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 ($1 \le N \le 10^4$, $0 \le M \le 10^4$) — 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 ($1 \le x, y \le N$).

Output

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

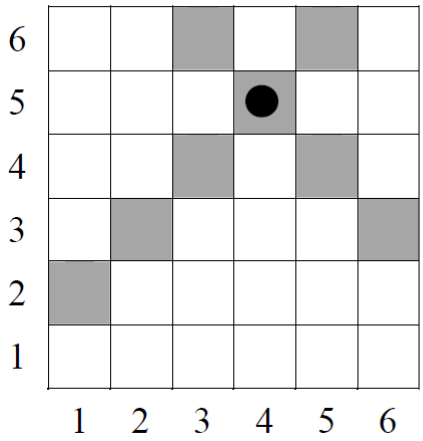
Examples

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 ($1 \le N \le 10^6$) и M ($1 \le M \le 10^5$) — 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 ($1 \le r_i, c_i \le N$). 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 4 7 8 5 8 7 6 2 9 7 8 4	33

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 ($1 \le K \le 1000$).

Output

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

Example

standard input	standard output
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 ($3 \le N \le 10^5$, $3 \le K \le 10^5$), 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 окружностиu; $0 \le X_i \le 10^9$). Гарантируется, что $X_N \mod K = 0$. 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 ($2 \le N$; $1 \le K$; $2 \cdot K \le N$; $2 \le N \cdot K \le 10^5$).

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 ($1 \le A_i \le 3$; $0 \le C_i \le 32767$; $1 \le S_i \le 32767$).

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 ($1 \le a \le 9$, $0 \le b \le 9$) — 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. T 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 перевёрнуты) и повторяет ту же самую процедуру — переворачивает и переходит к карте, заданной номером на только что перевёрнутой. Как только игрок дошёл до карты, которая уже была перевёрнута, он заканчивает первый раунд. После этого игрок начинает следующий раунд: берёт самую левую неперевёрнутую карту, переворачивает её, переходит... и так далее, пока все карты не будут перевёрнуты.

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

Input

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

Output

Выведите количество раундов, требуемое для завершения пасьянса.

Example

standard input	standard output
6 4	2

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

Очевидно, что каждая из этих последовательностей монотонно возрастает.

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

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

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

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

Input

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

Output

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

Example

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

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 $K_1, K_2, \ldots, K_{M-1}$ так, чтобы минимизировать количество нажатий кнопок, необходимое для набора заданного сообщения.

Input

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

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

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 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 $0 \le a_i \le 1000$, 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 ($1 \le k \le 3 \cdot 10^4$) — 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 ($0 \le a_i \le 1000$, как минимум два из a_i не равны 0). Последняя строка содержит одно целое число N ($1 \le N \le 10^5$) — 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 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	51

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 — $3, 16, 45, 96, 175, \ldots$.

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

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

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