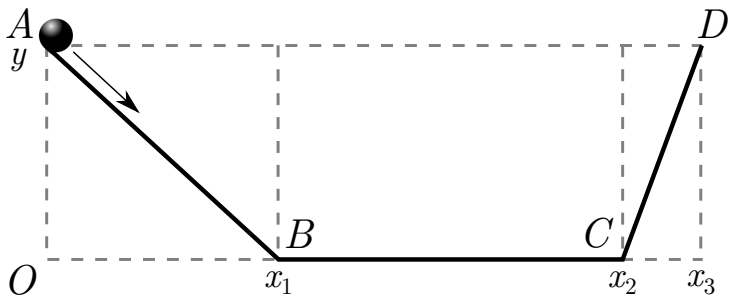
Задача A. Путь ядра

Имя входного файла:	`canonball.in`
Имя выходного файла:	`canonball.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Продольный разрез прямого жёлоба имеет вид, показанный на рисунке.



Точки A, B, C и D имеют координаты соответственно $(0,\ y)$, $(x_1,\ 0)$, $(x_2,\ 0)$, $(x_3,\ y)$.

В нулевой момент времени пушечное ядро находится в точке A и начинает скатываться в жёлоб. По участку AB ядро движется с ускорением a_1, по участку BC — с замедлением a_2, по участку CD — с замедлением a_3.

Если ядро останавливается на горизонтальном участке жёлоба (включая точки B и C), дальше оно не двигается. Если ядро останавливается на подъёме (не в верхней точке этого подъёма), оно сразу же начинает скатываться в обратную строну. Ускорение на спуске DC численно равно a_3, а замедление на подъёме BA — a_1. Наконец, если ядро достигает точек A или D, оно вылетает из жёлоба.

Рассчитайте время от начала скатывания ядра до его полной остановки или до вылета из жёлоба.

Формат входных данных

Первая строка входных данных содержит величины x_1, x_2, x_3, y (целые положительные числа, не превосходящие 1000, $x_1 < x_2 < x_3$).

Во второй строке записаны целые величины a_1, a_2, a_3 ($0 < a_1, a_3 \leqslant 100$, $0 \leqslant a_2 \leqslant 100$, $a_1 > a_2$, $a_3 > a_2$).

Формат выходных данных

Выведите единственное число — искомое значение времени, рассчитанное с относительной или абсолютной погрешностью не более 10^{-6}.

Примеры

canonball.in	canonball.out
10 50 70 20 5 3 5	7.975193500
10 50 70 20 5 0 5	17.312712690
20 60 70 20 25 5 7	3.422848811

Задача B. Ожерелье из монет

Имя входного файла:	`coins.in`
Имя выходного файла:	`coins.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Петя нашёл баночку с утерянными кем-то советскими 15-копеечными монетами. Количество монет в баночке равно N. Теперь это — его новая игрушка!

Однажды Петя решил сложить из имеющихся монет ожерелье так, чтобы каждая монета касалась только двух соседних, а центры монет образовывали правильный N-угольник. И это ему удалось!

Но, когда Петя похвастался перед отцом своими успехами, тот попросил мальчика найти максимальное расстояние между точками, находящимися на поверхности монет (в том числе и на их краю). Увы, Петя не знает решения этой задачи... Помогите ему!

Формат входных данных

В первой строке входных данных записано число N ($3 \leqslant N \leqslant 10\,000$), а во второй — диаметр одной монеты (целое положительное число, не превосходящее 1000).

Формат выходных данных

Выведите единственное число — ответ на задачу. Ответ будет принят, если относительная или абсолютная погрешность не будет превышать 10^{-9}.

