Problem A. Determine Robot's Score

Input file:	`robot.in`
Output file:	`robot.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

You built a robot for a Robot Challenge. The challenge is set up in a 100m by 100m space. Certain points are identified within the space as targets. They are ordered — there are target 1, target 2, etc. Your robot must start at $(0, 0)$. From there, it should go to target 1, stop for 1 second, then go to target 2, stop for 1 second, and so on. It must finally end up at $(100, 100)$ and then wait for one second.

Each target except $(0, 0)$ and $(100, 100)$ has a time penalty for missing it. So, if your robot went straight from target 1 to target 3, skipping target 2, it would incur penalty for the second target. Note that once it hits target 3, it cannot go back to target 2. It must hit the targets in order. Since your robot must stop for 1 second on each target point, if you pass the target without stopping there — it does not count. For example, if target 3 lies directly between target points 1 and 2, your robot can go straight from 1 to 2, right over 3, without stopping. Since it didn't stop, the judges will not mistakenly think that it hit target 3 too soon, so they won't count the penalty of the second target. Your final score is the amount of time (in seconds) your robot takes to reach $(100, 100)$, completing the course, plus all penalties. Smaller scores are better.

Your robot is very maneuverable, but a bit slow. It moves at 1 m/s, but can turn very quickly. During the 1 second it stops on a target point, it can easily turn to face the next target point. Thus, it can always move in a straight line between target points. Because your robot is a bit slow, it might be advantageous to skip some targets, and incur their penalty, rather than actually maneuvering to them. Given a description of a course, determine your robot's best (lowest) possible score.

Input

There will be several test cases. Each test case will begin with a line with one integer, N $(1 \le N \le 1000)$ which is the number of targets on the course. Each of the next N lines will describe a target with three integers, X, Y and P , where (X, Y) is a location on the course $(1 \le X, Y \le 99$, X and Y in meters) and P is the penalty incurred if the robot misses that target $(1 \le P \le 100)$. The targets will be given in order —the first line after N is target 1, the next is target 2, and so on. All the targets on a given course will be unique — there will be at most one target point at any location on the course. End of input will be marked by a line with a single 0.

Output

For each test case, output a single decimal number, indicating the smallest possible score for that course. Your answer must have error 10^{-3} or less. Print each answer on its own line, and do not print any blank lines between answers.

Examples

robot.in	robot.out

1
50 50 20
3
30 30 90
60 60 80
10 90 100
3
30 30 90
60 60 80
10 90 10
0
	143.421
	237.716
	154.421

Problem B. Cookie Clicker

Input file:	`cookie.in`
Output file:	`cookie.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Kostya is playing the computer game Cookie Clicker. The goal of this game is to gather cookies. You can get cookies using different *buildings*: you can just click a special field on the screen and get the cookies for the clicks, you can buy a cookie factory, an alchemy lab, a time machine and it all will bring lots and lots of cookies.

At the beginning of the game (time 0), Kostya has 0 cookies and no buildings. He has n available buildings to choose from: the i-th building is worth c_i cookies and when it's built it brings v_i cookies at the end of each second. Also, to make the game more interesting to play, Kostya decided to add a limit: at each moment of time, he can use only one building. Of course, he can change the active building each second at his discretion.

It's important that Kostya is playing a version of the game where he can buy new buildings and change active building only at time moments that are multiples of one second. Kostya can buy new building and use it at the same time. If Kostya starts to use a building at the time moment t, he can get the first profit from it only at the time moment $t + 1$.

Kostya wants to earn at least s cookies as quickly as possible. Determine the number of seconds he needs to do that.

Input

The first line contains two integers n and s ($1 \le n \le 2 \cdot 10^5$, $1 \le s \le 10^{16}$) — the number of buildings in the game and the number of cookies Kostya wants to earn.

Each of the next n lines contains two integers v_i and c_i ($1 \le v_i \le 10^8$, $0 \le c_i \le 10^8$) — the number of cookies the i-th building brings per second and the building's price.

Output

Output the only integer — the minimum number of seconds Kostya needs to earn at least s cookies. It is guaranteed that he can do it.

