Problem A. Another Tree Queries Problem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	1024 mebibytes

You have a tree of N vertices. Vertices are enumerated by sequential integers from 1 to N, and the i-th vertex contains the variable A_i. Initially $A_i = 0$ $(1 \le i \le N)$.

You have to process Q queries. Each query is one of the following:

- "1 u v": Root the tree at vertex u, consider the subtree of vertex v, and for each vertex i in the subtree of v, increase A_i by one.
- "2 u v": For each vertex i in the unique simple path from u to v, increase A_i by one.
- "3 v": Print $\sum_{i=1}^{N} \text{dist}(v, i) \cdot A_i$, where $\text{dist}(x, y)$ is the number of edges in the path from vertex x to vertex y.

Input

The first line contains an integer N, the number of vertices $(1 \le N \le 2 \cdot 10^5)$.

The next $N - 1$ lines contain the description of the tree. Each such line contains two integers u and v separated by a space, meaning that there is an edge connecting u and v $(1 \le u, v \le N)$. It is guaranteed that the resulting graph is a tree.

The next line contains an integer Q, the number of queries $(1 \le Q \le 2 \cdot 10^5)$.

The next Q lines contain queries, one per line. Each query is given in one of the formats described above $(1 \le u, v \le N)$. You may assume that there is at least one query of type "3".

Output

For each query of type "3", print the result on a separate line.

Example

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

Problem B. Best Meeting Places

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

A tree with N vertices is given. Vertices are numbered sequentially from 1 to N. The i-th edge connects vertices A_i and B_i, and has weight C_i, for $1 \le i \le N - 1$.

The *teleport distance* between two vertices of the tree is the maximum weight of the edge on the shortest path connecting them. The teleport distance between a vertex and itself is defined as 0.

People living on the tree want to hold N meetings. The i-th meeting is attended by people living in the vertices numbered from 1 to i. This year, because of the spread of coronavirus, the meeting participants will arrive at X selected locations, and then connect via Internet from these locations.

More formally, for each meeting, we will choose X pairwise distinct vertices v_1, v_2, ..., v_X. Once the vertices are determined, each person will move to one of the vertices v_1, ..., v_X with the minimum teleport distance to it. Let us define the *meeting cost* for the given X and i as the maximum of teleport distances for meeting participants. We will select the vertices v_1, ..., v_X in such a way that the meeting cost is minimal possible.

The value of X depends on the coronavirus situation, and may vary from 1 to K. To prepare for the meeting in advance, write a program that, for each of the N meetings, finds the sum of the meeting costs for all possible values of X from 1 to K, inclusive.

Input

The first line of input contains two integers N and K: the number of vertices and the upper limit for X, respectively ($1 \le K \le N \le 3 \cdot 10^5$).

The following $N - 1$ lines describe the tree. Each of these lines contains three integers, A_i, B_i, and C_i, telling that there is an edge between vertices A_i and B_i with weight C_i ($1 \le A_i, B_i, C_i \le N$). It is guaranteed that the resulting graph is a tree.

Output

Print N lines. On line i, print the sum of meeting costs of i-th meeting for all X from 1 to K, inclusive.

Examples

standard input	standard output
10 4 5 1 2 1 6 4 6 2 1 2 8 9 8 3 5 3 4 8 4 10 9 10 9 8 9 7 7	0 4 13 21 23 23 30 31 33 34
8 3 7 3 4 4 5 2 3 6 1 6 8 6 8 5 1 2 5 8 1 5 2	0 8 14 16 16 16 18 18

Problem C. Colorful Squares

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	1024 mebibytes

There are N points on a plane: P_1 (x_1, y_1), P_2 (x_2, y_2), ..., P_N (x_N, y_N). Each point has a color denoted by an integer from 1 to K.

Consider a square with edges parallel to coordinate axes. The square is *colorful* if it contains points of all K colors inside or on its border. It is allowed for the square to have edge length 0, thus covering just a single point.

Find a colorful square with the minimum side length, and print that length.

Input

The first line contains two integers N and K ($2 \le N \le 10^5$, $2 \le K \le N$).

Then N lines follow. The i-th of these lines contains three integers, x_i, y_i, and c_i: the coordinates of the i-th point and its color, respectively ($1 \le x_i, y_i \le 2.5 \cdot 10^5$, $1 \le c_i \le K$). You may assume that there is at least one point for each of k colors.

