

Problem A. Associated Vertices

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 256 mebibytes

Будем называть вершины a и b в ориентированном мультиграфе *ассоциированными*, если обе эти вершины достижимы из некоторой вершины c .

Ваша задача — по заданному ориентированному мультиграфу A посчитать количество различных пар (i, j) , где вершины i и j ассоциированы.

Input

Первая строка входа содержит два целых числа N и M ($1 \leq N \leq 10^4$, $0 \leq M \leq 10^4$) — количество вершин и рёбер в мультиграфе. Каждая из последующих M строк содержит два целых числа x и y и задаёт ребро из вершины x в вершину y ($1 \leq x, y \leq N$).

Output

Выведите одно целое число — количество различных пар (i, j) , где вершины i и j ассоциированы.

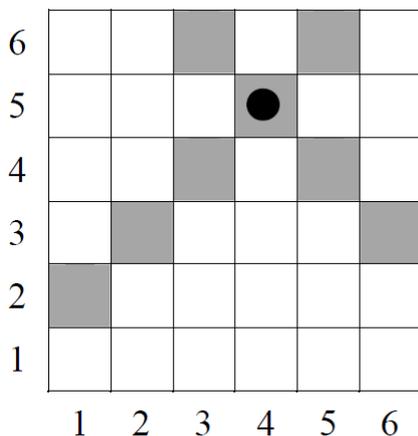
Examples

standard input	standard output
2 1 1 2	4
3 4 2 1 3 1 2 1 3 3	7

Problem B. Bishops

Input file: *standard input*
Output file: *standard output*
Time limit: 1 second
Memory limit: 256 mebibytes

Шахматный слон — это фигура, которая бьёт все клетки на шахматной доске, которые находятся на одной диагонали с ней в обоих диагональных направлениях.



Степан расставил M слонов на шахматной доске $N \times N$ и хочет найти количество полей, которые не бьются ни одним слоном.

Помогите ему их подсчитать.

Input

Первая строка входа содержит два целых числа N ($1 \leq N \leq 10^6$) и M ($1 \leq M \leq 10^5$) — размерность шахматной доски и количество слонов.

i -я из последующих M строк содержит два целых числа r_i и c_i — номера строки и столбца для i -го слона ($1 \leq r_i, c_i \leq N$). Гарантируется, что никакие два слона не стоят на одном и том же поле.

Output

Выведите одно целое число — количество полей, которые не бьются ни одним слоном.

Example

standard input	standard output
10 6	33
4 7	
8 5	
8 7	
6 2	
9 7	
8 4	

Problem C. Cool Numbers

Input file: *standard input*
Output file: *standard output*
Time limit: 0.5 seconds
Memory limit: 256 mebibytes

Степан называет целое положительное число p *весёлым*, если p и число p_1 , полученное прочтением его десятичной записи справа налево, являются различными простыми числами.

Напоминаем, что целое положительное число является простым, если оно не имеет целых делителей кроме единицы и самого себя.

По заданному K найдите K -е весёлое число.

Input

Первая строка входа содержит одно целое число K ($1 \leq K \leq 1000$).

Output

Если K -е весёлое число не превосходит 10^6 , выведите его. Иначе выведите -1 .

Example

	standard input	standard output
1	1	13

Problem D. Diagram

Input file: *standard input*
Output file: *standard output*
Time limit: 0.5 seconds
Memory limit: 256 mebibytes

Дамблдор только что завершил сложный магический ритуал, для которого, помимо прочего, использовалась магическая диаграмма с N попарно различными символами, каждый из которых находится в определённой точке окружности.

Пока Дамблдор спал, Гарри Поттер нашёл диаграмму и хочет использовать её для того, чтобы выполнить домашнее задание по заклинаниям. Но для этого ему нужен правильный K -угольник.

