Внимание: Согласно правилам олимпиады, каждая команда может использовать только **один** компьютер. Вы можете использовать дополнительные компьютеры **только** для связи с сокомандниками и для чтения условий задач.

В любой момент времени может быть только один активный компьютер, на котором вы можете писать, компилировать, запускать программы. Активный компьютер может переключаться во время тура, например, чтобы следующую задачу писал другой член команды. Однако нельзя проявлять активность больше чем на одном компьютере одновременно.

Использовать интернет разрешается.

Использовать предварительно написанный код разрешается.

Задача 1. Слесарь-интеллигент и работа

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Виктор Михайлович Полесов, как и положено слесарю-интеллигенту, отличается умом и сообразительностью. Он недавно узнал, что работа силы есть не что иное, как скалярное произведение вектора силы на вектор перемещения. Естественно, он хочет чтобы эта самая работа была как можно больше, чтобы ничего не пропало. Осталось только выбрать, как и куда перемещаться.

Перемещаться можно из точки $(0, 0)$ в любую целочисленную точку круга $x^2 + y^2 \leqslant R^2$. Вектор силы известен — он везде постоянен и имеет координаты (a, b). Требуется найти так заинтересовавшую Полесова максимальную работу.

Формат входных данных

Во входном файле в первой строке записано одно целое число T — количество тестов ($1 \leqslant T \leqslant 1\,000$). Далее идут T строк, в каждой по три целых числа a, b — координаты вектора силы и R — радиус окружности ($-10^9 \leqslant a, b \leqslant 10^9$, $1 \leqslant R \leqslant 10^9$).

Формат выходных данных

Для каждого теста выведите одно целое число — искомую максимальную работу.

Примеры

input.txt	output.txt
3 10 -10 2 2 3 3 5 1 3	20 10 15

Пояснение к примеру

В первом тесте внутри круга с центром в начале координат радиуса 2 лежит 13 целочисленных точек. Для точек $(2, 0)$, $(1, -1)$, $(0, -2)$ скалярное произведение на вектор силы $(10, -10)$ максимальное. Например: $10 \times 1 + (-10) \times (-1) = 20$.

Задача 2. Спортивное ориентирование

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Ходислав решил поучаствовать в соревнованиях по спортивному ориентированию. Арена соревнований — бесконечная плоскость, препятствия отсутствуют и скорость передвижения одинакова в любом направлении. На арене расположены N контрольных пунктов (КП), которые каждый участник должен посетить в порядке возрастания их номеров.

Система отметки бесконтактная — в каждом КП установлена базовая станция, которая автоматически делает отметку, если участник находится на расстоянии не более R. Гарантируется, что зоны действия КП не пересекаются, но могут касаться.

Участник стартует в любой точке в зоне действия первого КП и финиширует в момент получения отметки на последнем КП. Разрешается заходить в зоны действия других КП по дороге к нужному, но отметка в этом случае не происходит.

Ходислав настроен на победу и планирует пройти дистанцию оптимальным образом. Помогите вычислить, какое расстояние ему придется преодолеть.

Формат входных данных

В первой строке входного файла записано два целых числа: N — количество контрольных пунктов ($2 \leqslant N \leqslant 100$) и R — радиус отметки ($1 \leqslant R \leqslant 10^9$).

Далее идет N строк с описанием КП в том порядке, в которым их нужно отметить. Каждая строка — это пара целых чисел x_i, y_i с координатами КП ($-10^9 \leqslant x_i, y_i \leqslant 10^9$).

Формат выходных данных

Выведите одно вещественное число — пройденное расстояние при оптимальном прохождении дистанции.

Относительная или абсолютная погрешность не должна превышать **0.01**. Это означает, что если оптимальный ответ равен X, то ваш ответ должен отличаться от X не более чем на $\frac{1}{100}\max(X, 1)$.

Примеры

input.txt	output.txt
3 3 0 -3 0 100 0 50	141.0

Пояснение к примеру

Ходислав стартует в точке $(0, 0)$, отмечает второй КП в точке $(0, 97)$, далее разворачивается и отмечает третий КП в точке $(0, 53)$. Суммарное пройденное расстояние $97 + 44 = 141$.

Задача 3. Кости

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

Ходислав играет в настольную ролевую игру. Он наконец-то выбрал оружие, чтобы совладать с монстром, и наносит решающий удар. Для этого он бросает игральные кости, подсчитывает сумму чисел на гранях и называет её ведущему.

