Задача A. Морской бой — 3

Имя входного файла:	`battle3.in`
Имя выходного файла:	`battle3.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Петя с Васей играют в «Морской бой». Как известно, «Морской бой» — это игра для двух участников, в которой игроки по очереди называют координаты на неизвестной им карте соперника (говорят, что игрок произвел выстрел в эту клетке). Если у соперника по этим координатам имеется корабль, то корабль или его часть «топится», а попавший получает право сделать еще один ход. Цель игрока — первым поразить все корабли противника.

Игра у Васи с Петей происходит на клетчатом поле размером $n \times m$, причем на этом поле размещены корабли, имеющие форму прямоугольников размером $1 \times k$ или $k \times 1$. Уже сделано несколько ходов в игре, причем все выстрелы, которые произвел Вася оказались неудачными, то есть он ни разу не назвал клетку, в которой находится часть корабля, который поставил Петя.

Для того, чтобы прикинуть свои шансы на победу в игре, Вася хочет найти число способов поставить на поле один корабль размером $1 \times k$ или $k \times 1$. При этом ни в одну клетку, которую будет занимать корабль, не должен был быть Васей произведен выстрел.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m ($2 \le n, m \le 100$). Последующие n строк описывают поле — каждая из них содержит по m символов: j-ый символ i-ой из этих строк есть «`#`», если в ходе игры Вася производил выстрел в эту клетку, и «`.`» — в противном случае.

Последняя строка входного файла содержит целое число k ($1 \le k \le 5$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

<table>
<tr><th>battle3.in</th><th>battle3.out</th></tr>
<tr><td><pre>4 4
....
....
....
....
2</pre></td><td><pre>24</pre></td></tr>
<tr><td><pre>3 2
..
.#
##
2</pre></td><td><pre>2</pre></td></tr>
</table>

Задача B. Казино

Имя входного файла:	`casino.in`
Имя выходного файла:	`casino.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

Новое греческое казино «Лас Фигас» открывается в Лас Вегасе. Сергей Андреевич собирается посетить открытие казино. И конечно же он будет играть на рулетке.

В «Лас Фигас» планируется представить игрокам новую игру на рулетке. На рулетке имеется s секторов, и есть m возможных видов ставок, i-й вид ставки заключается в следующем. Игрок ставит одну фишку и выбирает любые p_i секторов рулетки. Если один из выбранных им секторов выпадает, то игрок получает w_i фишек и возвращает свою фишку. Иначе он теряет свою фишку. Каждый сектор выпадает с равной вероятностью $1/s$.

Сергей Андреевич планирует использовать следующую стратегию. Исходно у него n фишек. Он выбирает один из видов ставки и делает ее. После розыгрыша он снова выбирает ставку и делает ее. Так он продолжает игру, пока у него не окажется строго больше n фишек. В этом случае он заканчивает игру и уходит от рулетки, считая, что выиграл. Также он заканчивает игру, если проигрывает все фишки. Выбирая, какую ставку сделать, Сергей Андреевич пытается максимизировать вероятность того, что он выиграет.

По заданным параметрам рулетки и числу n определите максимальную возможную вероятность выигрыша.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m и s ($1 \le n \le 50$, $1 \le m \le 20$, $1 \le s \le 100$). Следующие m строк содержит ао два целых числа: p_i и w_i ($1 \le p_i \le s$, $1 \le w_i \le 50$).

Формат выходного файла

Выведите одно вещественное число — максимальную возможную вероятность выигрыша. Ваш ответ будет проверяться с точностью 10^{-7}.

Примеры

casino.in	casino.out
2 1 2 1 1	0.6666666666666667
10 8 38 18 1 12 2 9 3 6 5 4 8 3 11 2 17 1 35	0.8919856914626223

Задача C. Дешевые Китайские чипы

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

Воодушевленные призывами руководства страны развивать нанотехнологии, руководилети Научно-исследовательского института федеральных исследований глобальных аномалий (НИИ ФИГА) решили скопировать китайский нанопроцессор ЖО-2009. К сожалению определить, как устроен этот процессор, не так то просто.

Исследования показали, что в процессоре используются два типа наноэлементов, ботвизатор и расквантизатор. Эти наноэлементы располагаются на прямоугольной сетке размером $m \times n$ ячеек. Каждый ботвизатор имеет размер 1×1, а каждый расквантизатор — размер $h \times 1$. расквантизаторы всегда располагаются таким образом, чтобы их длинная сторона (длины h) была параллельная вертикальной стороне нанопроцессора (которая имеет длину m). Ботвизаторы и расквантизаторы полностью заполняют сетку, не оставляя свободных ячеек.

