Problem A. Fantastic Experiment

Input file:	*standard input* или fantastic.in
Output file:	*standard output* или fantastic.out
Time limit:	2 seconds
Memory limit:	64 mebibytes

Байтландские учёные в рамках практических исследований в области прикладной оптики (исследование предполагаемых свойств полной невидимости) провели следующий мысленный эксперимент.

Стрелок, находящийся в состоянии полной невидимости и вооружённый лазерным пистолетом, находится в комнате с зеркальными стенами. В некоторой точке комнаты находится цель. Задача стрелка — сделать «выстрел» (то есть направить луч) таким образом, чтобы тот, отразившись ровно N раз от стен, попал в цель. Отражение происходит в соответствии с законами оптики, стрелок и цель считаются точками, при этом стрелок считается прозрачным для луча, а цель поглощает свет целиком. При попадании в угол считается, что произошло сразу два отражения.

Учёных интересует минимальная длина получившейся «траектории» (то есть линии, образованной лучом лазера и идущей от точки выстрела до цели).

По заданным координатам стрелка и цели, а также размерам комнаты вычислите оптимальный результат (то есть наименьшую длину требуемой линии).

Input

Входной файл состоит из не более, чем 100 тестовых примеров. Каждый тестовый пример задан в одной строке и содержит 7 целых чисел X, Y, S_x, S_y, T_x, T_y, N — соответственно размеры комнаты, координаты стрелка, координаты цели, требуемое количество отражений ($2 \le X, Y \le 100,\ 0 < S_x, T_x < X,\ 0 < S_y, T_y < Y,\ 0 \le N \le 100$). Точка $(0, 0)$ находится в одном из углов комнаты, координаты всех точек комнаты неотрицательны. Входной файл завершается тестовым примером, состоящим из семи нулей, обрабатывать который не требуется.

Output

Для каждого тестового примера в отдельной строке выведите наименьшую длину очерчиваемой лучом линии с точностью 10^{-3}.

Example

standard input или fantastic.in	*standard output* или fantastic.out
18 14 7 1 6 1 1 20 40 2 4 14 32 2 0 0 0 0 0 0 0	2.236 39.39543

Problem B. Sum in Fibonacci System

Input file:	*standard input* или fibsum.in
Output file:	*standard output* или fibsum.out
Time limit:	2 seconds
Memory limit:	64 mebibytes

Байтландские учёные разработали экспериментальный компьютер, вместо двоичной системы использующий так называемую систему счисления Фибоначчи.

Известно, что любое целое положительное число можно разложить на сумму чисел, каждое из которых будет являться числом из последовательности Фибоначчи $(1, 2, 3, 5, 8, 13, \dots)$, более того, все эти числа будут различными и среди этих чисел не будет двух, которые в последовательности Фибоначчи идут друг за другом.

Каждое такое разложение можно представить в записи, которая очень похожа на двоичное представление числа. Если в i-ой позиции стоит единица, то i-ое число Фибоначчи присутствует в разложении числа, иначе там будет стоять ноль. Назовем такой способ записи системой счисления Фибоначчи. При этом в записи не будет двух подряд идущих единиц.

Недавно было объявлено о том, что в новом компьютере уже работает модуль суммирования. Ваша задача — проверить качество его работы, написав соответствующую программу для обычного компьютера. На вход даются два целых числа A и B, записанных в системе счисления Фибоначчи. На выходе требуется найти их сумму.

Input

В первой и второй строках входного файла находятся целые положительные числа A и B соответственно. Оба числа записаны в системе счисления Фибоначчи, длина записи каждого из чисел не превосходит 10^4 символов.

Output

Выведите представление суммы A и B в системе счисления Фибоначчи. При этом в записи должны отсутствовать ведущие нули.

Examples

standard input или fibsum.in	*standard output* или fibsum.out
1 1	10
1 10	100
100101 101	1000000
101010 10101	1010100

Problem C. Find a path

Input file:	*standard input* или findpath.in
Output file:	*standard output* или findpath.out
Time limit:	2 seconds
Memory limit:	64 mebibytes

Байтландские учёные поставили следующий эксперимент. В лабиринт, расположенный на вертящемся столе, была запущена мышь, умеющая находить кратчайший путь к кормушке. При этом лабиринт обладает тем свойством, что между двумя любыми его клетками существует единственный путь, не проходящий по какой-либо клетке дважды. Передвижения мыши фиксировались с помощью компаса.

