

Problem A. Распределение багажа

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Все экономные туристы знают о существовании лоу-кост авиакомпаний, реализующих авиабилеты по невообразимо низким ценам. А те туристы, которые хотя бы раз пользовались их услугами, прекрасно осведомлены и о таких особенностях бюджетных перелётов, как платная регистрация в аэропорту, платное питание и, конечно же, платный багаж.

Распределить багаж по сумкам так, чтобы он удовлетворял всем требованиям, предъявляемым лоу-кост авиакомпаниями — весьма непростая задача, которую желательно решать до приезда в аэропорт.

Турист, отправляясь в путешествие, оплатил в авиакомпании право на перевозку трёх мест багажа. Чтобы максимально эффективно организовать сборы, он заранее посчитал количество способов K распределить N килограммов своего багажа по трём сумкам так, чтобы весь багаж оказался в сумках, и вес каждой сумки был целочисленным и не равнялся нулю. При этом порядок весов различных сумок не учитывался (то есть распределения весов $2\ 2\ 1$ и $2\ 1\ 2$ считались одинаковыми).

По прилёту в пункт назначения турист отправился получать багаж и не обнаружил ни одной из трёх своих сумок. При заполнении заявления на розыск багажа туриста постигла ещё одна неприятность — он не смог вспомнить, сколько килограммов суммарно весил его багаж. Единственное, что ему удалось вспомнить, — это то, что посчитанное им число способов K распределения багажа по трём сумкам было не меньше L и не больше R .

Вам необходимо посчитать, сколько различных целочисленных весов N багажа мог перевозить турист.

Input

В единственной строке заданы числа L и R ($1 \leq L \leq R \leq 10^{17}$) — минимальное и максимальное допустимое количество способов распределения багажа туриста по трём сумкам.

Output

Число различных целочисленных весов N багажа, которые мог перевозить турист.

Examples

standard input	standard output
2 4	3

Note

Комментарий к Примеру 1: допустимые веса багажа туриста — 5, 6, 7.

Problem B. Миграционный контроль

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Никто не застрахован от потери документов. Даже если потерял паспорт, ситуация не является безвыходной. В паспортном столе потребуют подтвердить личность другими документами, заставят подождать несколько недель — и паспорт снова у законного владельца. Но что делать, если паспорт потерялся во время путешествия в далёкой экзотической стране? Выехать нельзя, до консульства не добраться, а языковой барьер только усугубляет проблему.

В миграционной службе одного островного государства фиксируют всех въезжающих туристов. Для идентификации туриста используется последовательность из K полей, в которую записывается составной *идентификатор личности* туриста в соответствии с предъявленным при въезде документом. Понятие идентификатора личности на острове довольно специфическое — нельзя предугадать, в какое из полей попадут настоящие имя, фамилия, отчество туриста, и будут ли они склеены при вводе или нет. Нельзя даже быть уверенным, что оператор заполнит всё без ошибок и не добавит при вводе что-нибудь от себя! Одно известно точно — все K полей любого идентификатора личности являются непустыми строками, состоящими только из строчных букв латинского алфавита.

Турист, обратившийся в миграционную службу, пытается, добавляя и удаляя буквы в конце полей запроса, найти свой идентификатор личности в базе данных. После каждого изменения запрос выполняется, а система отвечает, сколько идентификаторов личности удовлетворяют текущему запросу. Идентификатор личности считается удовлетворяющим запросу, если каждое из K полей в запросе является префиксом соответствующего поля идентификатора.

Изначально все K полей запроса являются пустыми строками. Турист выполнил Q запросов, изменяя текущий запрос одним из двух способов:

1. $\langle i \rangle + \langle c \rangle$: добавление символа c в конец i -го поля.
2. $\langle i \rangle -$: удаление одного символа в конце i -го поля.

Вам необходимо для каждого из Q выполненных запросов ответить, какое количество идентификаторов личности в базе данных ему соответствует.

Input

В первой строке задано число N ($1 \leq N \leq 10^5$) идентификаторов в базе данных миграционной службы и число K ($1 \leq K \leq 3$) полей в идентификаторах.