Output

Print one integer: the answer to the problem.

Examples

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

Problem D. Designing a PCB

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Dongkyu is trying to design a single-sided printed circuit board (PCB for short). A PCB is composed of pads on which components can be mounted, and conductive tracks connecting the pads. You can think of a PCB as an infinite two-dimensional plane, a pad as a point on the plane, and a track as a connected polyline on the plane.

In the circuit that Dongkyu wants to design, the $2n$ pads are arranged horizontally. The i-th pad from the left is located at coordinate $(i-1, 0)$. Each pad is assigned a label: an integer between 1 and n, inclusive. For each $1 \le i \le n$, there are exactly 2 pads labeled with i.

Dongkyu needs to draw n tracks to connect the pairs of pads with the same label. Each track must be a polyline consisting of segments of positive integer lengths, such that each segment is parallel to one of the coordinate axes. They start and end at the points representing the pads. No two tracks may share a common point.

Given the number of pads and the labeling, write a program to design the circuit.

Input

The first line of input contains a single integer n ($1 \le n \le 1000$).

The second line contains $2n$ integers p_i ($1 \le p_i \le n$). Here, p_i is the label on the i-th pad from the left. It is guaranteed that each integer between 1 and n appears exactly twice in the labels.

Output

If it is impossible to design a circuit conforming to the limitations described in the statement, print "`NO`".

Otherwise, print "`YES`" on the first line. Then, on the next n lines, output the descriptions of n tracks in order of increasing label number of the pads they are connecting.

Each track must be a polyline starting from the **leftmost** of the two connected pads. The description of a track starts with one integer L_i ($1 \le L_i \le 10$) describing the number of segments forming the track. Each segment is described by one letter for the direction, followed by a positive integer for the segment length. The directions are: '`D`' — down (decreasing y), '`U`' — up (increasing y), '`R`' — right (increasing x), and '`L`' — left (decreasing x). The segments must be listed from the starting pad to the ending pad in the order they are connected.

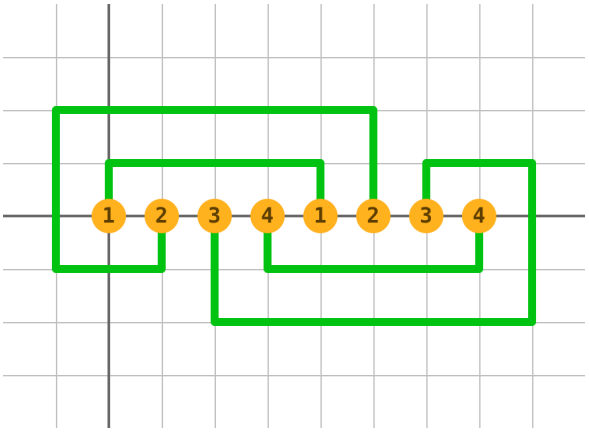
Each polyline must have no self-intersections and no self-touchings. Different polylines must have no common points. The resulting coordinates of the vertices of the polylines must be not greater than 10^4 by absolute value. Separate letters and integers with spaces. Check the sample output to clarify the format.

If there is more than one solution, any one of them will be accepted.

Examples

standard input	standard output
4 1 2 3 4 1 2 3 4	YES 3 U 1 R 4 D 1 5 D 1 L 2 U 3 R 6 D 2 5 D 2 R 6 U 3 L 2 D 1 3 D 1 R 4 U 1
4 1 2 3 4 1 3 2 4	NO

Note



One of the possible circuits for sample 1 is shown at the picture. In sample 2, we cannot connect the pads so that the different tracks do not intersect.

Problem E. Expected Distance

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	1024 mebibytes

Given are two integer sequences: $\{a_i\}$ of length $N-1$ and $\{c_i\}$ of length N. Let us build a tree T_N with N vertices in the following way:

- T_1 is a tree made up of only vertex 1.
- For $i > 1$, T_i connects vertex i to one of the vertices of T_{i-1}. The probability that vertex j will be chosen is $\frac{a_j}{a_1 + \cdots + a_{i-1}}$. The length of the added edge is then calculated as $c_i + c_j$.
- When T_N is built, the process stops.

You are given Q queries. Each query is a pair of vertices. For each query (u, v), calculate the expected distance between u and v in T_N.