Некоторые из клеток лабиринта были оборудованы специальными устройствами, которые в момент, когда через данную клетку проходит мышь, включают двигатели, мгновенно поворачивающие стол на 90 градусов против часовой стрелки. Оказалось, что это не мешает мыши по-прежнему находить кратчайший путь, то есть что мышь оперирует понятиями «влево-вправо-вперёд-назад», а не сторонами света.

Назовём переходом мыши движение между соседними клетками, а направлением этого перехода — направление относительно сторон света, в котором двигается мышь при переходе. По заданному лабиринту, начальному положению мыши и положению кормушки выведите последовательность направлений переходов при прохождении мышью кратчайшего пути к кормушке.

Input

В первой строке входного файла задано целое число T ($1 \le T \le 30$) — количество тестовых примеров. Далее задаются тестовые примеры. В первой строке каждого тестового примера заданы два целых числа X и Y — размеры лабиринта ($1 \le X, Y \le 500$). Следующая строка содержит четыре целых числа M_x, M_y, F_x, F_y — соответственно координаты клетки, из которой стартует мышь, и клетки, в которой расположена кормушка ($0 \le M_x, F_x < X$, $0 \le M_y, F_y < Y$). В последующих Y строках, каждая из которых имеет длину X, задан лабиринт. i-й символ в j-й строке описывает клетку с координатами $(i-1, j-1)$. Значением символа может быть цифра или заглавная латинская буква от '`A`' до '`V`'. Значение интерпретируется как цифра в 32-ичной системе счисления ('`A`' соответствует десятичному числу 10, '`B`' — 11 и так далее до '`V`', соответствующей десятичному числу 31) и строится следующим образом: изначально значение берётся равным нулю. Если из данной клетки есть проход на восток, к значению прибавляется 1, если есть проход на север — 2, на запад — 4, на юг — 8, если данная клетка оборудована устройством поворота стола — 16 (например, значение '`J`'$(16+2+1)$ соответствует клетке с поворотным устройством и выходами на восток и на север). Начало координат находится в северо-западном углу лабиринта. Описание клеток даётся по исходному (неповёрнутому) лабиринту. Гарантируется, что точка старта не оборудована устройством поворота стола.

Output

Для каждого тестового примера в отдельной строке выведите последовательность (возможно, пустую) символов '`N`', '`E`','`S`' и '`W`' (для направлений соответственно на север, восток, юг и запад) — последовательность направлений переходов при движении мыши от начальной точки к кормушке.

Example

standard input или findpath.in	*standard output* или findpath.out
2 1 1 0 0 0 0 1 2 3 0 1 1 0 88 AA JM	SNSS

Problem D. Fire!

Input file:	*standard input* или `fire.in`
Output file:	*standard output* или `fire.out`
Time limit:	2 seconds
Memory limit:	64 mebibytes

Байтландские учёные исследовали шансы на успех противостояния «новичок-опытный игрок» в популярной компьютерной игре «Realm of Tanks».

Была рассмотрена простейшая ситуация битвы N танков опытного игрока против M танков новичка.

Механика игры устроена следующим образом.

- Изначально танки готовы к стрельбе. Как только танк готов к стрельбе, он в тот же момент обязан стрелять по некоторому танку противника (выбор произволен с учётом ограничения, указанного ниже).
- У танков новичка по 107 HP, у танков опытного игрока по 41 HP. Выстрел одного танка новичка наносит урон 10.7 HP, при этом после выстрела готовность к стрельбе наступает через 1.07 секунды. Выстрел одного танка опытного игрока наносит урон 4.1 HP, при этом после выстрела готовность к стрельбе наступает через 0.41 секунды.
- Время полёта снаряда равно 0.0003 секунды.
- Пока количество HP у танка больше нуля, он может вести огонь, как только количество HP станет меньшим или равным нулю, танк уничтожается и в дальнейших боевых действиях не участвует.
- Если в некоторый момент танк противника уничтожается за n попаданий, но не уничтожается за $n-1$, то в этот момент огонь по соответствующему танку вести может не более n танков противника.
- Победителем считается сторона, уничтожившая все танки противника и сохранившая хотя бы один танк. Если уничтожены все танки с обеих сторон, игра считается закончившейся вничью.