Примеры

coins.in	coins.out
3 6	12.0000000000
7 1	3.2469796037

Задача C. Набор номера

Имя входного файла:	`dial.in`
Имя выходного файла:	`dial.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В крупных международных IT-компаниях для связи офисов, расположенных в разных городах, применяются системы видеотелефонии. В офисах организуют переговорные комнаты, оборудованные телевизорами и видеокамерами, при этом за каждой комнатой закреплён некоторый телефонный номер. Система видеотелефонии позволяет установить связь между двумя абонентами или организовать конференцию, в которой могут участвовать несколько абонентов. Если не вдаваться в подробности, звонок по видеотелефону с точки зрения пользователя осуществляется так же, как по обычному телефону: нужно набрать номер телефона вызываемого абонента на пульте дистанционного управления и нажать кнопку вызова. Трудность заключается в том, что на «эргономичных» пультах производства одной широко известной в узких кругах фирмы нет цифровой клавиатуры, а есть только клавиши управления курсором и бесполезная кнопка выключения микрофона.

Номер телефона приходится набирать нетривиальным способом. Изначально на экране телевизора отображается ряд из тринадцати символов: цифры от `0` до `5`, точка `.`, цифры от `6` до `9`, звёздочка `*` и решётка `#`. Курсор установлен посередине — на символе точки.

0	1	2	3	4	5	.	6	7	8	9	*	#

Для набора номера можно использовать три кнопки пульта дистанционного управления. Нажатием кнопок «Влево» и «Вправо» можно сдвигать курсор на одну позицию в горизонтальном направлении. Последовательность символов зациклена, то есть левее самой левой клетки следует самая правая и правее крайней правой клетки идёт самая левая. При нажатии кнопки «Ввод» текущий символ, на который указывает курсор, добавляется к номеру.

Вам необходимо определить, за какое минимальное число нажатий клавиш пульта дистанционного управления можно набрать заданный номер.

Формат входных данных

В первой строке содержится число T — количество тестовых примеров ($2 \leqslant T \leqslant 10$). В последующих T строках записаны непустые строки — номера, которые нужно набрать на видеотелефоне, по одному в строке. Длина каждого номера не превосходит 3000, а в его состав входят только символы, показанные в таблице, приведённой выше. В начале набора каждого номера курсор находится в исходном положении.

Формат выходных данных

Выведите T строк — минимальное количество нажатий на кнопки пульта дистанционного управления, необходимое для набора каждого номера.

Примеры

dial.in	dial.out
3 1 123 1808	6 10 22

Задача D. Делимость Фибоначчи

Имя входного файла:	`fibonacci.in`
Имя выходного файла:	`fibonacci.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Рассмотрим последовательность Фибоначчи

$$F_0 = 0,$$

$$F_1 = 1,$$

$$F_{n+1} = F_n + F_{n-1}, \quad n \geqslant 1,$$

которая имеет вид

$$0,\ 1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34,\ 55,\ 89,\ 144,\ \ldots$$

Сколько членов этой последовательности из K первых делятся на M?

Формат входных данных

Во входных данных задаются два целых числа K ($1 \leqslant K \leqslant 10^{18}$) и M ($1 \leqslant M \leqslant 10^6$).

Формат выходных данных

Выведите одно число — ответ на задачу.

Примеры

fibonacci.in	fibonacci.out
10 2	4

Задача E. Делимость Фибоначчи (сложная версия)

Имя входного файла:	`fibonacci2.in`
Имя выходного файла:	`fibonacci2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Рассмотрим последовательность Фибоначчи

$$F_0 = 0,$$

$$F_1 = 1,$$

$$F_{n+1} = F_n + F_{n-1}, \quad n \geqslant 1,$$

которая имеет вид

$$0,\ 1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34,\ 55,\ 89,\ 144,\ \ldots$$

Сколько членов этой последовательности из K первых делятся на M?

Формат входных данных

В первой строке входных данных задается число тестовых примеров T ($1 \leqslant T \leqslant 100$). Далее в каждой из T строк содержится по два целых числа K ($1 \leqslant K \leqslant 10^{18}$) и M ($1 \leqslant M \leqslant 10^9$).

