничений на чертеже ($0 \leqslant m \leqslant 10^5$). В каждой из следующих m строк описывается одно ограничение.

Каждое ограничение описывается так, как показано выше в условии: сначала записано слово (`hor`, `vert`, `par` или `perp`), определяющее тип, потом через пробел записан номер i первого отрезка-аргумента. Если тип равен `par` или `perp`, то ещё через пробел указан номер j второго отрезка-аргумента. При этом $1 \leqslant i \neq j \leqslant n$.

Отрезки пронумерованы числами от 1 до n, ограничения пронумерованы числами от 1 до m в порядке описания. Ограничения могут иметь полностью одинаковые описания.

Формат выходных данных

В первую строку требуется вывести слово `consistent`, если чертёж совместный, и `inconsistent`, если чертёж несовместный. Если чёртеж несовместный, то больше ничего выводить **не** нужно.

Если чертёж совместный, то во вторую строку нужно вывести целое число k — количество ограничений в найденном избыточном множестве ($0 \leqslant k \leqslant m$). В третью строку нужно вывести k различных целых чисел — номера ограничений в этом множестве в любом порядке.

Если искомых ответов несколько, разрешается вывести любой из них.

Примеры

input.txt	output.txt
2 4 vert 1 hor 2 vert 1 perp 2 1	consistent 2 4 1
2 4 vert 1 hor 2 vert 1 par 2 1	inconsistent

Пояснение к примеру

В первом примере на отрезок L_1 наложено две вертикальности, очевидно одну из них можно убрать, и множество решений не изменится. Перпендикулярность отрезков L_1 и L_2 тоже лишняя, так как один отрезок вертикальный, а другой горизонтальный. Если удалить три ограничения, то всегда можно привести решение, на котором один из отрезков **не** параллелен осям координат, то есть множество решений чертежа увеличится.

Во втором примере чертёж несовместный, поскольку ограничение 4 говорит, что отрезки L_1 и L_2 параллельны, тогда как из ограничений 1 и 2 следует, что они перпендикулярны.

Задача 10. Билеты

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

Киса и Ося уселись на добытых непосильным трудом стульях и играют в следующую занимательную игру. Каждый держит в руках трамвайный билет с шестизначным номером от 000000 до 999999. Они по очереди называют цифры слева направо на своем билете, и после каждой пары цифр проигравший, тот кто назвал меньшую цифру, выполняет желание победителя.

Определите, пожалуйста, кто сколько желаний исполнит.

Формат входных данных

В первой строке входного файла записан номер билета Оси, а во второй — номер билета Кисы. Каждый номер состоит из шести цифр в диапазоне от `0` до `9`. В номере могут быть ведущие нули.

Формат выходных данных

В первую строку выходного файла нужно вывести число желаний, которые исполнит Ося, а во вторую — сколько желаний исполнит Киса.

Пример

input.txt	output.txt
013936 132946	3 1

Пояснение к примеру

Ося проигрывает на первой, второй и пятой цифрах, а Киса — на третьей цифре.

Задача 11. Сглаживание логарифма

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

Жизненная траектория Васисуалия Лоханкина вся прямо-таки изогнутая, ну вылитый логарифм (правда, умноженный на некоторое число). На досуге, размышляя о трагической судьбе русской интеллигенции и о великой сермяжной правде, он решил эту траекторию исследовать повнимательнее. А именно, он решил ее заменить на более понятную широким массам ломаную линию. При этом он, естественно, хочет, чтобы, с одной стороны, выбранная им ломаная как можно меньше отличалась от исходного логарифма, а с другой, количество звеньев этой ломаной, не превышало заданного числа n.

Рассмотрим проблему более формально. Будем говорить, что определённая на отрезке $[a, b]$ функция является ломаной с не более чем n звеньями, если её график можно разбить на n частей, каждая из которых является отрезком прямой. Обозначим класс непрерывных ломаных, определённых на заданном отрезке $[a, b]$ и имеющих не более n звеньев, через $B_n[a, b]$. Расстоянием между двумя функциями $f(x)$ и $g(x)$ на отрезке $[a, b]$ назовём число:

$$D(f, g) = \max_{x \in [a,b]} |f(x) - g(x)|$$

Требуется найти ломаную из класса $B_n[a, b]$, расстояние от которой до функции $f(x) = c \ln x$ минимально.

Формат входных данных

В первой строке входного файла задано одно целое число m — количество тестовых примеров ($1 \leqslant m \leqslant 20$).

Далее следует m строк, в каждой из которых записано по четыре числа n, c, a и b, где n — количество звеньев ломаной ($1 \leqslant n \leqslant 20$), c — множитель при логарифме ($1 \leqslant c \leqslant 10^4$), a, b — левая и правая границы рассматриваемого отрезка соответственно ($10^{-2} \leqslant a < b \leqslant 10^2$). Числа n и c — целые, а числа a, b — вещественные, заданные не более чем с двумя знаками после десятичной точки.