Далее следуют N описаний идентификаторов. Каждое описание идентификатора представляет собой K полей — непустых строк, состоящих из строчных букв латинского алфавита. Суммарная длина всех полей в базе данных не превышает 10^5 .

В следующей строке записано число Q ($1 \leq Q \leq 10^5$) изменений запроса.

Далее следует Q строк с описаниями изменений запроса вида “ $\langle i \rangle + \langle c \rangle$ ” или “ $\langle i \rangle -$ ” в соответствии с условием. Гарантируется, что при удалении символа из некоторого поля, данное поле не является пустым.

Output

Выведите построчно Q чисел с ответами на запросы туриста в порядке их поступления.

Examples

standard input	standard output
5 2	4
abacaba	3
ccad	1
abadaba	1
cbad	3
abbbaaa	2
cbab	1
bbbba	2
ccc	3
abacabs	4
fuuuu	4
15	5
1 + a	0
2 + c	5
2 + c	1
1 + b	
2 -	
2 + b	
1 + a	
2 -	
2 -	
1 -	
1 -	
1 -	
1 + c	
1 -	
1 + b	

Problem C. Тропический Стоунхендж

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Одно из самых древних сохранившихся до наших дней сооружений на Земле — это Стоунхендж, находящийся в Англии. До сих пор нет единого мнения о его назначении: одни учёные утверждают, что это древнее святилище, другие доказывают, что сооружение представляет собой грандиозную обсерваторию. Но загадки не ограничиваются только английским Стоунхенджем — не так давно учёные обнаружили в лесах Амазонки похожее сооружение и назвали его *Тропическим Стоунхенджем*. Конечно, находка быстро стала одной из главных туристических достопримечательностей в регионе.

Тропический Стоунхендж на момент своего обнаружения находился в полуразрушенном состоянии. После анализа рельефа местности, почвы и состояния камней учёным удалось точно восстановить местоположение всех камней сооружения, кроме одного. Единственное, что может помочь им найти местоположение недостающего камня, — это древний текст, содержащий описание сооружения, из которого следует ряд фактов:

1. Тропический Стоунхендж представляет собой выпуклый многоугольник из N вершин, в которых расположены камни.
2. Никакие 3 вершины многоугольника не лежат на одной прямой.
3. Все вершины многоугольника имеют целочисленные координаты, абсолютное значение которых не превышает 10^6 .
4. Площадь многоугольника равна S .

Итак, зная координаты $N - 1$ камней и площадь S исходного многоугольника, Вам необходимо найти любое допустимое местоположение недостающего камня.

Input

В первой строке задано число N камней ($4 \leq N \leq 10^5$) и площадь исходного многоугольника S ровно с одним знаком после десятичной точки ($1 \leq S \leq 4 \times 10^{12}$).

В следующих $N - 1$ строках в порядке обхода против часовой стрелки заданы координаты известных камней в формате $X Y$ ($|X|, |Y| \leq 10^6$).

Output

Выведите любые допустимые целочисленные координаты недостающего камня в формате $X Y$ ($|X|, |Y| \leq 10^6$). Гарантируется, что для всех тестов существует хотя бы одно решение, удовлетворяющее ограничениям.

Examples

standard input	standard output
4 4.0 0 0 2 0 0 2	1 -2
5 10.0 5 3 1 3 2 1 4 1	1 5

Problem D. Платное шоссе

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Платными дорогами в Европе давно уже никого не удивишь. Как правило, они удобные, качественные и позволяют существенно сэкономить время в пути, по сравнению с обычными дорогами. Сейчас большинство автомобилистов считают затраты на проезд по платным дорогам оправданными, однако в первые годы после появления они не пользовались популярностью.

Ровно N лет назад было построено новое шоссе между двумя крупными городами, представляющее собой один непрерывный участок, проезд по которому был сделан платным на всём его протяжении. Однако эксперимент провалился — только туристы, плохо ориентирующиеся на местности, были готовы платить, чтобы не тратить время. А подавляющее большинство автомобилистов при виде обозначения платной дороги сворачивало в поисках объездного пути.

Проанализировав сложившуюся ситуацию, администрация решила проделать хитрый рекламный трюк, направленный на увеличение потока машин на платном шоссе.