Формат выходных данных

Выведите T чисел — количество членов последовательности из K первых, которые делятся на M. Числа следует выводить по одному в строку.

Примеры

fibonacci2.in	fibonacci2.out
2 100 2 1000000000000000000 987654321	34 172125453293

Задача F. Хэллоуин

Имя входного файла:	`halloween.in`
Имя выходного файла:	`halloween.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Через пару недель Васю ждут жаркие выходные, потому что приближается Хэллоуин! Вася хочет посетить как можно больше праздничных костюмированных вечеринок. Вечеринки, как правило, тематические, и нужно приходить на них в соответствующем облачении. Например, направляясь на вечеринку фанатов комиксов, Вася выберет костюм Супермена, а на закрытую вечеринку своих коллег по цеху, спортивных программистов, Вася придёт в костюме бродячего торговца — коммивояжёра.

Поскольку Вася за ночь с 31 октября на 1 ноября хочет успеть сходить на несколько вечеринок и при этом быть в соответствующих костюмах, ему придётся переодеваться несколько раз. Чтобы было проще, Вася может надеть сразу несколько костюмов один на другой (например, костюм коммивояжёра поверх костюма Супермена). Перед каждой вечеринкой Вася может снять с себя несколько костюмов или надеть новый. Так, если форма коммивояжёра надета сверху костюма Супермена и Вася идёт на тусовку супергероев, он может либо снять костюм коммивояжёра, либо надеть наверх новый, ещё один, костюм Супермена. К сожалению, Вася не любит надевать ношеные вещи вновь без предварительной стирки, поэтому после снятия формы коммивояжёра Вася, если ему вновь понадобится такой вид, возьмёт новую форму. За раз Вася может снять любое число костюмов, но, естественно, если он снимает k костюмов, то это будут те k костюмов, которые надеты последними (если костюм A надет перед костюмом B, то, чтобы снять A, надо предварительно снять B).

Зная, какие вечеринки запланированы в ночь Хэллоуина, определите минимальное число костюмов, которые нужно приготовить Васе.

Формат входных данных

В первой строке записаны целые числа N ($1 \leqslant N \leqslant 100$) — количество вечеринок — и M ($1 \leqslant M \leqslant 100$) — количество различных типов костюмов. Во второй строке записано N целых чисел $c_1, \ldots, c_N$, где c_i ($1 \leqslant c_i \leqslant M$) — тип костюма, который подходит к вечеринке i ($1 \leqslant i \leqslant N$). Вася посещает вечеринки в порядке $1, 2, \ldots, N$.

Формат выходных данных

Выведите одно число — минимальное количество костюмов.

Примеры

halloween.in	halloween.out
1 1 1	1
2 2 1 1	1
3 2 1 2 1	2
7 3 1 2 1 1 3 2 1	4

Задача G. Ленивый поварёнок

Имя входного файла:	`juice.in`
Имя выходного файла:	`juice.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

После окончания поварских курсов Гриша устроился на работу в элитный ресторан. Но, естественно, на первых порах сложных заданий ему не поручают...

Вот и сегодня Гриша должен приготовить N видов свежевыжатых соков, которые заказали организаторы намеченного на вечер банкета. Для приготовления соков требуется K различных овощей и фруктов, причём в состав одного сока может входить от 1 до K компонентов. Объём чаши соковыжималки, которой будет пользоваться Гриша, а также количество исходных материалов достаточны для того, чтобы приготовить весь объём требуемого сока за один раз.

По технологии, чашу соковыжималки надо помыть после приготовления каждого сока. Но Грише очень не нравится эта операция... Он заметил, что если в состав следующего сока входят все компоненты, требуемые для изготовления предыдущего, то чашу можно и не мыть. Так, после приготовления яблочного сока можно обойтись без мытья чаши перед приготовлением яблочно-ананасового сока.

Помогите Грише и рассчитайте минимальное количество операций мытья чаши, которые потребуются для приготовления всех соков. Естественно, после выполнения работы чаша должна быть чистой!