Гарри знает, что длина окружности — целое число, делящееся на K . Также он знает расстояния между соседними по окружности символами — также целые числа. Он хочет выбрать K символов так, чтобы они были вершинами правильного K -угольника.

Напишите программу, которая определит, сможет ли Гарри это сделать.

Input

Первая строка входе содержит целые числа N и K ($3 \leq N \leq 10^5$, $3 \leq K \leq 10^5$), за которыми следует монотонно возрастающая последовательность из $N + 1$ целых чисел X_i , задающих расстояния по часовой стрелке от нулевого символа до i -го по дуге окружности ($X_0 = 0$, а X_N — дуга окружности; $0 \leq X_i \leq 10^9$). Гарантируется, что $X_N \bmod K = 0$. Соседние числа во вводе разделены пробелами.

Output

Если Гарри сможет выбрать K символов, находящихся в вершинах правильного K -угольника, выведите 1. Иначе выведите 0.

Example

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

Problem E. Effective Hiring

Input file: *standard input*
Output file: *standard output*
Time limit: 1 second
Memory limit: 256 mebibytes

Степан собирается стать фермером. Он купил K уборочных комбайнов и собирается начать бизнес. Но комбайны без экипажа работать не начнут.

Экипаж комбайна состоит из двух человек — комбайнёр и помощник комбайнёра. Таким образом, Степан должен нанять ровно K экипажей — K комбайнёров и K помощников.

Неделю назад Степан дал объявление в местную газету, после чего он получил N резюме. Каждое резюме содержит следующую информацию: Few days ago Stefan posted these vacancies to a local newspaper. Today he received N different resumes of applicants. i -th of them contains following data:

1. Место, которое соискатель готов занять. A_i — целое число от 1 до 3. Если $A_i = 1$, то соискатель готов работать только комбайнёром, если $A_i = 2$ — только помощником. Наконец, если $A = 3$, то соискатель готов работать как комбайнёром, так и помощником.
2. Неотрицательное целое число C_i — стаж работы соискателя в часах.
3. Целое положительное число S_i — сумма контракта, которую требует соискатель.

Моральные принципы Степана и трудовой кодекс запрещают нанимать в один экипаж помощника с большим стажем, чем стаж комбайнёра. Иначе говоря, в каждом экипаже стаж комбайнёра не должен быть меньше стажа помощника. Это ограничение действует только внутри одного экипажа, то есть возможна ситуация, когда помощник в первом экипаже имеет стаж больше, чем комбайнёр во втором.

По данным N резюме требуется найти минимальную сумму контрактов, которую надо будет выплатить экипажам K комбайнов.

Input

Первая строка входа содержит два целых числа — N и K ($2 \leq N$; $1 \leq K$; $2 \cdot K \leq N$; $2 \leq N \cdot K \leq 10^5$).

Каждая из последующих N строк задаёт одно резюме. i -я из этих строк задаёт i -е резюме и состоит из трёх целых чисел A_i , C_i и S_i ($1 \leq A_i \leq 3$; $0 \leq C_i \leq 32767$; $1 \leq S_i \leq 32767$).

Output

Выведите минимальную сумму контрактов для K экипажей, набранных в соответствии с правилами. Гарантируется, что данные подобраны так, что решение всегда существует.

Example

standard input	standard output
3 1 2 2 3 1 1 2 3 1 2	4
4 1 3 0 7 1 0 10 3 0 9 2 0 5	12
6 2 1 20 6 2 6 7 3 4 8 2 3 10 3 8 5 1 4 3	22

Note

В первом примере выбираем пару комбайнёра и помощника (2,3), во втором — (1,4), в третьем — пары (1,5) и (6,3).

Problem F. First And Last

Input file: *standard input*
Output file: *standard output*
Time limit: 0.5 seconds
Memory limit: 256 mebibytes

По заданной ненулевой десятичной цифре a и десятичной цифре b найдите, существует ли такое неотрицательное n , что a является первой цифрой числа 2^n , а b — последней. Если существует, выведите наименьшее n с таким свойством.