Examples

cookie.in	cookie.out

3 9
1 0
2 3
5 4
	6

3 6
1 0
2 2
5 4
	5

3 13
1 0
2 2
6 5
	7

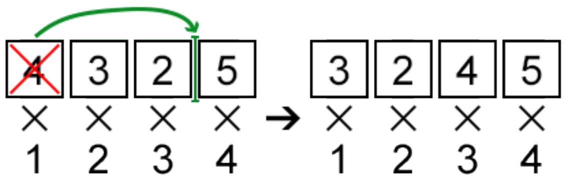
1 10000000000000000 1 0	10000000000000000

Problem C. Product Sum

Input file:	`sum.in`
Output file:	`sum.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Blake is the boss of Kris, however, this doesn't spoil their friendship. They often gather at the bar to talk about intriguing problems about maximising some values. This time the problem is really special.

You are given an array a of length n. The *characteristic* of this array is the value $c = \sum_{i=1}^{n} a_i \cdot i$ — the sum of the products of the values a_i by i. One may perform the following operation **exactly once**: pick some element of the array and move to any position. In particular, it's allowed to move the element to the beginning or to the end of the array. Also, it's allowed to put it back to the initial position. The goal is to get the array with the maximum possible value of characteristic.



Input

The first line of the input contains a single integer n ($2 \le n \le 200\,000$) — the size of the array a.

The second line contains n integers a_i ($1 \le i \le n$, $|a_i| \le 1\,000\,000$) — the elements of the array a.

Output

Print a single integer — the maximum possible value of characteristic of a that can be obtained by performing no more than one move.

Examples

sum.in	sum.out
4 4 3 2 5	39
5 1 1 2 7 1	49
3 1 1 2	9

Note

In the first sample, one may pick the first element and place it before the third (before 5). Thus, the answer will be $3 \cdot 1 + 2 \cdot 2 + 4 \cdot 3 + 5 \cdot 4 = 39$.

In the second sample, one may pick the fifth element of the array and place it before the third. The answer will be $1 \cdot 1 + 1 \cdot 2 + 1 \cdot 3 + 2 \cdot 4 + 7 \cdot 5 = 49$.

Problem D. Function

Input file:	`function.in`
Output file:	`function.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Serega and Fedor play with functions. One day they came across a very interesting function. It looks like that:

- $f(1, j) = a[j]$, $1 \le j \le n$.
- $f(i, j) = \min(f(i-1, j), f(i-1, j-1)) + a[j]$, $2 \le i \le n$, $i \le j \le n$.

Here a is an integer array of length n.

Serega and Fedya want to know what values this function takes at some points. But they don't want to calculate the values manually. So they ask you to help them.

Input

The first line contains integer n ($1 \le n \le 10^5$) — the length of array a. The next line contains n integers: $a[1], a[2], ..., a[n]$ ($0 \le a[i] \le 10^4$).

The next line contains integer m ($1 \le m \le 10^5$) — the number of queries. Each of the next m lines contains two integers: x_i, y_i ($1 \le x_i \le y_i \le n$). Each line means that Fedor and Serega want to know the value of $f(x_i, y_i)$.

Output

Print m lines — the answers to the guys' queries.

Examples

function.in	function.out

6
2 2 3 4 3 4
4
4 5
3 4
3 4
2 3
	12
	9
	9
	5

7
1 3 2 3 4 0 2
4
4 5
2 3
1 4
4 6
	11
	4
	3
	0

Problem E. Ciel and Gondolas

Input file:	`gondolas.in`
Output file:	`gondolas.out`
Time limit:	4 seconds
Memory limit:	512 megabytes

Fox Ciel is in the Amusement Park. And now she is in a queue in front of the Ferris wheel. There are n people (or foxes more precisely) in the queue: we use first people to refer one at the head of the queue, and n-th people to refer the last one in the queue.

There will be k gondolas, and the way we allocate gondolas looks like this:

- When the first gondolas come, the q_1 people in head of the queue go into the gondolas.
- Then when the second gondolas come, the q_2 people in head of the remain queue go into the gondolas.

  ...
- The remain q_k people go into the last (k-th) gondolas.

Note that q_1, q_2, ..., q_k must be positive. You can get from the statement that $\sum_{i=1}^{k} q_i = n$ and $q_i > 0$.