Формат входных данных

В первой строке записываются целые величины N и K ($1 \leqslant N, K \leqslant 300$). Далее следуют N строк, каждая из которых описывает рецепт изготовления одного сока. Первое число этой строки m ($1 \leqslant m \leqslant K$) — количество компонентов сока. Далее следуют m различных целых чисел, каждое из которых находится в интервале от 1 до K включительно — номера компонентов, которые входят в состав соответствующего сока. Гарантируется, что набор компонентов различен для различных соков.

Формат выходных данных

Выведите единственное число — минимальное количество операций по мытью чаши.

Примеры

juice.in	juice.out
3 2 1 1 1 2 2 1 2	2
4 4 1 1 1 2 1 3 1 4	4

Задача H. Серии пенальти

Имя входного файла:	`penalties.in`
Имя выходного файла:	`penalties.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Групповые турниры являются одним из этапов крупнейших футбольных соревнований. Суть группового турнира заключается в следующем: несколько команд формируют группу, участники которой играют по принципу «каждый с каждым» в один или два круга. На следующий этап проходят команды, набравшие по итогам игр в группе наибольшее количество очков. Так, в Лиге Чемпионов формируется восемь групп по четыре команды в каждой. По результатам группового турнира по две лучших команды из каждой группы продолжают борьбу в раунде плей-офф этого турнира, а команды, занявшие третьи места в группах, играют в Лиге Европы.

Проблема заключается в том, что не всегда можно определить победителей в группе только исходя из набранных в групповом турнире очков. Конечно, есть и дополнительные показатели, которые можно применить (разность забитых и пропущенных мячей, голы, забитые на поле соперника, и т. д.), однако и они не всегда срабатывают. Что делать, например, если все матчи в группе закончились с одинаковым счётом 1:1?

Одно из возможных предложений для разрешения этой коллизии состоит в реализации последовательных серий пенальти до выявления победителя. Пусть нам необходимо отобрать из N команд группы, имеющих одинаковые результаты по итогам группового турнира, Q лучших. Серия пенальти состоит из $N \cdot (N-1)$ одиннадцатиметровых ударов, где каждая пара команд пробивает по одному удару в ворота друг друга. Рассмотрим случай, когда игрок команды A бьёт пенальти в ворота команды B. Если пенальти забит, одно очко получает команда A, в противном случае — команда B. После окончания серии все команды ранжируются по убыванию набранных в этой серии очков. Если команды, занявшие Q-е и $(Q+1)$-е места, набрали различное количество очков, коллизия считается разрешённой. В противном случае назначается новая серия для команд, набравших то же количество очков, что и команда, оказавшаяся на Q-м месте.

Авторы этого предложения нуждаются в его статистических исследованиях. В частности, они хотят узнать, насколько часто одной серии пенальти будет достаточно для определения команд, выходящих в следующий этап.

Помогите авторам, определив, достаточно ли будет проведения единственной серии по её результатам!

Формат входных данных

В первой строке записаны величины N и Q ($1 \leqslant Q < N \leqslant 100$). Далее следуют $N \cdot (N-1)$ строк с результатами выполнения одного одиннадцатиметрового удара. Каждая такая строка содержит три числа: номер команды, игрок которой бьёт пенальти, номер команды, в ворота которой выполняется удар, и результат удара (1 — пенальти забит, 0 — нет). Нумерация команд начинается с единицы.

Формат выходных данных

Если проведения одной серии достаточно, первая строка должна содержать слово `yes`. Во второй строке записываются Q чисел — номера команд, прошедших на следующий этап турнира (в порядке возрастания этих номеров).

Если потребуются дополнительные серии, первая строка должна содержать слово `no`. Во второй строке записываются величины N_1 и Q_1 — количество команд в последующей серии и количество команд, которые необходимо отобрать для следующего этапа соревнований. Наконец, в третьей строке указываются N_1 чисел — номера команд, для которых будет проведена новая серия пенальти (в порядке возрастания этих номеров).

