Problem A. Associated Vertices

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

Lets say that in a directed multigraph two vertices a and b are *associated*, if both a and b can be reached from some vertex c.

Your task is for a given directed multigraph A to count number of distinct pairs of vertices (i, j), which are associated.

Input

First line of the input contains two integers N and M ($1 \le N \le 10^4$, $0 \le M \le 10^4$ — numbers of vertices and edges in the multigraph. Each of next M lines contains two integers x and y, meaning that here is edge from vertex x to vertex y ($1 \le x, y \le N$).

Output

Print one integer — total number of distinct pairs of associated vertices.

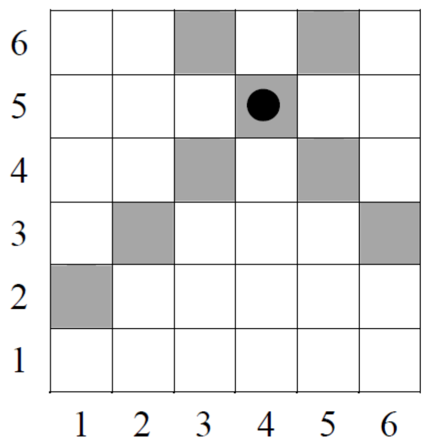
Examples

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

In chess, a bishop is a piece, which attacks all fields of the chessboard, which are placed at same diagonal with it (in both diagonal directions).



On the chessboard $N \times N$ Stefan placed M bishops. Now he wants to calculate number of fields, which are *not* attacked by any bishop.

Help Stefan to do it.

Input

First line of the input contains two integers N ($1 \leq N \leq 10^6$) and M ($1 \leq M \leq 10^5$) — dimension of the chessboard and number of bishops, respectively. Then M lines follow. Each of those M lines contains two space-separated integers r_i and c_i — 1-based numbers of row and column for i'th bishop ($1 \leq r_i, c_i \leq N$). No two bishops share the same field.

Output

Print one integer — number of fields, which are not attacked by a bishop.

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

Stefan calls an positive integer p *cool*, if p and number p_1, obtained by reversing its decimal notation, i.e. reading all digits from end to beginning, are distinct primes.

Remind that positive integer is prime, if it has no positive integer divisors except for 1 and itself.

Given K, find K-th cool number.

Input

First line of the input contains one integer K ($1 \leq K \leq 1000$).

Output

If K-th cool number is not greater than 10^6, print it. Otherwise print -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

Prof. Dumbledore has just completed his rather complicated ritual, which (among other things) involved a diagram with N different symbols drawn at specific positions on a circle.

Now, that he sleeps off his success, Harry Potter wants to reuse the diagram for his own, simpler, ritual. All he needs is a regular K-gon.

Harry knows for certain, that the circumference of a circle is an integer multiple of K. He also knows exact distances between symbols, along the circumference, all of which happen to be integers. He wants to find out, whether he can select K different symbols to become the vertices of his K-gon.

Your task is to write a program to determine just that.

Input

First line of the input contains integers N and K ($3 \le N \le 10^5, 3 \le K \le 10^5$), followed by a monotonically increasing sequence of $N + 1$ integers X_i, denoting clockwise distances from 0-th symbol to i-th along the circumference ($X_0 = 0$ and X_N is the circumference of the circle; $0 \le X_i \le 10^9$). It is guaranteed, that $X_N \mod K = 0$. All numbers are separated by whitespaces.

Output

Print a single integer: 1 if Harry can succesfully start ritual with selecting K different symbols to become the vertices of regular polygon and 0, if his attempt will fail.

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

Stefan wants to become a farmer. He bought K harvesters and plant to make big money. But just harvesters is not enough, Stefan needs personnel he does not have. And he wants to select them effectively.

Harvester crew consists of 2 people — harvester operator and his assistant, operator is the boss, assistant is his subordinate. So Stefan needs to hire exactly K crews, that is K operators and K assistants.

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. Position, the applicant applies to. A_i — integer from 1 to 3. If $A_i = 1$, this is an application for operator position; if $A_i = 2$, this is an application for assistant position; finally, if $A_i = 3$, the applicant is ready to work both as operator or assistant.
2. Non-negative integer C_i — work experience of the appplicant in hours. The more C_i is, the bigger experience is.
3. Positive integer S_i — salary, which this applicant wants to get.