Бросок группы одинаковых костей характеризуется тремя числами n, f и m, где n — количество костей, f — количество граней каждой кости, m — модификатор. На гранях написаны все числа от 1 до f, каждая грань может выпасть, броски костей независимы. Например, если $n = 3$, $f = 8$, $m = 5$, то для определения суммы следует бросить три восьмигранные кости, сложить результаты и добавить пять; обычно это записывается в виде $3d8 + 5$.

Ведущий хочет проверить, мог ли Ходислав набрать названную им сумму после броска игральных костей.

Формат входных данных

В первой строке входного файла записано единственное целое число B — количество ударов ($1 \leqslant B \leqslant 10^5$), нанесённых Ходиславом.

Далее в отдельных строках описаны удары. Первым следует целое число S — сумма, названная Ходиславом. Далее следует тройка целых чисел: n, f и m — описание группы игральных костей ($1 \leqslant S \leqslant 300$, $1 \leqslant n \leqslant 10$, $2 \leqslant f \leqslant 20$, $0 \leqslant m \leqslant 10$).

Формат выходных данных

Для каждого удара в порядке следования во входном файле выведите в отдельной строке `YES`, если сумма могла быть набрана, и `NO` иначе.

Примеры

input.txt	output.txt
5 3 1 6 0 1 1 8 1 16 1 12 3 1 2 4 0 42 3 20 1	YES NO NO NO YES

Задача 4. Системы счисления

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Ходислав узнал, что кроме десятичной системы счисления есть ещё, например, шестнадцатеричная. Его заинтересовало, как связаны значения в этих двух системах для одной и той же записи. Например, может ли у последовательности из K цифр шестнадцатеричное значение делиться нацело на десятичное значение? Изменится ли что-нибудь, если из десятичного значения вычесть заданное число D?

Запись — это последовательность цифр. В записи можно использовать цифры от 0 до 9: они есть и в десятичной, и в шестнадцатеричной системах. Запись числа не может начинаться с цифры 0. Десятичное значение записи — это число, которое получится, если интерпретировать запись как десятичное число. Шестнадцатеричное значение записи — это число, которое получится, если интерпретировать запись как шестнадцатеричное число.

Формат входных данных

В первой строке записано целое число T — количество тестов в файле ($1 \leqslant T \leqslant 100$). Далее записано T тестов, по одному в строке.

Для каждого теста записано два целых числа: K — количество цифр в записи и D — число, которое нужно вычесть из десятичного значения ($2 \leqslant K \leqslant 15$, $0 \leqslant D \leqslant 10^6$).

Формат выходных данных

Для каждого теста требуется найти все искомые записи. Это последовательности из K цифр без ведущих нулей, у которых значение в десятичной системе X, значение в шестнадцатеричной системе Y, при этом $X > D$ и Y делится нацело на $(X - D)$.

Ответ на тест выводится в одну строку. Сначала следует вывести количество найденных записей, а затем все найденные записи в порядке возрастания через пробел.

Пример

input.txt	output.txt
4 5 6 2 0 2 5 2 21	3 12510 24990 37950 0 2 16 22 4 22 24 34 96

Пояснение к примеру

Во первом тесте запись `37950` означает $X = 37950$ в десятеричной системе и $Y = 227664$ в шестнадцатеричной. Можно убедиться, что $Y = 227664$ делится нацело на $X - D = 37944$. Аналогично, для записи `24990` получаем, что $Y = 149904$ делится на $X - D = 24984$, а для записи `12510`: $Y = 75024$ делится на $X - D = 12504$,

Задача 5. Хоккей

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

Описанные в данном условии правила игры в хоккей несколько отличаются от общепринятых.

Хоккейный матч длится 60 минут, в течение которых две команды стараются забить друг другу как можно больше голов. Хоккейная команда состоит из пяти полевых игроков и одного вратаря.

Важной частью хоккея являются удаления. Во время игры полевому игроку может быть выписан штраф: в этом случае провинившийся игрок уходит с игровой площадки на определённое время, зависящее от вида штрафа. В результате, у его команды временно сокращается количество игроков на площадке. Штрафы в хоккее разделяют на два вида: *большой* и *малый*. При большом штрафе игрок удаляется на пять минут, а при малом — на две минуты. Когда время штрафа заканчивается, удалённый игрок возвращается на площадку.