Примеры

penalties.in	penalties.out
4 2 1 2 1 2 1 1 1 3 0 3 1 0 1 4 0 4 1 1 3 4 0 4 3 0 2 4 1 4 2 1 2 3 0 3 2 1	yes 3 4
4 2 1 2 1 2 1 1 1 3 0 3 1 0 1 4 0 4 1 1 3 4 0 4 3 0 2 4 1 4 2 1 2 3 1 3 2 1	no 2 1 2 3

Задача I. Авиабилеты

Имя входного файла:	`plane-tickets.in`
Имя выходного файла:	`plane-tickets.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Цены на авиабилеты существенно колеблются неделю за неделей, и их непредсказуемость является основным источником разочарования для путешественников. Когда по мере приближения даты вылета цены падают, пассажиры жалеют, что купили билеты слишком рано. Другие путешественники, наоборот, начинают переживать, что не приобрели билет заранее, когда цена вдруг подскакивает. В итоге все огорчаются, довольны только сами авиакомпании.

Наверняка есть какая-то причина для этого безумия. Получается, что авиакомпании устанавливают цены на билеты динамически, основываясь на том, сколько остаётся свободных мест и как скоро состоится вылет. Например, если с начала продажи билетов салон быстро заполнился, цена на оставшиеся места будет держаться на высоком уровне вплоть до момента, когда до рейса останется пара недель. Цена может упасть, чтобы все билеты оказались проданы. Так или иначе, авиакомпании хотят увеличить свои прибыли от каждого полёта.

Вы работаете в авиакомпании Байтландавиа, и ваша задача — установить цены на билеты по неделям. Авиакомпания собрала и проанализировала исторические данные, поэтому у вас есть надёжные оценки количества мест, которые будут проданы по определенной цене билета за определенное количество недель до полёта. Вам необходимо установить цену билета на каждую неделю, чтобы максимизировать выручку авиакомпании. Можно считать, что в каждую неделю при конкретной цене будет продано столько билетов, сколько прогнозируется, за исключением случая, когда свободных мест не хватает. В таком случае все оставшиеся места будут проданы.

Отметим, что повышение цен не обязательно означает, что будет продано меньше билетов. Более высокие цены могут иногда увеличить продажи, поскольку путешественники могут быть обеспокоены тем, что цены вскоре поднимутся ещё выше.

Формат входных данных

В первой строке записаны два целых числа — число оставшихся мест N ($0 < N \leqslant 300$) и число недель до вылета W ($0 \leqslant W \leqslant 52$). Следующие $W+1$ строк содержат оценки для W недель, $W-1$ недель, $\ldots$, одной недели и, наконец, недели перед вылетом. Каждая строка начинается с числа K ($1 \leqslant K \leqslant 100$) — количества различных цен для соответствующей недели. Далее следуют K чисел $0 < p_1 < p2 < \ldots < p_K < 1000$, задающих цены в долларах. Далее в строке записаны ещё K чисел s_1, s_2, $\ldots$, s_K (где $0 \leqslant s_i \leqslant N$), которые обозначают число билетов, продаваемых за определённую цену.

Формат выходных данных

В первой строке выведите максимальную общую выручку в долларах, которую может обеспечить авиакомпания от продаж начиная с текущей недели до момента вылета. Во второй строке выведите $W+1$ чисел — по какой цене нужно продавать в каждую из недель.