Каждый нечётный год все участки шоссе делят на два новых участка — левый и правый, длины которых находятся в отношении $X : Y$, причём левый участок на ближайший год делают бесплатным, а правый — платным. Каждый чётный год всё происходит наоборот: участки делят в отношении $Y : X$, левый участок делают платным, а правый — бесплатным.

Благодаря этой рекламной кампании, администрации удалось существенно увеличить поток машин на платном шоссе за счёт тех, кто начав движение по бесплатному участку, не решался возвращаться назад при виде следующего платного участка. Но не обошлось и без побочных эффектов — по прошествии N лет схема участков шоссе настолько запуталась, что почти никто не знает их точных длин.

Известно, что длину L_k участка с номером k можно получить из суммарной длины шоссе P по формуле $L_k = P \times (X/100)^{A_k} \times (Y/100)^{B_k}$ для некоторых целых чисел A_k и B_k . Причём A_k — это количество лет с момента первого разделения участков шоссе, которое участок был бесплатным, а B_k — платным.

Вам необходимо отвечать на множество запросов, содержащих номера участков K_i . Ответом на каждый запрос являются значения A_{K_i} и B_{K_i} , соответствующие участку с номером K_i .

Input

В первой строке задаётся число лет N ($1 \leq N \leq 10^{18}$), прошедших с момента постройки шоссе, и проценты $X Y$ ($1 \leq X, Y \leq 99$, $X + Y = 100$), использующиеся при ежегодном разделении участков на новые.

Во второй строке задаётся число запросов Q ($1 \leq Q \leq 10000$).

В следующих Q строках описываются запросы. Каждый запрос представляет собой номер участка K_i ($1 \leq K_i \leq 10^{18}$). Участки нумеруются с 1 в порядке слева-направо.

Output

Для каждого запроса выведите соответствующие участку с номером K_i значения A_{K_i} и B_{K_i} . Числа должны быть целыми и удовлетворять ограничениям $0 \leq A_{K_i}, B_{K_i} \leq 10^{18}$. Гарантируется, что для всех K_i во входе решение в заданных ограничениях существует.

Examples

standard input	standard output
3 25 75 4	2 1
1	3 0
3	1 2
5	1 2
8	

Problem E. Уличный эскалатор

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Самый длинный эскалатор в мире находится в Гонконге. В отличие от привычных эскалаторов в метро, этот эскалатор идёт прямо по улице и был построен для того, чтобы сократить число пробок в городе. Теперь он ежедневно перевозит по несколько десятков тысяч человек в день и уже стал местной туристической достопримечательностью.

Путешествуя в Гонконге, турист решил воспользоваться эскалатором, чтобы спуститься вниз. И каково же было его удивление, когда оказалось, что эскалатор едет только вверх. К счастью на эскалаторе не было ни одного пассажира, поэтому турист решил попробовать добраться до самого низа. Но стоило туристу начать идти по эскалатору, как внизу на эскалатор начали заходить пассажиры и ехать ему навстречу.

Эскалатор едет с постоянной скоростью вверх, а турист движется вниз с постоянной скоростью, превышающей по модулю скорость эскалатора. В любой момент времени каждый человек (турист или некоторый пассажир) находится либо на левой стороне эскалатора, либо на правой. Для каждого пассажира, который встретится туристу до конца его пути, известна сторона, на которую он изначально становится и вероятность того, что в некоторую секунду он меняет сторону. Кроме того, для каждого пассажира задан интервал времени T_i в секундах от момента его захода на эскалатор до момента встречи с туристом. Таким образом, за интервал времени T_i секунд пассажир ровно T_i раз принимает решение о смене стороны.

Турист начинает двигаться по эскалатору с левой стороны и меняет сторону только в том случае, если на ступеньке, на которую он должен сделать шаг, стоит человек с той же стороны, что и он сам. Вам необходимо найти математическое ожидание количества раз, которое турист сменит сторону, прежде чем пройдёт весь эскалатор. Туриста можно считать достаточно худым, чтобы он мог пройти между пассажирами, которые стоят на соседних ступеньках с разных сторон. Временем, которое затрачивается на смену стороны, можно пренебречь.

Input

В первой строке задано число N ($1 \leq N \leq 10^5$) пассажиров, заходящих на эскалатор снизу.