You know, people don't want to stay with strangers in the gondolas, so your task is to find an optimal allocation way (that is find an optimal sequence q) to make people happy. For every pair of people i and j, there exists a value u_{ij} denotes a level of unfamiliar. You can assume $u_{ij} = u_{ji}$ for all i, j ($1 \le i, j \le n$) and $u_{ii} = 0$ for all i ($1 \le i \le n$). Then an unfamiliar value of a gondolas is the sum of the levels of unfamiliar between any pair of people that is into the gondolas.

A total unfamiliar value is the sum of unfamiliar values for all gondolas. Help Fox Ciel to find the minimal possible total unfamiliar value for some optimal allocation.

Input

The first line contains two integers n and k ($1 \le n \le 4000$ and $1 \le k \le min(n, 800)$) — the number of people in the queue and the number of gondolas. Each of the following n lines contains n integers — matrix u, ($0 \le u_{ij} \le 9$, $u_{ij} = u_{ji}$ and $u_{ii} = 0$).

Please, use fast input methods (for example, please use BufferedReader instead of Scanner for Java).

Output

Print an integer — the minimal possible total unfamiliar value.

Examples

gondolas.in	gondolas.out

5 2
0 0 1 1 1
0 0 1 1 1
1 1 0 0 0
1 1 0 0 0
1 1 0 0 0
	0

8 3
0 1 1 1 1 1 1 1
1 0 1 1 1 1 1 1
1 1 0 1 1 1 1 1
1 1 1 0 1 1 1 1
1 1 1 1 0 1 1 1
1 1 1 1 1 0 1 1
1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 0
	7

3 2
0 2 0
2 0 3
0 3 0
	2

Note

In the first example, we can allocate people like this: 1, 2 goes into a gondolas, 3, 4, 5 goes into another gondolas.

In the second example, an optimal solution is : 1, 2, 3 | 4, 5, 6 | 7, 8.

Problem F. Hard drive disks

Input file:	`hard.in`
Output file:	`hard.out`
Time limit:	2 seconds
Memory limit:	512 megabytes

There are n pairs of hard drive disks(HDDs) in a cluster. Each HDD can be represented as a point on the infinite straight line with integer coordinate. In each pair one HDD is main and one is backup.

You want to set up k computers(which also points on the same line with integer coordinates) and then connect some HDDs with computers via wires. After that main and backup HDDs in each pair must be connected with the same computer. Each HDD must be connected with exactly one computer and each computer can be connected with any number of HDDs(possibly zero). Each wire must connect one HDD with one computer and its length is the distance between corresponding points on the line. What is the minimum possible total length of wires you can achieve?

Input

The first line contains two integers n and k - the number of pairs of HDDs and the number of computers. Each of the next n lines contain two integers a_i, b_i - coordinates of main and backup HDDs.

Constraints

$2 \leq k \leq n \leq 100000$

$4 \leq k \times n \leq 100000$

$-10^9 \leq a_i, b_i \leq 10^9$

Output

Output must contain single number - the answer to the question.

Example

hard.in	hard.out

5 2
6 7
-1 1
0 1
5 2
7 3
	13

Note

In the sample, it's optimal to place computers at the positions 0 and 6. Then connect the second and the third pairs of HDDs with the first computer and the others with the second. The total length of wires connected with the first computer will be 3 and 10 for the second one giving a total length of 13.

Problem G. Levels and Regions

Input file:	`levels.in`
Output file:	`levels.out`
Time limit:	3 seconds
Memory limit:	256 megabytes

Radewoosh is playing a computer game. There are n levels, numbered 1 through n. Levels are divided into k regions (groups). Each region contains some positive number of consecutive levels.

The game repeats the the following process:

1. If all regions are beaten then the game ends immediately. Otherwise, the system finds the first region with at least one non-beaten level. Let X denote this region.
2. The system creates an empty bag for tokens. Each token will represent one level and there may be many tokens representing the same level.
   - For each already beaten level i in the region X, the system adds t_i tokens to the bag (tokens representing the i-th level).
   - Let j denote the first non-beaten level in the region X. The system adds t_j tokens to the bag.
3. Finally, the system takes a uniformly random token from the bag and a player starts the level represented by the token. A player spends one hour and beats the level, even if he has already beaten it in the past.