Кроме того, малый штраф может быть завершён досрочно. Говорят, что команда играет в меньшинстве, когда у неё количество игроков на площадке меньше, чем у команды соперника. Если команда играет в меньшинстве, и ей забивают гол, то один из её игроков, удалённых **малым** штрафом, выходит на площадку, а его штраф завершается досрочно. Если у этой команды несколько игроков удалено малым штрафом, то среди них выбирается тот, который был удалён раньше всего. Если удалённых малым штрафом игроков в команде нет, то никаких досрочных выходов не случается.

По ходу игры удаления приводят к тому, что команды играют в самых разнообразных форматах по количеству полевых игроков на площадке. Будем обозначать формат игры записью вида `AxB`, который означает, что в данный момент у первой команды на площадке находится `A` полевых игроков, а у второй — `B`. Например, в начале игры у каждой команды по 5 полевых игроков на площадке и такой формат обозначается `5x5`. Если же у первой команды в данный момент удалены два игрока, а у второй — один, то формат игры будет обозначаться как `3x4`.

Вам дан протокол игры: в какие моменты времени происходили удаления и голы. Требуется по протоколу посчитать, в каких форматах и сколько времени играли команды.

Формат входных данных

В первой строке входного файла дано целое число N — количество событий в матче ($0 \leqslant N \leqslant 1\,000$).

В следующих N строках записаны события матча, по одному на каждой строке. События записаны в формате:

```
mm:ss.d team type
```

Здесь `mm:ss.d` — время события с точностью до десятых долей секунды ($0 \leqslant$ `mm` $\leqslant 59$, $0 \leqslant$ `ss` $\leqslant 59$, $0 \leqslant$ `d` $\leqslant 9$), `team` — номер команды (либо 1, либо 2), `type` — тип события:

- `goal` — команда забила гол;
- `minor` — игрок команды получил малый штраф;
- `major` — игрок команды получил большой штраф.

Гарантируется, что для событий типа `goal` десятая доля секунды ненулевая, то есть `d` $\neq 0$, а для событий типа `minor` и `major` — всегда нулевая, то есть `d` $= 0$.

События даны в хронологическом порядке, то есть они упорядочены по неубыванию времени события. Гарантируется, что в любой момент времени у каждой команды удалено не более 5 игроков.

Формат выходных данных

Для каждого формата игры, в котором команды отыграли ненулевое время, требуется на отдельной строке через пробел вывести обозначение формата и проведённое в нём время. Формат времени должен быть в точности такой же, который используется во входных данных.

Строки можно выводить в произвольном порядке.

Пример

input.txt	output.txt
10 06:41.0 1 minor 07:20.4 2 goal 22:22.0 2 minor 22:32.0 1 minor 23:00.1 1 goal 23:12.0 2 minor 23:59.9 1 goal 41:02.0 1 major 41:04.5 2 goal 59:00.0 1 minor	4x3 00:47.9 4x4 01:12.1 4x5 06:39.4 5x4 00:50.0 5x5 50:30.6

Пояснение к примеру

В игре из примера были следующие игровые отрезки:

- `[00:00.0; 06:41.0)` — до первого удаления игра шла в стартовом формате `5x5`;
- `[06:41.0; 07:20.4)` — после удаления команды играли в формате `4x5`, пока первая команда не пропустила гол, играя в меньшинстве, после чего вышел удалённый малым штрафом игрок;
- `[07:20.4; 22:22.0)` — до очередного удаления команды играли в полных составах `5x5`;
- `[22:22.0; 22:32.0)` — до следующего удаления команды играли в формате `5x4`;
- `[22:32.0; 23:00.1)` — до гола команды играли в формате `4x4`, однако, после гола никто не площадку не выходит, т.к. он был забит в равных составах;
- `[23:00.1; 23:12.0)` — до очередного удаления команды продолжили играть `4x4`;
- `[23:12.0; 23:59.9)` — затем команды играли в формате `4x3`, пока не был забит гол и не вышел игрок второй команды, удалённый в `22:22.0`;
- `[23:59.9; 24:32.0)` — команды играли `4x4` пока не закончился штраф игрока первой команды;
- `[24:32.0; 25:12.0)` — команды играли `5x4` пока не закончился штраф игрока второй команды;
- `[25:12.0; 41:02.0)` — до большого штрафа игра шла в полных составах `5x5`;
- `[41:02.0; 41:04.5)` — до гола команды играли в формате `4x5`, но, поскольку игрок пропустившей команды имел большой штраф, то он **не** выходит на площаку;
- `[41:04.5; 46:02.0)` — команды продолжали игрока `4x5`, пока не закончился штраф игрока первой команды;
- `[46:02.0; 59:00.0)` — до удаления команды играли в полных составах `5x5`;
- `[59:00.0; 60:00.0)` — команды заканчивали игру в формате `4x5`.