Moral principle and labor code prohibits hiring of subordinate with bigger experience than his boss has. In other words, experience of operator should be not less than experience of his assistant. However, this rule puts restrictions on each separate crew, that is there could be two different crews where operator in one crew has less experience than assistant in the other.

Your task is following: given N resumes calculate minimal amount of money, necessary to hire personnel for K harvesters.

Input

The first line of input contains two integers — N and K ($2 \le N$; $1 \le K$; $2 \cdot K \le N$; $2 \le N \cdot K \le 10^5$).

Each of following N lines describes one resume. i-th resume is described by three integers A_i, C_i and S_i ($1 \le A_i \le 3$; $0 \le C_i \le 32767$; $1 \le S_i \le 32767$).

Output

One integer – minimal amount of money, necessary to hire personnel for K harvesters. It can be assumed that solution exists.

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

In the first sample pair (2,3) gives optimal amount, in the second — pair (1,4), in the third — (1,5) and (6,3).

Problem F. First And Last

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

By non-zero decimal digit a and decimal digit b find if it is possible to find non-negative integer n, such that a is the first digit of 2^n in decimal notation, and b is the last one, and, if it's possible, print such minimal n.

Input

Input file consists of two integers a and b ($1 \le a \le 9$, $0 \le b \le 9$) - the given digits.

Output

Print minimal n, which is an answer for the task, otherwise print -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

We propose the following game of solitaire. There is a deck of N cards which are numbered 1 to N. The numbers are written on the front (face) side of the cards. The cards are laid out in a row face up, card 1 being the one to the far left and card N being the one to the far right. Then the player picks a positive integer K so that it is less than N. He then shifts the first K cards to the end of the row. For example, if there were 6 cards ($N = 6$) and the player picked number 4 ($K = 4$), the final arrangement would be like this: 5 6 1 2 3 4.

Once this step is completed, the player performs the following. He takes the leftmost card, turns it over (so it's face down now); then, if the number written on this card is M, he proceeds to the card at the position M (that is, the M-th leftmost card, including cards lying face down) and repeats the very same procedure: turns it over and proceeds to the card at the position determined by this last card. The player goes on until he proceeds to a card which has already been turned over once before. That makes a round. The player then takes the leftmost card that still lies face up and starts off another round. After several rounds, all cards will be lying face down. Your task is to determine the number of rounds the player will perform knowing the number of cards in the deck (N) and the initial number chosen by the player (K).

Input

The input contains two integers, N and K, $1 \le K < N \le 10^9$.

Output

Output the total number of rounds of the solitaire game.

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

Hero arrives in the Questland. His goal is to connect cities with two-way roads. At the beginning, he is teleported to the city 1. In each city, hero does next two steps:

- tries to invoke road building spell; probability of its success in i-th city is p_i. If spell is invoked successfuly, then some magical creature arrives from darkness, connects *random pair* of cities with two-way road (pairs are chosen equiprobably; if cities are already connected with a some roads, creature builds another one) and leaves back.
- If after first step there exists at least one pair of cities, one of which can not be reached from another using existing road network, hero is teleported randomly to one of s_i cities (s_i depends only on current city i, all cities in set have the same probability). The teleportaton process, however, takes one minute. Otherwise goal is reached and hero is immediately teleported outside of the Questland.

Initially all cities in Questland are disconnected. Hero wants to know expected time in minutes to reach his goal. Help him to do it.

Input

First line of the input contains an integer n — number of cities ($1 \le n \le 20$). Then n space-separated numbers follow; p_i denotes the probability to invoke a magic for the i-th city ($0 < p_i \le 1$, $1000 \cdot p_i$ is integer). i'th of next n lines starts with integer s_i, following by s_i integers c_{ij} — cities, to which hero can be teleported from the i-th city ($1 \le s_i \le n$, $1 \le c_{ij} \le n$).

Output

Print the expected time for the hero to reach his goal in minutes with absolute error not greater than 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

Given a string A. Let's denote the substring $A[p-x+1, p-x+2, \ldots, p]$ by pair (x, p). Indices in A are zero-based. You starts to receive messages, related to this string.