Опытный игрок действует оптимальным для себя образом, новичок действует наихудшим для себя образом (но в соответствии с правилами). Байтландских учёных заинтересовал вопрос — кто же выиграет в этом случае?

Input

В первой строке входного файла задано целое число T ($1 \le T \le 10^4$) — количество тестовых примеров. Каждый тестовый пример задаётся двумя целыми числами N и M — соответственно количество танков у опытного игрока и у новичка ($1 \le N, M \le 10^5$).

Output

Для каждого тестового примера выведите в отдельной строке "`Newbie`", если выиграет новичок, "`Master`", если выиграет опытный игрок и "`Elimination`", если партия закончится вничью.

Example

standard input или fire.in	*standard output* или fire.out
2 4 4 8 4	Newbie Master

Problem E. Fleet of Byteland

Input file:	*standard input* или fleet.in
Output file:	*standard output* или fleet.out
Time limit:	2 seconds
Memory limit:	64 mebibytes

Байтландские учёные провели исследование, направленное на повышение эффективности радиообмена между судами байтландского торгового флота за счёт ликвидации пауз между буквами или словами при передаче сообщений азбукой Морзе.

В частности, была рассмотрена следующая задача: дано сообщение, записанное азбукой Морзе, в котором нет пауз ни между буквами, ни между словами. Требуется восстановить исходное сообщение, записанное буквами латинского алфавита, согласно таблице соответствия букв латинского алфавита и символов азбуки Морзе.

A	.-	B	-...	C	-.-.	D	-..
E	.	F	..-.	G	--.	H	....
I	..	J	.---	K	-.-	L	.-..
M	--	N	-.	O	---	P	.--.
Q	--.-	R	.-.	S	...	T	-
U	..-	V	...-	W	.--	X	-..-
Y	-.--	Z	--..				

Разумеется, в данных обстоятельствах способов восстановить сообщение может быть несколько, например сообщение "`.....---...`" может быть интерпретировано и как "`SOS`", и как "`VGI`", и как "`EEETTTEEE`". Поэтому было предложено после сообщения указывать номер его интерпретации при лексикографическом упорядочении.

Ваша задача — по заданному k найти k-ю в лексикографическом порядке интерпретацию сообщения.

Input

Первая строка входного файла содержит полученное сообщение и состоит из не более, чем 50 символов '`.`' (точка) и '`-`' (минус). Во второй строке задано целое число k ($1 \le k \le 2^{63}$) — номер требуемой интерпретации.

Output

Выведите k-ю в лексикографическом порядке интерпретацию исходного сообщения — последовательность прописных латинских букв. Если такой интерпретации не существует, выведите число 0.

Examples

standard input или fleet.in	*standard output* или fleet.out
-... 1	B
.- 4	0

Problem F. Foundation

Input file:	*standard input* или foundation.in
Output file:	*standard output* или foundation.out
Time limit:	2 seconds
Memory limit:	64 mebibytes

Деятельность байтландских учёных поддерживается разнообразными фондами. Одному из таких фондов необходимо распределить A одинаковых грантов между M математиками и B одинаковых грантов между F физиками (соискатель является либо математиком, либо физиком, но не тем и другим одновременно) так, чтобы каждый учёный получил не более одного гранта. Сколькими способами фонд может распределить гранты?

Input

В первой строке входного файла заданы четыре целых числа M, F, A, B ($1 \le A \le M \le 15$, $1 \le B \le F \le 15$) — соответственно количество математиков, физиков, грантов по математике и грантов по физике.

Output

Выведите единственное целое число — количество различных способов распределить гранты.

Examples

standard input или foundation.in	*standard output* или foundation.out
2 2 2 2	1
10 10 9 1	100