Задача 6. Дни недели

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

Архитектор создал мультивселенную, в которой M вселенных. В каждой вселенной изначально выставлен свой день недели. Также у архитектора есть N кнопок. Каждая кнопка подключена к некоторому набору вселенных. При нажатии на кнопку во всех подключенных к ней вселенных день недели переводится на один вперёд.

Архитектор интересуется: а существует ли такая конфигурация дней недели во вселенных, которая недостижима никакой последовательностью нажатий на кнопки? Помогите архитектору решить эту задачу.

Во всех вселенных используется земная неделя: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday. При переводе дня недели на один вперёд: Monday меняется на Tuesday, Tuesday на Wednesday, и так далее, Sunday меняется на Monday.

Архитектор может нажимать на кнопки сколько угодно раз и в любом порядке. При нажатии на кнопку все дни недели в подключенных к кнопке вселенных изменяются мгновенно и одновременно.

Формат входных данных

В первой строке записано целое число T — количество тестов во входном файле ($1 \leqslant T \leqslant 500$). Далее записаны тесты.

Каждый тест начинается со строки с двумя целыми числами: N и M — количество кнопок и количество вселенных соответственно ($1 \leqslant N, M \leqslant 500$).

Во второй строке теста записана начальная конфигурация: какой день недели выставлен изначально в каждой вселенной. Вселенные пронумерованы от 1 до M: в этом же порядке записываются дни недели для вселенных в конфигурации.

Далее описываются кнопки. Для каждой кнопки в отдельной строке записано, к каким вселенным она подключена, в формате: первое число K задаёт количество этих вселенных, а следующие K чисел обозначают номера этих вселенных ($0 \leqslant K \leqslant M$). Гарантируется, что все заданные номера разные для каждой кнопки.

Гарантируется, что суммарное количество кнопок по всем тестам не превышает 500, и суммарное количество вселенных по всем тестам также не превышает 500.

Формат выходных данных

Для каждого теста в отдельной строке требуется вывести ответ.

Если все 7^M возможных конфигураций дней недели можно получить, выведите лишь слово `NO` в качестве ответа. Иначе выведите слово `YES`, а затем через пробел любую недостижимую конфигурацию.

Примеры

input.txt	output.txt
3 3 3 Monday Saturday Thursday 1 1 1 2 1 3 3 3 Friday Thursday Thursday 1 3 1 3 1 3 4 3 Sunday Sunday Monday 2 1 3 3 1 2 3 0 1 3	NO YES Saturday Thursday Thursday NO

Задача 7. Шары

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

Алиса и Боб играют в игру. Перед ними разложены B синих и R красных шаров. Алиса ходит первой, далее игроки чередуют ходы. Алиса в свой ход выбирает один случайный шар и убирает его. Боб, в свою очередь, убирает один красный шар.

При выборе случайного шара Алисой все разложенные шары выбираются с равной вероятностью независимо от их цвета. Какой именно красный шар убирать Бобу — значения не имеет.

Игра завершается, как только наступает один из двух исходов:

- синих шаров не осталось — в этом случае радуется Алиса;
- синих шаров стало строго больше чем красных — в этом случае радуется Боб.

Алиса и Боб стремятся к балансу исходов, поэтому их интересует, сколько для игры с $C = B + R$ шарами нужно взять синих шаров, чтобы вероятность радоваться Алисе h была как можно ближе к 50%. То есть требуется минимизировать значение $|h - 0.5|$.

Формат входных данных

В первой строке входного файла записано единственное целое число G — количество игр, которые Алиса и Боб планируют сыграть ($1 \leqslant G \leqslant 10^5$).

Далее в отдельных строках указано количество шаров C в каждой игре ($2 \leqslant C \leqslant 2 \cdot 10^5$).

Формат выходных данных

Для каждой игры в порядке следования во входном файле выведите в отдельной строке количество синих шаров, при котором вероятность радоваться Алисе как можно ближе к 50%.