Input

Вход состоит из двух целых чисел a и b ($1 \leq a \leq 9$, $0 \leq b \leq 9$) — заданных цифр.

Output

Если существует такое неотрицательное целое n , что первая цифра 2^n равна a , а вторая — b , выведите минимальное значение n , при котором это выполняется. Иначе выведите -1 .

Example

standard input	standard output
2 2	1
5 5	-1

Problem G. Game of Solitaire

Input file: *standard input*
Output file: *standard output*
Time limit: 0.5 seconds
Memory limit: 256 megabytes

Рассмотрим следующую разновидность пасьянса. Задана колода из N карт, пронумерованных последовательными целыми числами от 1 до N . Числа написаны на лицевой стороне карт. Карты лежат в ряд лицом вверх, карта 1 левее всех, карта N — правее всех.

Игрок выбирает целое положительное число $K < N$. После этого он перемещает первые K карт в конец ряда.

Например, если $N = 6$ и игрок выбрал число 4 ($K = 4$), расстановка после перемещения будет следующей: 5 6 1 2 3 4.

Далее игрок делает следующее. Он берёт самую левую карту, переворачивает её лицом вниз; пусть на этой карте написано число M ; тогда он переходит к M -й слева карте (включая и те, которые уже перевернуты) и повторяет ту же самую процедуру — переворачивает и переходит к карте, заданной номером на только что перевернутой. Как только игрок дошёл до карты, которая уже была перевернута, он заканчивает первый раунд. После этого игрок начинает следующий раунд: берёт самую левую неперевернутую карту, переворачивает её, переходит... и так далее, пока все карты не будут перевернуты.

По заданным N и M определите количество раундов, требуемое для завершения пасьянса.

Input

Вход содержит два целых числа N и K , $1 \leq K < N \leq 10^9$.

Output

Выведите количество раундов, требуемое для завершения пасьянса.

Example

standard input	standard output
6 4	2

Problem H. Hero's Quest

Input file: *standard input*
Output file: *standard output*
Time limit: 1 second
Memory limit: 256 mebibytes

Герой прибыл в Questland. На этот раз его задача — соединять города дорогами с двусторонним движением с помощью магии. Первоначально никакие два города не соединены дорогой. В начале Quest-а героя телепортируют в первый город. В каждом городе герой делает следующие два действия:

- пытается применить заклинание «строим дорогу»; вероятность срабатывания заклинания зависит от города, где находится герой, и в i -м городе равна p_i . Если заклинание применено успешно, из тьмы появляется джинн, соединяет *два случайных различных города* дорогой с двусторонним движением (вероятность выбора любой пары одинакова; если дорога уже есть, джинн строит ещё одну), после чего уходит назад во тьму.
- Если после первого действия существует пара городов, в один из которых нельзя пройти из другого по цепи дорог, герой случайно телепортируется в один из s_i городов (Список s_i зависит также только от города i , все города в списке имеют равную вероятность). Телепортация занимает одну минуту. В противном случае цель достигнута, Quest пройден и герой немедленно телепортируется из Questland.

Герой хочет узнать математическое ожидание времени в минутах, необходимого, чтобы достигнуть цели и пройти Quest. Помогите ему сделать это.

Input

Первая строка входа содержит целое число n — количество городов ($1 \leq n \leq 20$). Далее следуют n чисел, разделённых пробелами; p_i задаёт вероятность срабатывания заклинания в i -м городе ($0 < p_i \leq 1$, $1000 \cdot p_i$ — целое). i -я из n следующих строк начинается с целого числа s_i — количества городов, в которые переносится герой из i -го города, за которым идут s_i целых чисел c_{ij} — номера этих городов ($1 \leq s_i \leq n$, $1 \leq c_{ij} \leq n$).

Output

Выведите матожидание времени в минутах, которое потребуется герою для прохождения Quest-а, с абсолютной погрешностью не хуже 10^{-6} .