В следующих N строках задаётся описание каждого пассажира, которое содержит через пробел: интервал времени T_i ($1 \leq T_i \leq 10^6$) в секундах от момента захода пассажира на эскалатор до момента встречи с туристом; сторону C_i ($C_i \in \{ 'L' - \text{левая}, 'R' - \text{правая} \}$), на которую становится пассажир в начале пути; вероятность P_i ($0 \leq P_i \leq 1$), заданную десятичной дробью с не более, чем тремя знаками после десятичной точки. Все значения T_i во входных данных различны.

Пассажиры во входных данных заданы в порядке захода на эскалатор, то есть в порядке убывания T_i .

Output

Выведите ожидаемое число смен сторон туристом в виде десятичной дроби. Ваш ответ будет засчитан, если абсолютная или относительная ошибка не превысит 10^{-6} .

Examples

standard input	standard output
2	1.000000000000
5 L 0.5	
1 R 0.4	

Problem F. Посреди Мексики

Input file: *standard input*
Output file: *standard output*
Time limit: 2.5 секунды
Memory limit: 128 мегабайт

Мексика — одна из самых интересных стран для путешественников. Бескрайние песчаные пляжи, самобытная культура, древние храмы индейцев, разнообразная природа — тропические леса, горы, пустыни и... конечно же, кактусы!

Путешественник прибыл в Мексику и решил досконально изучить страну. Для этого он взял в прокате машину и составил план посещения городов страны на K дней вперёд.

Известно, что в Мексике есть N городов и M дорог. Каждая дорога соединяет два различных города, и никакие две дороги не соединяют одинаковую пару городов. При этом все дороги имеют одинаковую длину и являются двусторонними, а из каждого города можно добраться по дорогам в любой другой.

Кроме того, если представить дорожную сеть Мексики в виде графа с вершинами в городах и ребрами-дорогами, то такой граф будет являться *вершинным кактусом* — связным неориентированным графом, в котором каждая вершина лежит не более чем на одном простом цикле.

Турист начинает своё путешествие по городам Мексики в городе 0. Согласно плану, каждый день турист выбирает город для осмотра и едет в него из города, в котором он был в предыдущий день по кратчайшему пути. Если город при этом не меняется, то турист совершает поездку по достопримечательностям в пределах города.

Каждый раз, когда турист куда-то едет, он должен заправить машину в городе, который находится ближе всего к середине его пути. Если таких городов существует несколько, то выбирается любой из них.

Вам необходимо, зная карту дорог в Мексике и номера городов, которые планировал осмотреть турист, определить города, в которых он заправлял машину.

Input

В первой строке задано число N ($1 \leq N \leq 10^5$) городов и число M дорог в Мексике.

В следующих M строках заданы дороги в виде пар чисел U_i и V_i ($0 \leq U_i, V_i \leq N - 1$), обозначающих города, соединяемые i -й дорогой.

Следующая строка содержит целое число K ($1 \leq K \leq 10^5$) — количество городов, запланированных туристом для осмотра.

В следующей строке записаны K номеров осматриваемых туристом городов, согласно плану.

Output

Выведите K целых чисел через пробел — номера городов, в которых турист заправлял машину, перечисленные в хронологическом порядке. Если решений несколько, выведите любое.

Examples

standard input	standard output
6 6	4 2 0 3 4
0 1	
0 2	
2 3	
3 4	
2 4	
4 5	
5	
5 1 3 3 4	

Problem G. Бесплатный Wi-Fi

Input file: *standard input*
Output file: *standard output*
Time limit: 7 секунд
Memory limit: 128 мегабайт

Национальная библиотека Франции — одна из крупнейших библиотек в мире. Но не только уникальное содержание библиотечных фондов привлекает туда туристов. Многие из них посещают библиотеку для того, чтобы воспользоваться бесплатной Wi-Fi сетью.

Здание библиотеки представляет собой поле $N \times M$ клеток. Клетки бывают трех типов:

- 1) препятствие — обозначается символом '#';
- 2) коридор — обозначается символом '.';
- 3) комната — обозначается номером от 1-го до 5-ти, в зависимости от назначения (номера могут повторяться).

Посетители могут проходить между соседними по четырем сторонам клетками, но не могут пройти в клетку с препятствием.