Примеры

input.txt	output.txt
5 2 3 6 7 8	1 1 2 1 2

Задача 8. Ромуальдыч и остатки

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

Прознал старик Ромуальдыч про деление с остатком и аж заколдобился. Индо взопрел вместе с озимыми. И постигла его дума тягучая, а не сыщется ли в наперед заданном интервале $[a, b]$ некоторое число x, да не абы какое, а непременно, чтобы оно при делении на некоторое число y выдало бы в остатке заданное число r. Да вот только с проблемой этой ему никак не управиться, сколь ни нюхай свою портянку.

Формат входных данных

В первой строке записано одно целое число T — количество тестов в файле ($1 \leqslant n \leqslant 200\,000$).

В каждой из последующих T строк записано по три целых числа: a, b — границы интервала, и r — требуемый остаток ($0 \leqslant a \leqslant b \leqslant 10^{18}$, $0 \leqslant r \leqslant 10^{18}$).

Формат выходных данных

Выведите T ответов в том порядке, в котором тесты перечислены во входном файле, по одному в строке.

Каждый ответ — это два целых числа x и y, такие что $a \leqslant x \leqslant b$, $1 \leqslant y \leqslant 2 \cdot 10^{18}$, и остаток от деления x на y равен r. Если вариантов ответа несколько, выберите любой вариант с минимальным x. Если вариантов ответа нет, выведите два целых числа $x = -1$ и $y = -1$.

Пример

input.txt	output.txt
2 6 8 0 3 5 10	6 3 -1 -1

Пояснение к примеру

В первом тесте 6 делится на 3, то есть остаток от деления действительно 0. При этом 6 — наименьшее число в интервале [6,8]. Можно вывести вместо этого ответ $x = 6$ и $y = 2$ (минимальность y не требуется), а вот ответ $x = 8$ и $y = 4$ вывести нельзя, т.к. в нём x не минимален.

Во втором тесте ответов нет, так как для x в интервале [3,5] остаток 10 получить нельзя, на какой бы y мы не делили.

Задача 9. Многочлены

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

На левой стороне доски выписаны N многочленов, а на правой — M многочленов. Ваша задача состоит в том, чтобы *сконструировать* каждый из M многочленов с правой стороны доски за минимальное число действий.

Чтобы сконструировать многочлен с правой стороны доски, сначала нужно выбрать любой из N многочленов с левой стороны доски. Затем к выбранному многочлену можно применять преобразования следующих видов:

- Дифференцирование: многочлен заменяется на его производную.
- Интегрирование: многочлен заменяется на его первообразную с произвольной константой интегрирования.

Преобразования можно применять в любом порядке сколько угодно раз, однако каждое применение считается за одно действие. Можно вообще не применять преобразования, если многочлен с правой стороны доски сам по себе совпадает с каким-нибудь многочленом с левой стороны.

Формат входных данных

В первой строке записано два целых числа: N и M ($1 \leqslant N, M \leqslant 10^5$).

В каждой из следующих $N + M$ строк описаны многочлены, по одному многочлену в строке. Первые N многочленов — это многочлены с левой стороны доски, остальные — с правой.

Описание многочлена $a_0 + a_1 x + \cdots + a_K x^K$ начинается с целого неотрицательного числа K — его степени. Далее записаны целые числа $a_0, a_1, \ldots, a_K$ — коэффициенты многочлена ($-10^9 \leqslant a_i \leqslant 10^9$). При этом $a_K \neq 0$.

Гарантируется, что сумма степеней всех многочленов с левой стороны доски не превышает 10^5. Аналогично для многочленов с правой стороны доски.

Формат выходных данных

Для каждого многочлена с правой стороны доски выведите минимальное число действий, необходимое для его конструирования. Ответы выводите по одному в строке, в том же порядке, в котором многочлены заданы во входном файле.

Примеры

input.txt	output.txt
2 1 2 1 1 1 2 7 6 2 2 7 6 1	4
2 1 2 1 1 1 0 1 1 1 1	1

Пояснение к примеру

Во втором примере на левой стороне доски написаны многочлены $p_1(x) = 1 + x + x^2$ и $p_2(x) = 7 + 6x + 2x^2$. Нужно получить многочлен $q(x) = 7 + 6x + x^2$. Продифференцируем p_1 дважды, получив $p_1'(x) = 1+2x$, а затем $p_1''(x) = 2$. Теперь проинтегрируем $p_1''(x)$ с константой 6, получив $6 + 2x$. Результат проинтегрируем с константой 7, получив $7 + 6x + x^2$. Итого 4 действия.