There are 3 types of messages:

1. Substring (x, p) is important (note that if substring (x, p) is important, it *does not* mean that its substrings become important automatically).
2. Substring (x, p) is not important.
3. List all the distinct important strings which contain (x, p) as suffix, then sort them by length in ascending order, then tell the length of K-th important string.

Initially all substrings are not important.

You must process received messages and answer the requests.

Input

First line of the input contains string A, composed by lowercase English letters, $|A| \leq 10^5$.

Second line contains one integer q ($1 \leq q \leq 2 \cdot 10^5$) — number of messages. Then q lines follow. Each line contains at least three integers t, x, p; t is type of message (1, 2 or 3), x and p are substring parameters and selected in such a way that (x, p) is substring of A. If t is equal to 3, then fourth integer K ($1 \leq K \leq 50$) — number of the requested important string — follows.

Output

For each message of type 3 print in a separate line length of K-th important string, which contains (x, p) as a suffix. If number of those strings is less than K, print -1 instead.

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

Consider the set of infinite sequences:

- sequence #1, named $S(1)$, is $1, 2, 3, \ldots, n, \ldots$;
- sequence #2, named $S(2)$, is $1, 4, 9, \ldots, n^2, \ldots$;
- sequence #3, named $S(3)$, is $1, 8, 27, \ldots, n^3, \ldots$;
- and so on;
- sequence $\#k$, named $S(k)$, is $1, 2^k, 3^k, \ldots, n^k, \ldots$;
- and so on;

Obviously, each of these sequences is monotonically increasing.

We say that sequence $S(i_1, i_2, \ldots, i_m)$ is a union of sequences $S(i_1), S(i_2), \ldots, S(i_m)$ if:

- each element of each sequence $S(i_1), S(i_2), \ldots, S(i_m)$ belongs to $S(i_1, i_2, \ldots, i_m)$;
- each element, that belongs to several sequences $S(i_1), S(i_2), \ldots, S(i_m)$, belongs to $S(i_1, i_2, \ldots, i_m)$ exactly once;
- sequence $S(i_1, i_2, \ldots, i_m)$ is monotonically increasing.

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

Your task is to write a program which will process a series of queries in the form "find the N-th element of $S(i_1, i_2, \ldots, i_m)$", where $N, m, i_1, i_2, \ldots, i_m$ are input data.

Input

The first line of the input contains single integer — quantity of queries $q (1 \le q \le 987)$. Afterwards, input data contain exactly q queries. Each query takes two lines. The first line of each query contains N and m, where $N (1 \le N \le 10^9)$ is the index (1-based) of the element to be determined, and $m (1 \le m \le 42)$ is the quantity of sequences to be united. The second line of each query contains integers $i_1, i_2, \ldots, i_m$ (all different, all in range $1 \le i_k \le 50$).

Output

The program should output results for all queries, each in a separate line. It's guaranteed that answer does not exceed 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

There is a message written in N-symbol alphabet. The message contains the first symbol of the alphabet f_1 times, the second symbol — f_2 times, etc, the N-th symbol — f_N times.

The message should be typed on M-key ($M < N$) keyboard, using method, similar to the method, typical for old cellular phones.

Remind, that typical for old cellular phones method of typing text messages (such as SMS texts) is: letters 'a', 'b' and 'c' are assigned to the (hardware) key '2'; letters 'd', 'e' and 'f' — to key '3', and so on. Typing letter 'a' requires pressing the key '2' once, typing 'b' — twice, and typing 'c — three times.

To type sequential letters 'b' and 'a', one should press the key '2' twice at a run, wait (approx. 1 sec) for timeout, and press the same key once more.

In our case, the alphabet symbols from the first to K_1-th should be assigned to the key number 1, from $(K_1 + 1)$-th to K_2-th — to the key number 2, and so on, till $K_M = N$. Note, that $K_1, K_2, \ldots, K_{M-1}$ are not given. Your task is to choose $K_1, K_2, \ldots, K_{M-1}$ in such way, that minimizes the total number of key pressings, considering the given quantities of letters $f_1, f_2, \ldots, f_N$ in the message.