Example

standard input	standard output
3 1 1 1 2 2 3 2 1 3 2 1 2	2.5
3 0.75 0.5 0.25 3 1 2 3 2 2 3 1 1	4.26666667

Problem I. Important Or Not?

Input file: *standard input*
Output file: *standard output*
Time limit: 6 seconds
Memory limit: 512 mebibytes

Задана строка A . Обозначим подстроку $A[p - x + 1, p - x + 2, \dots, p]$ парой (x, p) . Символы в A нумеруются с нуля. Вы получаете сообщения, относящиеся к этой строке.

Всего есть три типа сообщений:

1. Подстрока (x, p) является важной (отметим, что если (x, p) — важная, это не значит автоматически, что все её подстроки становятся важными).
2. Подстрока (x, p) не является важной.
3. Выписать все различные важные строки, содержащие (x, p) как суффикс, затем отсортировать их по возрастанию длины, затем сообщить длину K -й из них.

Первоначально ни одна подстрока не является важной.

Вы должны обрабатывать полученные сообщения и отвечать на запросы.

Input

Первая строка входа содержит непустую строку A , составленную из строчных латинских букв, $|A| \leq 10^5$.

Вторая строка содержит одно целое число q ($1 \leq q \leq 2 \cdot 10^5$) — количество сообщений. Каждая из последующих q строк описывает одно сообщение и состоит не менее, чем из трёх целых чисел t, x, p ; t — тип сообщения (1, 2 или 3), x и p — параметры подстроки; гарантируется, что (x, p) является подстрокой A . Если t равно трём, то далее в строке следует четвёртое целое число K ($1 \leq K \leq 50$) — номер запрашиваемой важной строки.

Output

Для каждого сообщения типа 3 выведите в отдельной строке длину K -й важной строки, содержащей (x, p) как суффикс. Если таких строк меньше K , выведите -1 .

Example

standard input	standard output
zzzzz 4 2 2 1 1 3 2 3 1 4 1 3 4 4 1	3 -1
abacaba 3 1 2 1 1 2 5 3 1 1 2	-1
abacaba 3 1 2 1 2 2 5 3 1 1 1	-1

Problem J. Joining Powers

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 256 mebibytes

Рассмотрим множество бесконечных последовательностей:

- последовательность #1, обозначаемая $S(1)$, есть $1, 2, 3, \dots, n, \dots$;
- последовательность #2, обозначаемая $S(2)$, есть $1, 4, 9, \dots, n^2, \dots$;
- последовательность #3, обозначаемая $S(3)$, есть $1, 8, 27, \dots, n^3, \dots$;
- и так далее
- последовательность # k , обозначаемая $S(k)$, есть $1, 2^k, 3^k, \dots, n^k, \dots$;
- и так далее

Очевидно, что каждая из этих последовательностей монотонно возрастает.

Определим *объединение* $S(i_1, i_2, \dots, i_m)$ последовательностей $S(i_1), S(i_2), \dots, S(i_m)$ как последовательность, для которой:

- каждый элемент каждой последовательности $S(i_1), S(i_2), \dots, S(i_m)$ принадлежит $S(i_1, i_2, \dots, i_m)$;
- каждый элемент, принадлежащий более, чем одной из последовательностей $S(i_1), S(i_2), \dots, S(i_m)$, входит $S(i_1, i_2, \dots, i_m)$ ровно один раз;
- последовательность $S(i_1, i_2, \dots, i_m)$ является монотонно возрастающей.

Например, $S(2, 3, 5)$ начинается как $1, 4, 8, 9, 16, 25, 27, 32, 36, 49, 64, 81, 100, 121, 125, \dots$

Напишите программу, которая будет выполнять запросы вида “найдите N -й элемент $S(i_1, i_2, \dots, i_m)$ ”, где $N, m, i_1, i_2, \dots, i_m$ заданы.