Input

The first line of input contains two integers N and Q: the number of vertices and the number of queries, respectively ($2 \le N, Q \le 3 \cdot 10^5$).

The second line contains $N-1$ integers a_1, a_2, ..., a_{N-1} ($1 \le a_i \le 2000$).

The third line contains N integers c_1, c_2, ..., c_N ($1 \le c_i \le 2000$).

Each of the following Q lines describes one query and contains two integers u and v separated by a space: numbers of vertices to find the expected distance ($1 \le u, v \le N$).

Output

It can be proven that each answer is a rational number and can be written in the form $ans_i = \frac{p_i}{q_i}$, where p_i and q_i are coprime non-negative integers and $0 < q_i < 10^9 + 7$. For each query, print the integer $(p_i \cdot q_i^{-1}) \bmod (10^9 + 7)$.

Examples

standard input	standard output
5 7 1 1 1 1 1 2 4 8 16 1 3 2 5 4 3 1 5 3 3 4 5 1 2	7 666666697 15 666666697 0 333333366 3
5 4 17 19 23 29 2 3 5 7 11 1 2 3 4 5 2 3 5	5 927495315 106531441 450222593

Problem F. Find the XOR

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

There is a connected graph G consisting of N vertices and M weighted undirected edges. In G, the weight of the path is obtained by XORing the weights of all edges in the path. Note that it is allowed to walk along one edge multiple times, in this case, you are XORing the weight that number of times.

For vertices u and v, let $d(u, v)$ be the **maximum weight** of a path from u to v.

You need to answer Q queries of the following form:

Given l and r, for all i and j such as $l \le i < j \le r$, find the XOR of the values of $d(i, j)$.

Input

The first line contains three integers, N, M, and Q ($1 \le N, M, Q \le 10^5$).

Each of the following M lines contains three integers, u, v, and w, describing an edge of weight w connecting vertices u and v ($1 \le u, v \le N$, $0 \le w < 2^{30}$). Note that multiple edges and loops are **allowed** in this task.

Each of the following Q lines describes one query and contains two integers l and r ($1 \le l < r \le N$).

Output

For each query, print the answer on a separate line.

Example

standard input	standard output
8 10 7 1 2 662784558 3 2 195868257 3 4 294212653 4 5 299265014 6 5 72652580 6 7 29303370 7 8 183954825 2 1 752722885 5 3 197591314 8 4 877461873 4 8 5 7 6 7 2 3 7 8 3 4 2 7	0 713437792 738051848 716356296 736682272 1003204975 987493236

Problem G. Generate The Array

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Consider an array A of length N. You are planning to do several queries: for a segment $[i, j]$ of the array, find the maximum value on that segment of the array. The query for the indices i and j will be done $Q_{i,j}$ times.

But the array is not given, and you are going to build it right now.

For each i from 1 to N, you can select one of K_i different values $V_{i,j}$ as the value of A_i, and the respective costs of choosing these values are $C_{i,j}$.

After all queries, your *score* will be the sum of the results of all the interval queries you are planning to do, minus the cost of choosing the values A_i. Find the maximum possible score that can be achieved.

Input

First line of the input contains one integer N ($1 \le N \le 300$).

Then N lines follow. The i-th of those lines contains integers from $Q_{i,i}$ to $Q_{i,N}$ ($0 \le Q_{i,j} \le 999$). The query for the maximum element in the array between A_i and A_j inclusively shall be performed exactly $Q_{i,j}$ times.

After that, the input describes possible values of A_i for each i from 1 to N. The i-th description has the following format:

- The first line contains a positive integer K_i: the number of possible values for A_i.
- Each of the following K_i lines contains two integers $V_{i,j}$ and $C_{i,j}$: a possible value and the cost of picking that value, respectively ($0 \le V_{i,j} \le 10^8$, $0 \le C_{i,j} \le 10^{13}$).

It is guaranteed that the sum of K_i is at most $3 \cdot 10^5$.

Output

Print one integer: the maximum possible score.