Покрытие сети Wi-Fi в библиотеке достаточно слабое, поэтому в комнатах и коридорах помещения часто можно увидеть посетителей с гаджетами, которые ходят по зданию библиотеки в поисках места с хорошим приёмом сигнала.

Однажды турист с друзьями пришёл в библиотеку, чтобы воспользоваться бесплатным Интернетом и после продолжительных поисков обнаружил место с хорошим приёмом Wi-Fi. В силу плохого знания французского языка точно определить своё местоположение в здании библиотеки турист не смог. Тем не менее он решил рассказать своим друзьям об этом месте и отправился на их поиски. Чтобы впоследствии легко вернуться обратно, он решил записать, сколько клеток и в каком направлении он прошёл. Также он решил записывать последовательность A номеров комнат, в которые приходил, но к этой задаче отнесся не очень внимательно и, возможно, пропустил номера некоторых пройденных комнат. Пока турист разыскивал друзей, наступило время закрытия библиотеки, и он был вынужден отправиться домой.

На следующий день турист вновь пришёл в библиотеку. Получив на входе схему библиотеки, он решил по вчерашним заметкам найти клетку, с которой он вчера отправился на поиски друзей.

Искомая клетка должна удовлетворять следующим критериям:

1. Если начать движение из клетки и перемещаться строго в соответствии с записями, то ни в какой момент времени не окажешься за пределами библиотеки или в клетке с препятствием.
2. Если при движении в соответствии с записями выписать последовательность B номеров посещаемых комнат, то записанная ранее туристом последовательность A будет являться подпоследовательностью B .

Вам необходимо определить, сколько клеток в библиотеке подходят туристу. Учтите, что турист мог ошибиться в записях, и в этом случае искомой клетки не существует.

Input

В первой строке задаются размеры библиотеки N и M ($1 \leq N, M \leq 1000$).

Далее следует описание поля в виде N строк по M символов в каждой. Допустимые символы: '#', '.', '1', '2', '3', '4', '5'. Строки задаются от верхней к нижней. Символы в строках от левого к правому.

Далее следует число K перемещений туриста ($1 \leq K \leq 100$).

В следующих K строках задаются перемещения туриста в виде пар $C_i P_i$, где C_i — символ, обозначающий направление движения (L — движение справа налево, R — движение слева направо, U — движение снизу вверх, D — движение сверху вниз), а P_i ($1 \leq P_i \leq 999$) — число клеток, пройденных в заданном направлении.

В последней строке без пробелов задаётся последовательность номеров комнат A ($1 \leq |A| \leq 100, 1 \leq A_i \leq 5$), выписанных туристом при поиске друзей.

Output

Количество клеток, из которых турист мог отправиться на поиски друзей.

Examples

standard input	standard output
4 5 22.#. 1211# ..#.1 .#122 2 L 2 U 1 121	1
4 5 ..3#. 213.# .#33# 1214. 3 L 1 R 2 U 1 1213	3

Note

Комментарий к Примеру 1: подходит клетка [2 3] (строка столбец).

Комментарий к Примеру 2: подходят клетки [2 2], [4 2], [4 3].

Problem H. Футбольные ставки

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

Путешествуя по Англии, невозможно не заразиться страстью местных жителей к футболу. Почти каждый здесь по своему увлекается футболом — одни играют, другие смотрят, а самые азартные делают ставки на результаты.

Букмекерская контора перед началом чемпионата Англии по футболу запустила новую акцию. Для того чтобы принять в ней участие, игроки должны поставить фиксированную (одинаковую для всех) сумму денег на одну из N команд-участниц чемпионата. Все игроки, угадавшие команду, которая станет чемпионом, получают обратно свою ставку и в качестве выигрыша делят между собой поровну половину от всех ставок, сделанных на проигравшие команды.

Известно, что в ходе прошедшей акции игроки сделали как минимум одну ставку, ни на одну из команд не было сделано более K ставок, а по завершении турнира букмекерская контора получила прибыль в размере P процентов от общей суммы ставок. Вам необходимо найти любое удовлетворяющее условию задачи распределение ставок по командам или ответить, что это невозможно.