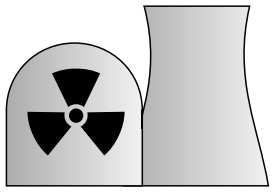
Примеры

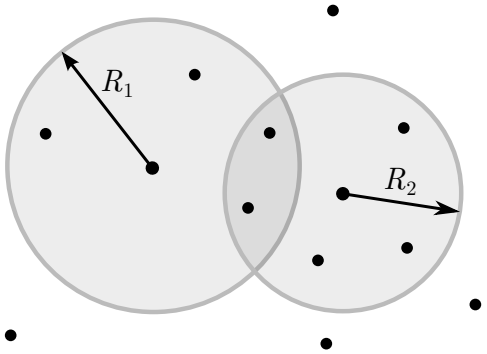
plane-tickets.in	plane-tickets.out
10 1 1 100 8 2 200 300 7 9	1400 100 300
50 2 1 437 47 3 357 803 830 13 45 46 1 611 14	23029 437 830 611

Задача J. Радиационная безопасность

Имя входного файла:	`radiation.in`
Имя выходного файла:	`radiation.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Атомные электростанции (АЭС) — это и благо, и беда современной цивилизации. С эксплуатацией АЭС связаны некоторые риски, но это по-прежнему один из самых дешёвых способов получения электроэнергии в развитых странах мира. В этой задаче мы будем рассматривать ситуацию в регионе, где две атомные электростанции находятся недалеко друг от друга.





Представим, что Земля плоская и на её поверхности задана двумерная прямоугольная система координат для определения положения точек. Пусть координаты двух атомных электростанций (a_x, a_y) и (b_x, b_y). В районе расположения АЭС экспертами определено N точек для возможного размещения автоматических пунктов радиационного контроля, которые будут осуществлять круглосуточный мониторинг радиационного фона. Оптимально было бы построить все N пунктов контроля, чтобы держать под наблюдением как можно большую площадь и оперативно реагировать на аварийный рост радиоактивности, но, к сожалению, бюджет проматомнадзора не резиновый... В соответствии с действующим законодательством вокруг первой АЭС устанавливается зона наблюдения радиуса R_1, поэтому руководство станции финансирует строительство всех пунктов контроля, лежащих на расстоянии не более R_1 от станции. Аналогично, радиус зоны наблюдения около второй АЭС равен R_2, и строительством пунктов контроля, попадающих в этот радиус, по закону должна заниматься администрация второй электростанции. Однако, поскольку станции расположены рядом, некоторые пункты контроля из N указанных могут попасть сразу в две зоны наблюдения. Естественно, нет смысла в этих пунктах устанавливать два одинаковых комплекта дозиметрического оборудования. Однако деньги уже выделены и должны быть освоены, поэтому решено построить дополнительно столько пунктов контроля вне зон наблюдения, сколько пунктов попало одновременно в обе зоны наблюдения (если такие пункты, конечно, есть среди N запланированных, иначе оставшиеся деньги придётся осваивать другим способом).

Определите, сколько пунктов радиационного контроля из запланированных N так и не будет построено при разных значениях радиусов зон наблюдения несмотря на рациональное использование бюджета.

Формат входных данных

В первой строке записано целое число N ($1 \leqslant N \leqslant 200\,000$) — количество спроектированных пунктов радиационного контроля. В последующих N строках записано по два целых числа x_i, y_i ($0 \leqslant x_i, y_i \leqslant 20\,000$) — координаты точки, где может быть расположен i-й пункта контроля. Гарантируется, что все точки различны. В следующей строке записано пять целых чисел: координаты двух АЭС a_x, a_y, b_x, b_y ($0 \leqslant a_x, a_y, b_x, b_y \leqslant 20\,000$) и число запросов Q ($1 \leqslant Q \leqslant 20\,000$). Точки расположения АЭС различны и не совпадают с точками установки пунктов контроля. Далее идут Q строк, в каждой записано по два целых числа R_1 и R_2, задающих радиусы зон наблюдения ($0 < R_1, R_2 \leqslant 13\,000$) вокруг первой и второй АЭС.

Формат выходных данных

Выведите Q строк. Для каждой пары радиусов зон наблюдения R_1, R_2 выведите одно число — количество пунктов дозиметрического контроля из N запланированных, которые так и не будут построены из-за недостатка финансирования.