Examples

standard input	standard output
5 1 0 2 2 0 0 2 2 0 2 2 2 1 2 0 2 0 27 1 19 2 7 25 1 1 2 8 7 4 18 2 8 7 4 4 2 0 25 4 26	78
2 1 1 1 2 1 100 2 50 1 1 100	-145

Problem H. Hotspot-2

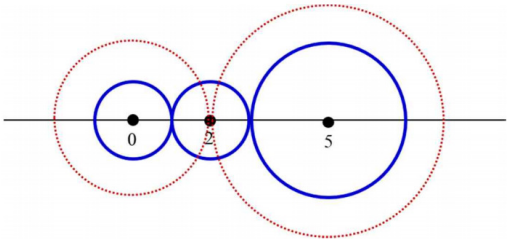
Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

A hotspot is a physical location where people can generally use Wi-Fi to access the Internet through a wireless local-area network (WLAN) with a router connected to an Internet service provider. Most people call these places "Wi-Fi hotspots". Public hotspots are typically created from wireless access points, shortly, APs. Specifically, a hotspot is a zone within distance r from where an AP is installed. In other words, it is a circle with radius r centered at the location of AP.

In a city, there is a long straight road. The APs are already installed along the road. The city officials need to set the radii of hotspots. Then, for any two different APs, the hotspots created from them should not overlap, but at their boundaries, they can meet. As a special case, if the radius of a hotspot is zero, and another hotspot contains it inside, then the two hotspots overlap, and it should not happen. But even if the radius of hotspot is zero, the hotspot can touch a boundary of another hotspot.

The city officials are trying to set the radii of hotspots such that their total coverage area is as large as possible. Thus, they shall maximize the sum of areas of hotspots, simply, the sum of squares of radii of hotspots. To achieve the goal, some radii of hotspots may be set to zero.

The road is considered as a line on the plane, and the locations of APs installed on the road are points on the line. Write a program that, given n points on the line, will determine the radii of non-overlapping hotspots such that the sum of squares of these radii is maximum possible.



For example, there are three APs located at 0, 2, and 5, respectively, in the above figure. As a candidate, the blue and red hotspots are given. The radii of the blue hotspots are 1, 1, and 2, from left to right. Then the sum of squares of radii is 6. But for the red hotspots, their radii are 2, 0, and 3, from left to right. Thus, the sum of squares of radii is 13, which is the maximum.

Input

The first line of input contains an integer n, the number of APs ($2 \le n \le 3 \cdot 10^5$). The second line contains n distinct space-separated integers in strictly increasing order representing the locations of APs on the line, where the integers are between 0 and 10^9, inclusive.

Output

Print one integer: the maximum sum of squares of radii of hotspots.

Examples

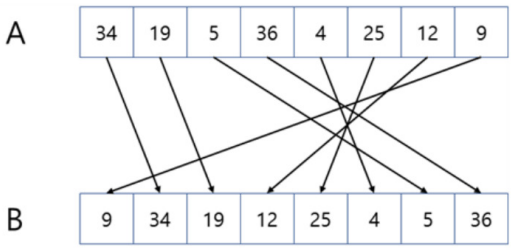
standard input	standard output
3 0 2 5	13
4 0 1 3 6	10
5 5 7 12 13 15	9

Problem I. Integer Array Shuffle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Given an integer array A of size N, the *shuffle* operation is defined as follows.

- Initially, you create an empty integer array B.
- Then, while A is not empty, you remove either the leftmost or rightmost element of A, and append the value to the right in B.
- If A is empty, replace A with B and stop.



If the shuffle operation is performed as shown in the picture above, the value of the array A is changed as follows:

$$(34, 19, 5, 36, 4, 25, 12, 9) \to (9, 34, 19, 12, 25, 4, 5, 36).$$

Let A_i be the i-th element of array A. When the condition "if $1 \le i < j \le N$, then $A_i \le A_j$" is established, it is said that array A increases monotonically.

Write a program that, given an integer array A of size N, calculates the minimum number of shuffle operations required to make the array A monotonically increasing.

Input

The first line of input contains one integer N, the number of elements in array A ($1 \le N \le 3 \cdot 10^5$).

The second line contains N integers $A_1, \ldots, A_N$: the initial values of elements of array A ($1 \le A_i \le 10^9$).

Output

Output the minimum number of shuffle operations required to make the array A monotonically increasing.

Examples

standard input	standard output
3 2 2 5	0
6 1 5 8 10 3 2	1

Problem J. Junkyeom's Contest

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Junkyeom and his friends Myung and Myeong are planning to hold a programming contest with one gold (first place), two silver (places 2 and 3), and four bronze medals (places 4, 5, 6, 7).