Формат выходных данных

В выходной файл необходимо вывести m строк, в каждой из которых должно быть записано одно вещественное число — минимально возможное максимальное отклонение ломаной с не более чем n звеньями от логарифма, ответ на соответствующий тестовый пример. Абсолютная или относительная погрешность вашего ответа не должна превышать 10^{-8}.

Пример

input.txt	output.txt
2 1 1 1 10 1 1 0.5 7	0.309517460 0.398765993

Задача 12. Космические сигналы

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

Группа космических исследований по-прежнему занимается поисками внеземного разума. Некоторое время назад из космоса было принято два сигнала, предположительно отличающиеся от простого шума. Этим сигналам присвоили имена X и Y. Оба сигнала закодировали в виде строк, состоящих из маленьких латинских букв.

Недавно космических исследователей вновь постигла удача — был принят ещё один сигнал. Этому сигналу дали имя Z и также закодировали в виде строки из маленьких латинских букв. Теперь стоит следующая задача: требуется выяснить, является ли Z принципиально новым сигналом или он — всего лишь комбинация уже известных сигналов X и Y.

Считается, что сигнал Z является комбинацией сигналов X и Y в том и только в том случае, если он содержит непересекающиеся вхождения сигналов X и Y. Иными словами, если можно указать в Z две неперекрывающихся подстроки, одна из который равна X, а другая равна Y (если X является подстрокой Y или Y является подстрокой X, подстроки считаются перекрывающимися).

Формат входных данных

В первой строке входного файла содержится количество тестовых наборов данных (от 1 до 1000). Далее следуют тестовые наборы в указанном количестве, по три строки каждый. В первой строке тестового набора содержится представление сигнала Z, в следующих двух строках записаны представления сигналов X и Y. Каждая из этих строк не пуста.

Суммарная длина всех строк во всех тестовых наборах не превосходит 10^6.

Формат выходных данных

Для каждого тестового набора в выходной файл нужно вывести ответ в отдельную строку, «`YES`» (без кавычек), если сигнал Z является комбинацией сигналов X и Y, и «`NO`» (без кавычек) — в противном случае.

Примеры

input.txt	output.txt
3 abacada aba ada abacada aba aca abacada aba aaa	YES NO NO

Замечание

Во втором тестовом наборе обе строки X и Y присутствуют в строке Z как подстроки. Однако эти подстроки перекрываются: третий символ строки Z принадлежит им обеим.

Задача 13. Weather Balloon

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Ковбой Джо в течение длительного времени наблюдал за направлением ветра в прериях. В результате этих наблюдений он построил формулу, описывающую поведение метеорологического зонда. В частности, формула предсказывет высоту a (в метрах выше уровня земли) в момент t после запуска зонда.

$$A = -6t^4 + ht^3 + 2t^2 + t$$

При этом h — целое число между 1 и 100 включительно, зависящее от влажности воздуха.

Джо хочет дождаться момента, когда после запуска зонд коснётся земли. При этом, если зонд коснётся земли позже, чем через M часов, то считается, что Джо не удалось дождаться соответствующего момента. Примем, что зонд коснулся земли в случае, если $A \leqslant 0$.

Формат входных данных

Во входном файле заданы два неотрицательных целых числа: h — влажность воздуха, и M — максимальное время, в течение которого Джо готов ждать ($0 \leqslant h \leqslant 100$ and $0 < M < 240$).

Формат выходных данных

В случае, если зонд в первый раз коснётся земли позже, чем через N часов, выведите "`The balloon does not touch ground in the given time.`", в противном случае в первой строке выходного файла выведите "`The balloon first touches ground at hour:`", а во второй — целое положительное число T — наиболее раннее время, в которое высота зонда не будет превосходить 0.

Примеры

input.txt	output.txt
30 10	The balloon first touches ground at hour: 6
70 10	The balloon does not touch ground in the given time.

Задача 14. Blood

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

На станции переливания крови имеется некоторое количество крови всех четырёх групп: первой, второй, третьей и четвёртой. Также для крови определён "резус-фактор", который может быть как положительным, так и отрицательным.

Своей очереди на переливание крови ждёт некоторое количестов пациентов. При этом каждому пациенту требуется одна объёмная единица крови. Совместимость крови донора и пациента определяется следующими правилами:

- Пациентам с первой группой крови может быть перелита только кровь первой группы.
- Пациентам со второй группой крови может быть перелита кровь первой или второй группы.
- Пациентам с третьей группой крови может быть перелита кровь первой или третьей группы.
- Пациентам с четвёртой группой крови может быть перелита кровь любой группы.