Примеры

radiation.in	radiation.out
11 95 75 27 6 93 5 124 13 34 49 65 61 81 49 77 33 110 50 91 22 110 25 57 42 97 36 2 31 25 25 25	2 2

Задача K. Риск

Имя входного файла:	`risk.in`
Имя выходного файла:	`risk.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

«Риск» — настольная игра, местом действия в которой служит карта мира. Границы делят мир на регионы. Каждый регион находится под контролем игрока (это либо вы, либо один из ваших соперников). В каждом регионе, который вы контролируете, базируются ваши вооружённые силы: одна или несколько армий.

В течение хода вы можете изменять положение своих войск. Армия может либо остаться на месте, либо переместиться в смежный регион, также находящийся под вашим контролем. Движения выполняются одно за другим в любом порядке на ваше усмотрение. Однако в любой момент каждый регион должен содержать хотя бы одну армию.

Из стратегических соображений важно переместить свои армии в регионы, которые граничат с регионами соперников, и минимизировать число армий на внутренних регионах, которые окружены лишь регионами под вашим контролем. Ваша задача — добиться того, чтобы ваш слабейший рубеж, т. е. граничный регион с минимальным числом армий, был как можно более сильным. Какое наибольшее число армий можно разместить там?

Формат входных данных

В первой строке записано целое число N ($1 \leqslant N \leqslant 100$) — количество регионов в игре. Во второй строке записано N целых чисел $a_1, a_2, \ldots, a_N$ (где $0 \leqslant a_i \leqslant 100$), задающих число армий в каждом регионе. Если i-е число равно нулю, то регион i контролируется вашим соперником. Далее следуют N строк по N символов, каждый из которых `Y` или `N`. Если регион i граничит с регионом j, то на j-м месте i-й строки стоит символ `Y`, иначе `N`. Отношение смежности регионов симметричное. Также гарантируется, что i-й символ i-й строки всегда `N`. Гарантируется, что под вашим контролем находится хотя бы один регион, равно как и под контролем соперников есть хотя бы один регион. Более того, хотя бы один из ваших регионов граничит с регионом соперника.

Формат выходных данных

Выведите одно число — наибольшее число армий на вашем слабейшем рубеже.

Примеры

risk.in	risk.out
3 1 1 0 NYN YNY NYN	1
7 7 3 3 2 0 0 5 NYNNNNN YNYYNNN NYNYYNN NYYNYNN NNYYNNN NNNNNNY NNNNNYN	4

Задача L. Треугольники и четырёхугольник

Имя входного файла:	`triangles.in`
Имя выходного файла:	`triangles.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Маленькому Евклиду всего два года, а он уже любит геометрию. Сейчас он занят тем, что пытается из двух бумажных треугольников составить определённый бумажный четырёхугольник. Фигуры можно двигать, вращать в плоскости и переворачивать. Треугольники не должны накладываться друг на друга.

Формат входных данных

Входные данные содержат несколько отдельных тестов.

В первой строке записано целое число T — количество тестов ($1 \leqslant T \leqslant 500$). Каждый тест описывается десятью строками, каждая строка содержит координаты x и y некоторой точки на плоскости. Вначале три строки задают координаты вершин первого треугольника. Следующие три строки содержат координаты вершин второго треугольника. Наконец, следующие четыре строки задают координаты вершин четырёхугольника, заданные в порядке обхода по или против часовой стрелки.

Гарантируется, что все фигуры имеют положительную площадь, четырёхугольник не имеет самопересечений и самокасаний, все его вершины попарно различны. Координаты являются целыми числами и по модулю не превосходят 15 000.

Формат выходных данных

Выведите T строк. Для каждого теста выведите `Yes`, если из двух треугольников можно составить четырёхугольник, или `No` в противном случае.

Примеры

triangles.in	triangles.out
2 0 0 0 1 1 0 0 0 0 1 1 0 -1 0 0 0 1 0 0 1 0 0 -1 1 1 0 0 0 0 1 1 0 -1 0 0 0 1 0 0 1	Yes No