Input

В единственной строке задаётся число N ($2 \leq N \leq 100$) команд-участниц турнира, максимально возможное число K ($1 \leq K \leq 100$) ставок на одну команду и число P ($0 \leq P \leq 100$) — процент от суммы ставок, полученный букмекерской конторой в качестве прибыли. Все числа во входных данных целые.

Output

В первой строке выведите номер команды победителя W ($1 \leq W \leq N$).

Во второй строке выведите N целых чисел A_i ($0 \leq A_i \leq K$) — количество ставок, сделанных на каждую из команд.

Если решения, удовлетворяющего условию задачи, не существует — выведите 0.

Examples

standard input	standard output
4 100 10	2 10 80 5 5
2 4 40	1 1 4
5 10 23	0

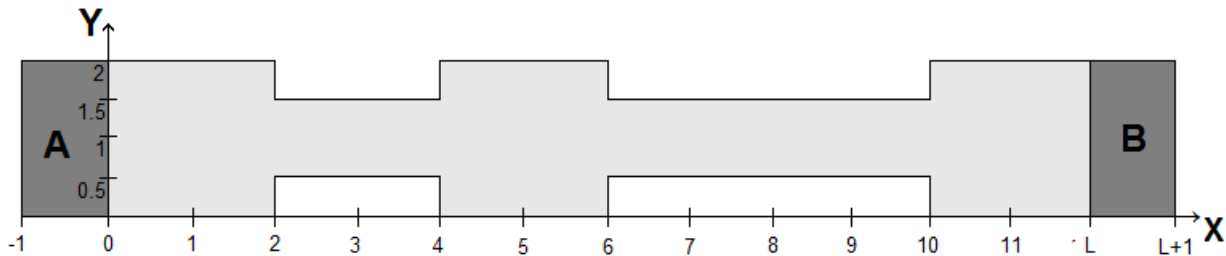
Problem I. Дорога в Драгобрат

Input file: *standard input*
 Output file: *standard output*
 Time limit: 1 секунда
 Memory limit: 128 мебибайт

Драгобрат — самый высокогорный горнолыжный курорт Украины, пользующийся особой популярностью среди фрирайдеров. Но не только катание по хвойному лесу, захватывающие дух пейзажи и устойчивый снежный покров привлекают сюда туристов-экстремалов. Немалую долю адреналина можно получить, добираясь до курорта по узкой и крутой горной дороге, на которой часто не могут разъехаться даже две машины.

Дорога начинается в посёлке Ясиня (пункт A) и заканчивается в Драгобрате (пункт B). В момент времени 0 из пункта A в пункт B и из пункта B в пункт A выходит по одной машине. Как только машина, стартовавшая из некоторого начального пункта, прибывает в свой конечный пункт, она исчезает, а из её начального пункта начинает движение следующая машина. Таким образом, на дороге в любой момент времени находятся не более двух машин.

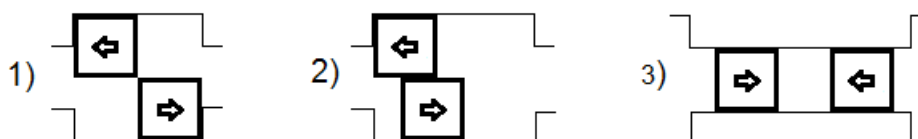
Дорога имеет длину L и представляет собой последовательность из K однополосных и двухполосных участков некоторой длины H_i ($\sum_{i=1}^K H_i = L$). Известно, что 1-й и K -й участки всегда двухполосные и никакие два соседних участка не имеют равного числа полос. Пункты A и B расположены за пределами дороги. Взаиморасположение однополосных и двухполосных участков дороги продемонстрировано на рисунке:



Машина представляет собой квадрат 1×1 , который движется по дороге в соответствии со следующими ограничениями:

- стороны машины всегда параллельны осям координат;
- машина, стартовавшая из пункта A , всегда прижимается к нижней на рисунке стороне дороги;
- машина, стартовавшая из пункта B , всегда прижимается к верхней на рисунке стороне дороги;
- модуль X -составляющей скорости движения всех машин в любой момент времени либо равен нулю, либо равен 1, при этом изменение скорости происходит мгновенно;
- временем движения по оси Y можно пренебречь (например, при перестраивании с двухполосного на однополосный участок и обратно).