Однако узнать точную структуру нанопроцессора не представляется возможным, поскольку процессор упакован в прочный термостойкий наноконтейнер. Все что удалось узнать с использованием тестирования с использованием черного наноящика — это количество ботвизаторов в каждой строке сетки и количество расквантизаторов в каждом столбце сетки.

Теперь вам поручено попытаться восстановить структуру процессора.

Формат входного файла

Первая строка входного файла содержит m, n и h ($1 \le m, n \le 500$, $1 \le h \le m$). Вторая строка содержит m целых чисел a_i ($0 \le a_i \le n$) — количество ботвизаторов в каждой строке сетки (сверху вниз). Третья строка содержит n целых чисел b_i ($0 \le b_i \le m$) — количество расквантизаторов в каждом столбце (слева направо).

Формат выходного файла

Выведите m строк по n символов — реконструированный нанопроцессор. Если в соответствующей ячейке находится ботвизатор, выведите '`*`', если же там находится расквантизатор, выведите '`+`' если это верхняя или нижняя ячейка расквантизатора, или '`|`', если это его внутренняя ячейка. Если решений несколько, выведите любое.

Если реконструировать чип невозможно, потому что данные несовместны, выведите "`inconsistent`" на первой строке выходного файла.

Примеры

chip.in	chip.out

`4 4 3`
`2 1 1 3`
`1 1 0 1`
	`+**+`
	`|+*|`
	`+|*+`
	`*+**`

`4 4 3`
`1 1 1 1`
`1 1 1 1`
	`inconsistent`

Задача D. Дискуссии

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

— Парламент — не место для дискуссий!
Борис Грызлов, спикер Государственной Думы Российской Федерации

Министерство Бюрократии подготовило новый законопроект, который оно планирует внести в парламент. В министерстве хотят чтобы в парламенте были дискуссии по проекту, но они не хотят, чтобы эти дискуссии были на тему: принять проект или нет. Поэтому они планируют подготовить несколько вариантов законопроекта, отличающиеся формулировкой нескольких пунктов, чтобы парламент обсуждал различия в вариантах, и потом принял один из них.

В министерстве планируют предложить парламентариям n вариантов законопроекта и потом порекомендовать отобрать k из них для финального голосования. При этом министр бюрократии хочет, чтобы количество способов сделать это совпадало с z — любимым числом спикера парламента. Теперь ему интересно, какое минимальное число вариантов законопроекта надо подготовить, чтобы можно было подобрать такое k.

Формат входного файла

Входной файл содержит целое число z ($1 \le z \le 10^{100}$).

Формат выходного файла

Выведите минимальное число вариантов законопроекта, которое надо подготовить.

Примеры

discuss.in	discuss.out
6	4
7	7

Задача E. Балда

Имя входного файла:	`game.in`
Имя выходного файла:	`game.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайтов

Последнее время в среде компьютерщиков и программистов набирает популярность игра в балду на программистском сленге.

Напомним правила игры в балду. Два игрока начинают с пустого слова и по очереди добавляют к нему буквы. Можно добавлять букву либо в начало, либо в конец слова. Тот игрок, после хода которого получается слово из программистского сленга, проигрывает. Также игрок проигрывает, если после его хода слово не является подстрокой никакого слова из программистского сленга.

Например, игра может происходить следующим образом:

Первый игрок: “m”

Второй игрок: “mp”

Первый игрок: “0mp”

Второй игрок: “c0mp”

“c0mp” — это слово из программистского сленга, второй игрок проиграл.

По заданному набору слов программистского сленга, определите, кто выигрывает при оптимальной игре обоих игроков, и если это первый игрок, то какую букву ему следует добавить к слову первым ходом.

Формат входного файла

Первая строка входного файла содержит n — количество слов в компьютерном сленге ($1 \le n \le 200$). Следующие n строк содержат по одному слову каждая, слово состоит из символов с ASCII кодами от 33 до 126, длина каждого слова не превышает 50. Большие и маленькие буквы следует различать.

Формат выходного файла

Выведите “`First`”, если первый игрок может выиграть, либо “`Second`”, если выигрывает второй игрок. Если первый игрок выигрывает, то на второй строке выведите все символы, которыми он может начать игру. Символы следует выводить в порядки возрастания их ASCII кодов.

Пример

game.in	game.out
10 one two three four five six seven eight nine ten	First fu

Задача F. Четырехугольник

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

На плоскости заданы четыре точки (возможно, некоторые из них совпадают). Необходимо выяснить, являются ли они четырьмя вершинами некоторого квадрата. Стороны этого квадрата необязательно должны быть параллельны осям координат.