Задача 10. Найти комнату

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	6 секунд
Ограничение по памяти:	512 мегабайт

Ходислав играет в свою любимую roguelike игру. Каждый уровень игры является клеточным полем, в каждой клетке которого либо свободно, либо стоит стена. Но пока игрок не окажется рядом с клеткой, он не знает, что в ней находится.

Особенно полезные комнаты-сокровищницы могут быть спрятаны так, что игрок не может добраться до них обычным образом. Чтобы в них попасть, нужно использовать заклинание, которое открывает часть клеток уровня, а также заклинание телепортации. К счастью, на wiki-странице игры приведена схема сокровищницы прямоугольной формы. Положение сокровищницы на уровне может быть любым, но ориентация должна быть в точности такой же, как на схеме.

Ходислав уже открыл часть уровня и хочет узнать, где может располагаться сокровищница. Найдите все возможные положения верхнего левого угла сокровищницы, которые не противоречат тому, что известно об уровне. Сокровищница должна входить на уровень целиком.

Формат входных данных

В первой строке записано четыре целых числа: R, C — количество строк и столбцов соответственно на карте уровня, и A, B — количество строк и столбцов соответственно на схеме сокровищницы ($1 \leqslant R, C \leqslant 2\,000$, $1 \leqslant A \leqslant R$, $1 \leqslant B \leqslant C$).

Далее записана карта уровня: R строк по C символов в каждой. Для каждой клетки записан один из трёх символов:

‘#’ (ASCII 35) — в данной клетке стена,
‘.’ (ASCII 46) — в данной клетке пусто,
‘_’ (ASCII 95) — содержимое данной клетки неизвестно.

В оставшихся A строках записана схема сокровищницы, по B символов в каждой. Для каждой клетки записан один из трёх символов:

‘#’ (ASCII 35) — в данной клетке должна быть стена,
‘.’ (ASCII 46) — в данной клетке должно быть пусто,
‘_’ (ASCII 95) — содержимое данной клетки может быть любым.

Формат выходных данных

В первую строку выведите целое число K — количество возможных положений сокровищницы ($0 \leqslant K \leqslant (R-A+1)\cdot(C-B+1)$). В оставшиеся K строк выведите эти положения, по одному в строке. Каждое положение определяется двумя целыми числами u и v, записанными через пробел, — номерами строки и столбца уровня соответственно, в которых расположена левая верхняя клетка из схемы сокровищницы ($1 \leqslant u \leqslant R-A+1$, $1 \leqslant v \leqslant C-B+1$).

Положения должны быть упорядочены по возрастанию u, а в случае равенства u — по возрастанию v.