Авария на дороге происходит в том случае, если квадраты, представляющие машины, имеют положительную общую площадь.



На рисунке выше случай 1 не приводит к аварии. В случае 2 авария возникнет, если машина, движущаяся влево, попытается заехать на односторонний участок до того, как другая машина полностью

проедет мимо. Случай 3 приведёт к аварии, если ни одна из машин не поменяет направление движения.

Постоянные снегопады затрудняют видимость на дороге, поэтому два водителя видят друг друга только в том случае, если расстояние между машинами по оси X не превышает R . При этом машина в любом случае невидима, когда она находится за пределами дороги (перед началом движения из пункта A или B).

Когда водители двух встречных машин видят друг друга, то по известной карте участков дороги им необходимо определить, есть ли риск возникновения аварии и скорректировать движение таким образом, чтобы её избежать. При этом действуют следующие правила:

1. Если аварии не случится при условии, что скорость машин останется неизменной, то оба водителя продолжают двигаться с неизменной скоростью.
2. Если при неизменной скорости произойдёт авария, то один из водителей должен скорректировать своё движение, а другой — продолжать движение с неизменной скоростью в своём направлении. При этом водитель, корректирующий движение, выбирается так, чтобы время разезда машин было минимальным. В случае если выбор водителя не влияет на время разезда, то движение корректирует водитель машины, стартовавшей из пункта B .

Под временем разезда понимается интервал между моментом времени, когда водители впервые видят друг друга, и моментом времени, когда машина, стартовавшая из пункта A , окажется полностью справа от машины, стартовавшей из пункта B . Никакие другие машины, кроме двух находящихся в текущий момент времени на дороге, при расчёте времени разезда не учитываются.

Вам необходимо посчитать число машин, прибывших из пункта A в B и из пункта B в A за время T . Машина считается прибывшей в конечный пункт, если она полностью покинула дорогу.

Input

В первой строке задано время T ($1 \leq T \leq 10^9$) и радиус видимости на дороге R ($1 \leq R \leq 10^6$).

Во второй строке задано нечетное число K ($1 \leq K < 1000$) участков дороги.

В третьей строке задано K чисел H_i ($2 \leq H_i \leq 1000$) — длины участков дороги.

Все числа во входных данных целые.

Output

Выведите два числа: количество машин, прибывших из пункта A в B , и количество машин, прибывших из пункта B в A , за время T .

Examples

standard input	standard output
40 2 5 2 2 2 4 2	2 3
30 7 3 2 2 2	3 3

Note

Комментарий к Примеру 1:

— из пункта A в B машины прибывают в моменты времени: 15, 28.

— из пункта B в A машины прибывают в моменты времени: 13, 26, 39.

Комментарий к Примеру 2:

- из пункта A в B машины прибывают в моменты времени: 7, 15, 23.
- из пункта B в A машины прибывают в моменты времени: 11, 19, 27.

Problem J. Спортивный ЧГК (Added for Division 1!)

Input file: *standard input*
Output file: *standard output*
Time limit: 4 секунды
Memory limit: 128 мегабайт

Среди студентов очень популярна игра в спортивный "Что? Где? Когда?". Смысл игры заключается в том, что несколько соревнующихся команд последовательно отвечают на различные вопросы, используя логику, эрудицию, а зачастую и просто интуицию. Побеждает команда, давшая максимальное количество правильных ответов.

Студенческая команда, проигравшая важнейшую игру сезона, решила разобраться, что послужило причиной такого провала? Капитан команды резонно предположил, что главная ошибка — неправильный выбор стартового состава на игру и замен в процессе игры.

В команде играют N человек. При этом за игровым столом при обсуждении вопроса всегда должно находиться ровно K игроков. После каждого вопроса команда может либо оставить состав игроков за столом неизменным, либо сделать ровно одну замену — убрать из-за стола одного игрока, и посадить туда игрока, который не был за столом во время обсуждения предыдущего вопроса.

Известно, что команда гарантированно отвечает правильно на вопрос только в том случае, если все игроки за столом имеют правильные версии. Для каждого игрока известно, на какие вопросы у него есть правильная версия ответа, а на какие — нет. Вам необходимо определить такой стартовый состав на игру и последовательность замен, при которых команда гарантированно бы ответила правильно на все Q вопросов в игре.