Input

Первая строка входа содержит одно целое число — количество запросов q ($1 \leq q \leq 987$). Далее задаются q запросов. Каждый запрос задаётся двумя строками. Первая строка описания запроса содержит два целых числа N и m , где N ($1 \leq N \leq 10^9$) — номер элемента (начиная с 1), который нужно найти, а m ($1 \leq m \leq 42$) — количество объединяемых последовательностей. Вторая строка содержит m попарно различных целых чисел i_1, i_2, \dots, i_m ($1 \leq i_k \leq 50$).

Output

Для каждого запроса выведите ответ на него. Гарантируется, что ответ не будет превышать 10^{17} .

Example

standard input	standard output
2	81
12 3	38416
2 3 5	
17 2	
4 7	

Problem K. Keyboard Map

Input file: *standard input*
Output file: *standard output*
Time limit: 5 seconds
Memory limit: 256 mebibytes

Задано сообщение, записанное N -буквенным алфавитом. Сообщение содержит все буквы алфавита, при этом первую букву содержит f_1 раз, вторую — f_2 раз, и так далее, N -ю — f_N раз.

Сообщение должно быть набрано на клавиатуре с M ($M < N$) клавишами, используя метод, применявшийся в кнопочных сотовых телефонах.

Напомним, что метод заключался в следующем: буквы 'a', 'b' и 'c' были назначены на физическую кнопку '2'; буквы 'd', 'e' и 'f' — на кнопку '3', и так далее. Набор буквы 'a' требует нажатия '2' один раз, набор 'b' — дважды, и набор 'c' — трижды.

Чтобы набрать буквы 'b' и 'a' подряд, надо было нажать кнопку '2' дважды подряд, подождать где-то одну секунду и нажать ту же самую кнопку ещё один раз.

В нашем случае Вы сами выбираете раскладку: символы от первого до K_1 -го должны быть на кнопке с номером 1, от $(K_1 + 1)$ -го до K_2 -го — на кнопке номер два и так далее, до $K_M = N$.

Вы должны выбрать K_1, K_2, \dots, K_{M-1} так, чтобы минимизировать количество нажатий кнопок, необходимое для набора заданного сообщения.

Input

Первая строка входа содержит два целых числа N и M . ($3 \leq N \leq 5000$, $2 \leq M \leq 3000$, $N > M$) — размер алфавита и количество кнопок на клавиатуре, соответственно.

Вторая строка содержит N целых чисел f_1, f_2, \dots, f_N , где f_i задаёт количество i -й по алфавиту буквы в заданном сообщении ($1 \leq f_i \leq 1000$).

Гарантируется, что для любого распределения алфавита по кнопкам (даже не для оптимального) общее количество нажатий, требующихся для набора заданного сообщения, строго меньше 2^{31} .

Output

Выведите одно число — наименьшее количество нажатий на кнопки, необходимое, чтобы набрать сообщение.

Example

standard input	standard output
5 3 3 2 5 7 1	21

Note

Ответ 21 может быть получен при $K_1 = 2$, $K_2 = 3$ (первые два символа назначены на первую кнопку, третий — на вторую, четвёртый и пятый — на третью. В этом случае имеем $3 \times 1 + 2 \times 2 + (5 \times 1) + (7 \times 1 + 1 \times 2) = 21$ нажатий.

Problem L. Light Sources

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 256 mebibytes

На плоскости заданы n точечных источников света; каждый источник светится одним из k возможных цветов. Также заданы m несветящихся точек; гарантируется, что ни один источник света не лежит на отрезке, соединяющем несветящиеся точки.

Требуется выяснить, возможно ли соединить некоторые из несветящихся точек отрезками так, чтобы получился ровно один простой многоугольник (то есть многоугольник ненулевой площади без самопересечений и самокасаний), который содержит внутри источники света всех k цветов.