Кроме того, пациентам с отрицательным резус-фактором может быть перелита только кровь с отрицательным резус-фактором. Пациентам с положительным резус-фактором может быть перелита кровь как с положительным, так и с отрицательным резус-фактором. Выясните, какому наибольшему количеству пациентов удастся сделать успешное переливание крови.

Формат входных данных

В первой строке входного файла содержатся 8 целых чисел — количество объёмных единиц крови первой группы с отрицательным резус-фактором, первой группы с положительным резус-фактором, второй группы с отрицательным резус-фактором, второй группы с положительным резус-фактором, третьей группы с отрицательным резус-фактором, третьей группы с положительным резус-фактором, четвёртой группы с отрицательным резус-фактором, четвёртой группы с положительным резус-фактором соответственно. Во второй в аналогичном формате задано количество пациентов с соответствующим сочетанием группы крови и резус-фактора. Все числа во входном файле неотрицательны и не превышают 10^7.

Формат выходных данных

Выведите одно целое число — наибольшее количество пациентов, которым удастся сделать успешное переливание крови.

Пример

input.txt	output.txt
5 5 3 1 2 11 5 12 2 4 9 2 3 9 7 3	33

Задача 15. Card Game

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Рассмотрим карточную игру для двух игроков. Карты в этой колоде делятся по четырём мастям (красная, жёлтая, зелёная или чёрная) и значениям (1, 2, 3, 4, 5). Всего в колоде содержатся 20 карт — каждая комбинация масти и значения представлена ровно один раз.

Каждому игроку раздаётся по 10 карт. Игра состоит из десяти раундов. Задача каждого игрока — выиграть как можно больше раундов. В каждом раунде определяется 'атакующий' игрок, который делает ход одной из карт, имеющихся у него на руках. Другой игрок должен сыграть картой той же масти, если таковая у него есть. Если нет, он сбрасывает любую карту, после чего обе карты выходят из игры на данный раунд. Атакующий выигрывает, если второй игрок не нашёл карты той же масти, или же если значение карты атакующего больше значения карты второго игрока. В противном случае выигрывает другой игрок.

Право атаки в первом раунде определяется жребием, в последующих девяти раундах атакует игрок, выигравший предыдущий раунд.

Выясните, сколько раундов выиграет игрок, атакующий первым, при том, что оба игрока придерживаются оптимальной стратегии.

Формат входных данных

Входной файл состоит из не более, чем 10 тестовых примеров. Каждый тестовый пример состоит из одной строки ввода, описывающей карты, которые сданы игроку, атакующему в первом раунде. Карты описываются парой «буква-цифра», где буква — одна из букв 'R','Y','G','B' соответственно для красной, жёлтой, зелёной и чёрной мастей, а цифра — одна из цифр (1,2,3,4,5), задающая значение карты. Остальные 10 карт розданы другому игроку.

После последнего тестового примера следует строка, содержащая 10 звёздочек, разделённых пробелами. Эту строку обрабатывать не следует.

Формат выходных данных

Для каждого тестового примера в отдельной строке выведите целое число — количество раундов, выигранных при оптимальной стратегии игроком, атакующим первым.

Пример

input.txt	output.txt
G1 G3 B2 R2 Y1 R3 R5 Y2 Y3 G5 * * * * * * * * * *	3

Задача 16. Multiple choice tests

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Школьный учитель Е.Г. Обломский часто даёт тесты, в которых требуется выбрать один правильный вариант из пяти. Е.Г. считает, что, во-первых, даже у не знающего предмет школьника остаётся интерес к тестам как к упражнениям на угадку, а, во-вторых, проверку таких тестов легко автоматизировать.

Ваша задача — написать систему автоматической проверки подобных тестов. Возможные ответы обозначаются буквами ‘`A`’,‘`B`’,‘`C`’,‘`D`’ or ‘`E`’.

Формат входных данных

В первой строке входного файла содержится целое число N ($0 < N < 10^4$) — количество заданий в тесте. В последующих N строках содержатся ответы ученика на соответствующие вопросы теста. Каждый ответ представляет собой один из символов ‘`A`’, ‘`B`’, ‘`C`’,‘`D`’ или ‘`E`’. Далее в аналогичном формате в N строках заданы правильные ответы, при этом вопросы перечислены в порядке их следования в тесте, то есть строка с номером i содержит ответ ученика, а строка с номером $N + i$ — правильный ответ на один и тот же вопрос.

Формат выходных данных

Выведите одно целое число — количество правильных ответов, данных учеником.

Примеры

input.txt	output.txt
3 A B C A C B	1
3 A A A A B A	2