Формат входного файла

Входной файл содержит четыре строки, каждая из которых содержит по два целых числа — координаты одной из заданных точек. Все числа во входном файле не превосходят 1000 по абсолютной величине.

Формат выходного файла

Если заданные точки являются вершинами некоторого квадрата, то выведите в выходной файл слово `YES`, иначе — выведите слово `NO`.

Примеры

quadr.in	quadr.out
0 0 1 1 0 1 1 0	YES
0 0 0 0 1 1 0 1	NO

Задача G. Последовательность

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

Вася очень любит последовательности. Целыми днями он выписывает их в тетрадке на полях во время уроков. Но Вася выписывает не случайные числа. Вася выписывает последовательность, заданную следующими соотношениями:

$$a_1 = w$$

$$a_i = (x \cdot a_{i-1} + y) \mod z, \text{ если } i > 1$$

Здесь x, y, z и w — некие натуральные числа, которые первыми пришли Васе на ум. Не один и не два, а три дня Вася выписывал на полях свою последовательность и, наконец, понял, что выписал n чисел.

И все бы хорошо, если бы Васин друг Петя, не спросил, какое число является k-ым по порядку, если эту последовательность упорядочить по неубыванию.

И тут на Васю напала лень, поэтому он обратился к Вам за помощью. Ответьте на Петин вопрос.

Формат входного файла

В первой строке находятся четыре числа x, y, z и w. Эти числа положительны и не превосходят 10000. Во второй строке находятся два целых числа n и k ($1 \le k \le n \le 10^5$).

Формат выходного файла

В выходной файл выведите ответ на Петин вопрос.

Примеры

sequence.in	sequence.out
1 1 100 2 5 2	3
1 1 100 200 3 2	2

Задача H. Счастливый билетик — 3

Имя входного файла:	`ticket3.in`
Имя выходного файла:	`ticket3.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Назовем *билетиком* последовательность цифр длины $2 \cdot n$. Билетик A называется *счастливым*, если выполняется хотя бы один из следующих критериев:

- сумма цифр первой половины билетика равна сумме цифр второй половины $(\sum\limits_{i=1}^{n} a_i = \sum\limits_{i=n+1}^{2n} a_i)$;
- сумма цифр на нечетных позициях равна сумме цифр на четных позициях $(\sum\limits_{i=1}^{n} a_{2i} = \sum\limits_{i=1}^{n} a_{2i-1})$;
- сумма четных цифр билетика равна сумме его нечетных цифр.

Ваша задача — проверить, является ли заданный билетик счастливым.

Формат входного файла

Первая строка входного файла содержит число n ($1 \le n \le 5000$). Во второй строке содержатся цифры a_1, a_2, $\ldots$, a_{2n} ($0 \le a_i \le 9$), разделенные пробелами.

Формат выходного файла

В выходной файл выведите `YES`, если билетик счастливый, и `NO` — в противном случае.

Примеры

ticket3.in	ticket3.out
3 2 1 1 1 1 2	YES
3 2 1 1 1 1 3	NO

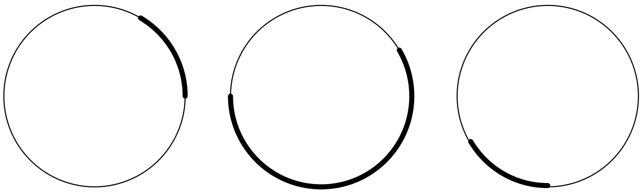
Задача I. Две дуги

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

В этой задаче речь пойдет о дугах одной и той же окружности.

Каждая дуга будет задаваться градусными мерами ее двух концов. Эти градусные меры являются целыми числами от $0°$ до $359°$ включительно и измеряются против часовой стрелки от луча, направленного вправо. Так как в общем случае существуют две дуги с заданными концами, мы будем считать, что дуга проходит из первой точки в сторону возрастания угла (с учетом перехода через отметку $0°$).

На рисунке изображены дуги, задаваемые соответственно как $(0; 60)$, $(180; 30)$, $(210; 270)$:



В случае совпадения концов дуг условимся считать, что таким образом задана дуга, состоящая из одной точки.

Заданы две дуги окружности. Необходимо проверить, имеют ли они хотя бы одну общую точку.

Формат входного файла

В первой строке входного файла находится два целых числа S_1, F_1 — описание первой дуги. Во второй строке находится описание второй дуги в том же формате.

Все числа во входном файле неотрицательны и строго меньше 360.

Формат выходного файла

В выходной файл выведите «`YES`», если дуги имеют хотя бы одну общую точку, или «`NO`», если не имеют общих точек.

Примеры

twoarcs.in	twoarcs.out
0 60 210 270	NO
210 270 180 30	YES
180 30 0 60	YES