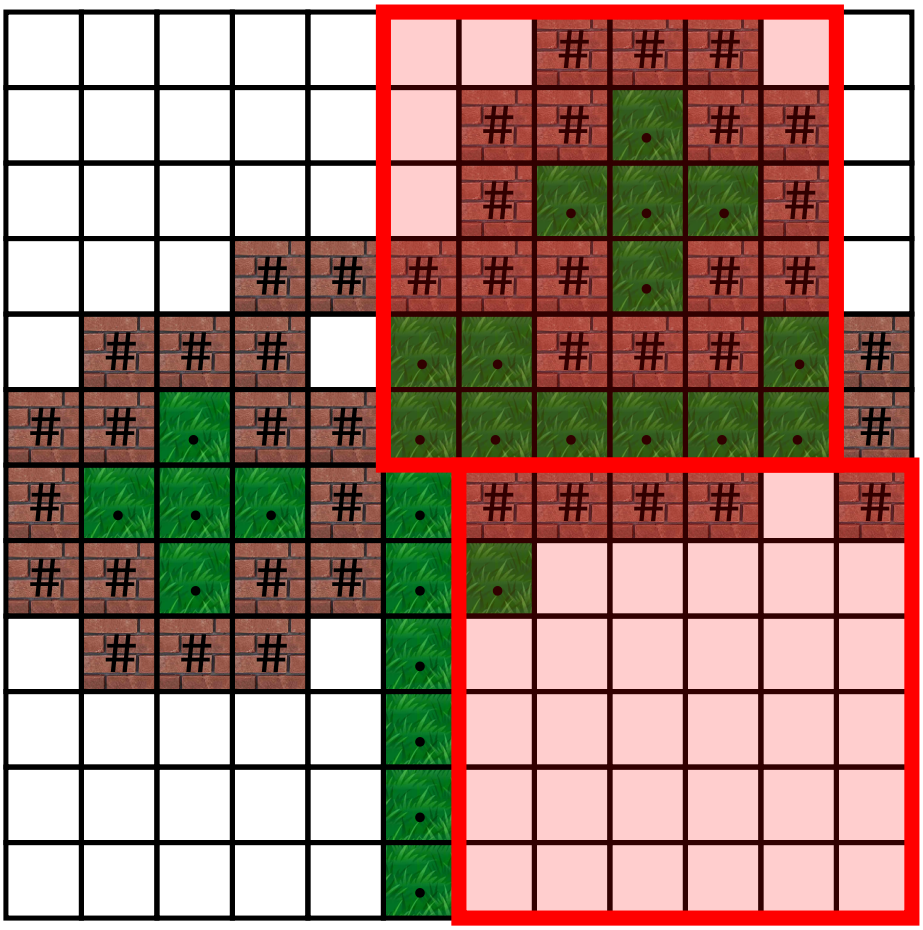
Примеры

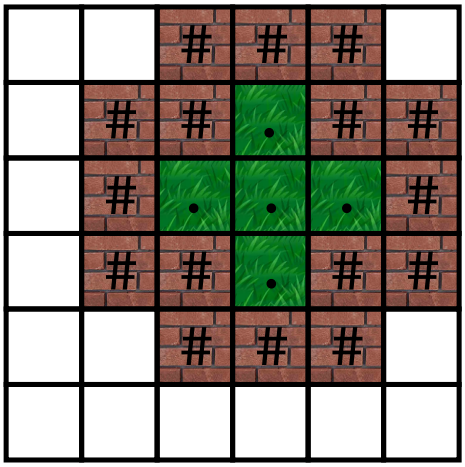
input.txt	output.txt
12 12 6 6 _____###__ _____##.##_ _____#...#_ ___#####.##_ _###_..###.# ##.##.....# #...#.####_# ##.##..____ _###_.____ _____·____ _____·____ _____·____ __###_ _##.## _#...# _##.## __###_ _____	2 1 6 7 7
4 5 2 3 _____ _._## _·_·_ __#__ _## ##.	0

Иллюстрация

Иллюстрация к первому примеру приведена на следующей странице.

Два возможных положения сокровищницы показаны жирной красной рамкой. В нижнем положении содержимое большинства клеток неизвестно: лишь две клетки точно заданы и на карте уровня, и на схеме сокровищницы. Слева видно структуру, очень похожую на сокровищницу. Однако она не полностью входит на уровень — не хватает одного столбца слева.





Задача 11. Увлекательный процесс

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Юля написала на доске число x, а Яша — y. Ребята заскучали и придумали крайне интересное занятие. Раз в минуту ребята одновременно стирают свои числа и пишут на их месте новые. Юля пишет на месте своего числа новое по следующему правилу: если Юлино число было равно i, то оно заменяется на число f_i. Яша делает со своим числом то же самое, но по другому правилу: если Яшино число было равно i, то оно заменяется на число g_i.

Ребята закончат своё занятие, когда их числа совпадут. Возможно, это произойдёт сразу же (если $x = y$), может быть позже, а может их числа не совпадут никогда. Ваша задача состоит в том, чтобы определить для разных значений x и y, закончат ли ребята выписывать числа.

Формат входных данных

В первой строке записаны два целых числа: N и Q ($1 \leqslant N, Q \leqslant 10^5$).

Во второй строке через пробел записаны N чисел: $f_1, \ldots, f_N$.

В третьей строке в том же формате числа: $g_1, \ldots, g_N$.

В j-ой из следующих Q строк — начальные числа x_j и y_j.

Гарантируется, что все числа f_i, g_i, x_j, y_j целые и лежат в диапазоне от 1 до N.

Формат выходных данных

Выведите Q строк: в j-ой напишите `YES`, если процесс, начавшись с чисел x_j и y_j, закончится, и `NO` в противном случае.

Примеры

input.txt	output.txt
3 2 2 3 1 2 3 1 1 2 1 1	NO YES
4 2 2 3 4 2 2 4 4 1 1 2 1 4	NO YES