Input

The first line of the input contains two integers N and M. ($3 \le N \le 5000$, $2 \le M \le 3000$, $N > M$) — size of the alphabet and number of keys on the keyboard, respectively.

The second line contains N space-separated integers $f_1, f_2, \ldots, f_N$, where f_i denotes quantity of i-th symbols of the given alphabet in the message ($1 \le f_i \le 1000$).

Additionally, it's guaranteed, that for all possible splittings of the alphabet (even for non-optimal splittings) total number of pressing the keys is strictly less than 2^{31}.

Output

Print one integer — minimal total number of pressing the keys.

Example

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

Note

The answer 21 can be reached with $K_1 = 2$, $K_2 = 3$ (the first two symbols are assigned to first key, the third — to second, the fourth and fifth – to third). Then answer is $3\times1+2\times2)+(5\times1)+(7\times1+1\times2) = 21$.

Problem L. Light Sources

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

There are n light sources (points) on a plane; each such a point is glowing with one of k possible colors. Also m solid points are given; it is guaranteed that no light source lie on the segment connecting any two solid points.

Your task is to connect some of the solid points by segments to build exactly one simple polygon (i.e. polygon with nonzero area without self-intersection and degeneration) which contains light sources of all k colors inside in such a way that sum of lengths of all the used segments will be minimal possible, or to determine that its impossible to build at least one such a polygon.

Input

First line of the input contains three integers n, m and k ($1 \le n \le 300$, $1 \le m \le 40$, $1 \le k \le 6$). Then n lines follow, each containing two integers x_i and y_i — coordinates of i-th light source. Next line contains n integers c_i; i'th of them denotes color of i'th light source ($1 \le c_i \le k$). Each of next m lines contains coordinates of the solid point in the same format. Coordinates of all points do not exceed 23332 by absolute value. It is guaranteed that segment, connecting any two of solid points, does not contains light source inside and that all light sources are pairwise distinct.

Output

If it is impossible to connect some solid points and obtain simple polygon, containing light sources of all k colors inside, print -1. Otherwise print minimal perimeter of such a polygon with absolute or relative error not worse than 10^{-6}.

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

Consider the set of infinite sequences, each of which is formed with substitution of $n = 1$, $n = 2$, $n = 3$ etc. to some polynomial

$$a_7 n^7 + a_6 n^6 + a_5 n^5 + a_4 n^4 + a_3 n^3 + a_2 n^2 + a_1 n + a_0$$

All coefficients a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0 are integers in range $0 \le a_i \le 1000$, and at least two of them satisfy additional constraint $a_i \ge 1$. Obviously, due to all these constraints, each of these sequences is monotonically increasing.

We say that sequence is *merging* of given sequences, if:

- resulting sequence contains all elements of all given sequences;
- resulting sequence may contain the same number several times;
- quantity of each number in resulting sequence equals to the sum of quantities of the number in all given sequences;
- resulting sequence is monotonically non-decreasing.

Write a program, which will find the N-th element of merging of given sequences.

Input

The first line of the input contains single integer k ($1 \le k \le 3 \cdot 10^4$) — the number of the sequences to be merged. Then k lines follow, each of those lines contains eight integers $a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0$ — coefficients of the polynomial ($0 \le a_i \le 1000$, atleast two of a_i are not equal to zero). Last line contains one integer N ($1 \le N \le 10^5$) — 1-based index.

Output

Print one integer — value of N-th element of the merging. It is guaranteed that answer will does not exceed 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

Consider three given in the sample sequences:

For the first one, vector a_i is equal to $(0, 0, 0, 0, 1, 2, 0, 0)$, then polynomial is $n^3 + 2 \cdot n^2$, and values are $3, 16, 45, 96, 175, \ldots$.

For second one, vector a_i is equal to $(0, 0, 0, 0, 0, 0, 10, 6)$, then polynomial is $10n + 6$, and values are $16, 26, 36, 46, 56, \ldots$.

For third one, vector a_i is equal to $(0,0,0,0,0,0,25,1)$, then polynomial is $25n+1$, and values are $26, 51, 76, 101, 126, \ldots$.

So, their merging is $3, 16, 16, 26, 26, 36, 45, 46, 51, 56 \ldots$.