Input

В первой строке задано общее число N ($1 \leq N \leq 100$) игроков в команде, число игроков за столом K ($1 \leq K \leq N$) и количество вопросов Q ($1 \leq Q \leq 1000$).

В следующих N строках задаются описания игроков. Каждое описание содержит строку S_i ($|S_i| = Q$) из нулей и единиц, такую что если $S_{i,j} = 1$, у i -го игрока есть правильная версия ответа на j -й вопрос, а если $S_{i,j} = 0$ — правильной версии нет.

Output

Если есть решение, гарантирующее правильные ответы команды на все Q вопросов, выведите в первой строке WIN.

Во второй строке выведите последовательность A ($1 \leq A_i \leq N$, $|A| = K$), состоящую из K номеров игроков, — стартовый состав команды, находящийся за столом при ответе на первый вопрос.

В третьей строке выведите количество замен Z .

В следующих Z строках в хронологическом порядке выведите описания производимых замен в формате $T_i X_i Y_i$, где T_i ($1 \leq T_i \leq Q - 1$, $T_{i+1} > T_i$) — вопрос, после ответа на который производится замена; X_i — номер игрока, уходящего из-за стола; Y_i — номер игрока, садящегося за игровой стол.

Если существует несколько решений, выведите любое.

Если решений не существует, выведите единственное слово FAIL.

Examples

standard input	standard output
4 3 7 1110111 0111111 1011111 1111100	WIN 1 3 4 3 1 3 2 2 1 3 5 4 1
2 2 4 1100 1010	FAIL

Problem K. Энергия Солнца (Added for Division 1!)

Input file: *standard input*
Output file: *standard output*
Time limit: 1 секунда
Memory limit: 128 мегабайт

В последние годы всё более популярными становятся двигатели, работающие на солнечных батареях. В ближайшие дни даже планируется организовать полёт самолёта вокруг земного шара с использованием только солнечной энергии!

Подготовка кругосветного перелёта самолёта на солнечных батареях включает в себя ряд испытательных полётов. Во время каждого испытательного полёта самолёт стартует из некоторого аэродрома A_i и по кратчайшему маршруту летит к аэродрому A_{i+1} . Таким образом, за исключением первого полёта самолёт всегда вылетает из аэродрома, в котором закончился предыдущий испытательный полёт.

Известно, что аэродромы, на которых проводятся испытательные полёты, расположены строго в линию от 1-го до N -го. Для каждой пары соседних аэродромов, согласно метеоусловиям и расстоянию между ними, известна *дельта энергии* $\Delta Q_{i,i+1}$, которая затрачивается (если $\Delta Q_{i,i+1} < 0$) или приобретается (если $\Delta Q_{i,i+1} \geq 0$) самолётом при перелёте между ними. При перелёте между любой парой аэродромов u и v , не являющихся соседними, дельта энергии определяется по формуле:

$$\Delta Q_{u,v} = \sum_{i=\min(u,v)}^{\max(u,v)-1} \Delta Q_{i,i+1}.$$

Вам необходимо найти любую последовательность A из K номеров различных аэродромов такую, что сумма модулей дельт энергии S при перелётах между аэродромами в порядке, заданном этой последовательностью, была наименьшей.

Input

В первой строке задано общее число N ($2 \leq N \leq 10^5$) аэродромов и длина K ($2 \leq K \leq N$) искомой последовательности аэродромов.

Во второй строке заданы $N - 1$ целых чисел $\Delta Q_{1,2}, \Delta Q_{2,3}, \dots, \Delta Q_{n-1,n}$ — дельты энергии при перелёте между парами соседних аэродромов ($|\Delta Q_{i,i+1}| \leq 10^6$).

Output

В первой строке выведите минимально возможную сумму S .

Во второй строке выведите K чисел через пробел — искомую последовательность A номеров аэродромов (нумерация с 1).

Если решений существует несколько, выведите любое.

Examples

standard input	standard output
4 3 -3 2 10	3 2 3 1
6 4 -3 2 1 -2 -1	2 2 6 5 3