The sponsors gave N gift cards for the contest, i-th of them costs A_i. Each medalist shall be awarded exactly one card. Let P_i be the prize for the card awarded to the contestant taking i-th place. The distribution is considered *fair* if the following two inequalities are held:

$$P_1 \geq P_2 \geq P_3 \geq P_4 \geq P_5 \geq P_6 \geq P_7$$

and

$$P_1 < P_2 + P_3 < P_4 + P_5 + P_6 + P_7.$$

Given the values A_i, find out if a fair distribution of prizes exists. If it does, print the maximum possible sum of P_i for a fair distribution.

Input

The first line of input contains one integer N, the number of gift cards ($7 \leq N \leq 5 \cdot 10^5$).

The second line contains N integers A_i: the prizes for the cards ($1 \leq A_i \leq 2 \cdot 10^8$).

Output

If a fair distribution of prizes is impossible, print -1.

Otherwise print one integer: the maximum possible total prize of the fairly distributed gift cards.

Examples

standard input	standard output
7 1 2 3 4 5 6 7	-1
8 1 2 3 4 5 6 7 8	35
10 5 5 5 5 5 5 10 5 5 5	35

Problem K. Knowledge Is...

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

Today, N projects are scheduled to be posted at Songjuk Haksa. Jongyoung and his friends don't want to do a lot of work, so they decided to share the projects and implement them.

The project i can be accessed at time L_i and shall be implemented at time R_i or earlier, but the students' learning ability is not very good, so work on the project consumes all the available time.

In addition, a student cannot solve several projects at the same time, so if a student selected two projects i and j, $R_i < L_j$ or $R_j < L_i$ must be satisfied. Also, students are not very interested in hard work, so each student will work on a maximum of two projects.

A group of M strong students want to collectively implement as many projects as possible. Help them decide which projects each student needs to solve. If there are multiple possible ways, you may choose any one of them.

Input

The first line of input contains two integers N and M, the number of projects and the number of students, respectively ($1 \le M \le N \le 3 \cdot 10^5$).

Each of the following N lines contains two integers L_i and R_i: the start and end time of i-th project ($1 \le L_i < R_i \le 10^9$).

Output

Print N integers. The i-th of those integers must be the number of student (from 1 to M) who will handle project i. If no student will do project i, output 0 instead.

The number of implemented projects must be the maximum possible. If there are several possible solutions, print one any of them.

Examples

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

Problem L. Lights On The Road

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	1024 mebibytes

The *Gyeongin Expressway*, the oldest expressway in South Korea, can be divided into N *blocks* of the same size. Block i ($1 \le i \le N$) has block $i-1$ on the left (except for case when $i = 1$), and block $i+1$ on the right (except for case when $i = N$).

Junwun Kim, a resident of Nambu Beltway 1119-gil, decided to install streetlights on several blocks. For each block i, Junwun Kim decides whether to install a streetlight on block i and pay W_i, or not to install it and pay nothing. After the work is finished, for each block, it must have a streetlight installed, or at least one of its neighbors must have a streetlight installed. The *total cost* of the work is the sum of the cost of installing each streetlight.

Let us consider all ways for Junwun Kim to install streetlights while satisfying the above-mentioned conditions. Two ways are considered different if there is a block i ($1 \le i \le N$) that has a streetlight installed in one of the ways but not in the other. Sort all these ways by total cost in non-descending order. Then, for given K, print the total cost for each of the first K ways in this sorted list. If, for some x such that $1 \le x \le K$, there are less than x ways overall, print -1 for that x instead.

Input

The first line contains two integers N and K, the number of blocks and the number of ways to be printed, respectively ($1 \le N, K \le 2.5 \cdot 10^5$).

The second line contains N integers W_1, W_2, ..., W_N: the costs of installing a streetlight in each block ($0 \le W_i \le 10^9$).

Output

Print K lines. On the i-th of these lines, print the total cost of the i-th way to install streetlights in the sorted list. If the number of ways is less than i, print -1 instead.

Examples

standard input	standard output
5 3 1 3 10 3 1	4 4 5
12 1 317 448 258 208 284 248 315 367 562 500 426 390	1525
12 20 317 448 258 208 284 248 315 367 562 500 426 390	1525 1566 1602 1616 1633 1652 1697 1725 1730 1733 1747 1761 1764 1766 1773 1775 1783 1792 1811 1824
3 9 0 0 0	0 0 0 0 0 -1 -1 -1 -1