Если это возможно, требуется вывести минимальный периметр соответствующего многоугольника.

Input

Первая строка входа содержит три целых числа n , m и k ($1 \leq n \leq 300$, $1 \leq m \leq 40$, $1 \leq k \leq 6$). Каждая из последующих n строк содержит два целых числа x_i и y_i — координаты i -го источника света. Следующая строка содержит n целых чисел c_i ; i -е из этих чисел задаёт цвет i -го источника света ($1 \leq c_i \leq k$). Каждая из последующих m строк содержит координаты несветящейся точки в аналогичном формате. Все координаты не превосходят по модулю 23332. Гарантируется, что ни один источник света не лежит на отрезке, соединяющем две несветящиеся точки и что все источники света находятся в попарно различных точках.

Output

Если можно соединить несколько несветящихся точек так, чтобы внутри получившегося простого многоугольника были источники цвета всех k цветов, выведите минимальный периметр такого многоугольника с абсолютной или относительной погрешностью не хуже 10^{-6} . Иначе выведите -1 .

Example

standard input	standard output
2 4 2 1 1 3 1 1 2 2 0 2 2 -2 0 6 2	16.944271909999
2 4 2 1 1 3 1 2 2 2 0 2 2 -2 0 6 2	-1

Problem M. Merging

Input file: *standard input*
Output file: *standard output*
Time limit: 1 second
Memory limit: 256 mebibytes

Рассмотрим множество бесконечных последовательностей, каждая из которых строится подстановкой $n = 1$, $n = 2$, $n = 3$ и так далее в полином

$$a_7n^7 + a_6n^6 + a_5n^5 + a_4n^4 + a_3n^3 + a_2n^2 + a_1n + a_0$$

Все коэффициенты $a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0$ — целые числа в диапазоне $0 \leq a_i \leq 1000$, и как минимум два из них не равны нулю. Очевидно, что при этих ограничениях каждая последовательность является монотонно возрастающей.

Скажем, что последовательность является *слиянием* заданных последовательностей, если:

- полученная последовательность содержит все элементы всех заданных последовательностей;
- полученная последовательность может содержать одно и то же число несколько раз;
- каждое число входит в последовательность столько раз, сколько раз оно входит во все заданные последовательности в сумме;
- полученная последовательность является неубывающей.

Напишите программу, которая находит N -й элемент слияния заданных последовательностей.

Input

Первая строка входа содержит целое число k ($1 \leq k \leq 3 \cdot 10^4$) — количество заданных последовательностей. Далее следуют k строк, каждая из этих строк содержит по 8 целых чисел $a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0$ — коэффициенты полинома ($0 \leq a_i \leq 1000$, как минимум два из a_i не равны 0). Последняя строка содержит одно целое число N ($1 \leq N \leq 10^5$) — требуемый номер (начиная с нуля).

Output

Выведите одно целое число — значение N -го элемента слияния заданных последовательностей. Гарантируется, что ответ не будет превосходить 10^{17} .

Example

standard input	standard output
3	51
0 0 0 0 1 2 0 0	
0 0 0 0 0 0 10 6	
0 0 0 0 0 0 25 1	
9	

Note

Рассмотрим три последовательности из примера.

Для первой вектор a_i равен $(0, 0, 0, 0, 1, 2, 0, 0)$, полином — $n^3 + 2 \cdot n^2$, а значения — $3, 16, 45, 96, 175, \dots$

Для второй вектор a_i равен $(0, 0, 0, 0, 0, 0, 10, 6)$, полином — $10n + 6$, а значения — $16, 26, 36, 46, 56, \dots$

Для третьей вектор a_i равен $(0, 0, 0, 0, 0, 0, 25, 1)$, полином — $25n + 1$, а значения — $26, 51, 76, 101, 126, \dots$

Таким образом, их слияние имеет вид $3, 16, 16, 26, 26, 36, 45, 46, 51, 56, \dots$