Given n, k and values $t_1, t_2, \ldots, t_n$, your task is to split levels into regions. Each level must belong to exactly one region, and each region must contain non-empty consecutive set of levels. What is the minimum possible expected number of hours required to finish the game?

Input

The first line of the input contains two integers n and k ($1 \leq n \leq 200\,000$, $1 \leq k \leq min(50, n)$) — the number of levels and the number of regions, respectively.

The second line contains n integers $t_1, t_2, \ldots, t_n$ ($1 \leq t_i \leq 100\,000$).

Output

Print one real number — the minimum possible expected value of the number of hours spent to finish the game if levels are distributed between regions in the optimal way. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct if $\frac{|a-b|}{max(1,b)} \le 10^{-4}$.

Examples

levels.in	levels.out

`4 2`
`100 3 5 7`
	`5.7428571429`

`6 2`
`1 2 4 8 16 32`
	`8.5000000000`

Note

In the first sample, we are supposed to split 4 levels into 2 regions. It's optimal to create the first region with only one level (it must be the first level). Then, the second region must contain other three levels.

In the second sample, it's optimal to split levels into two regions with 3 levels each.

Problem H. Kalila and Dimna in the Logging Industry

Input file:	`logging.in`
Output file:	`logging.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Kalila and Dimna are two jackals living in a huge jungle. One day they decided to join a logging factory in order to make money.

The manager of logging factory wants them to go to the jungle and cut n trees with heights $a_1, a_2, \ldots, a_n$. They bought a chain saw from a shop. Each time they use the chain saw on the tree number i, they can decrease the height of this tree by one unit. Each time that Kalila and Dimna use the chain saw, they need to recharge it. Cost of charging depends on the id of the trees which have been cut completely (a tree is cut completely if its height equal to 0). If the maximum id of a tree which has been cut completely is i (the tree that have height a_i in the beginning), then the cost of charging the chain saw would be b_i. If no tree is cut completely, Kalila and Dimna cannot charge the chain saw. The chainsaw is charged in the beginning. We know that for each $i < j$, $a_i < a_j$ and $b_i > b_j$ and also $b_n = 0$ and $a_1 = 1$. Kalila and Dimna want to cut all the trees completely, with minimum cost.

They want you to help them! Will you?

Input

The first line of input contains an integer n ($1 \le n \le 10^5$). The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$). The third line of input contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 10^9$).

It's guaranteed that $a_1 = 1$, $b_n = 0$, $a_1 < a_2 < \cdots < a_n$ and $b_1 > b_2 > \cdots > b_n$.

Output

The only line of output must contain the minimum cost of cutting all the trees completely.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Examples

logging.in	logging.out

`5`
`1 2 3 4 5`
`5 4 3 2 0`
	`25`

`6`
`1 2 3 10 20 30`
`6 5 4 3 2 0`
	`138`

Problem I. Candies and Stones

Input file:	`candies.in`
Output file:	`candies.out`
Time limit:	20 seconds
Memory limit:	256 megabytes

Little Gerald and his coach Mike play an interesting game. At the beginning of the game there is a pile consisting of n candies and a pile consisting of m stones. Gerald and Mike move in turns, Mike goes first. During his move Mike checks how many candies and stones Gerald has eaten. Let Gerald eat a candies and b stones. Then Mike awards Gerald $f(a, b)$ prize points. Gerald during his move either eats a candy from the pile of candies or a stone from the pile of stones. As Mike sees that Gerald has eaten everything apart one candy and one stone, he awards points for the last time and the game ends. Gerald is not allowed to eat all the candies, and he is not allowed to eat all the stones too. Tell Gerald how to play to get the largest possible number of points: it is required to find one of the possible optimal playing strategies for Gerald.

Input

The first line contains three integers n, m, p ($1 \le n, m \le 20000$, $1 \le p \le 10^9$). The second line contains n integers $x_0, x_1, \ldots, x_{n-1}$ ($0 \le x_i \le 20000$).

The third line contains m integers y_0, y_1, …, y_{m-1} ($0 \le y_i \le 20000$). The value of $f(a,b)$ is calculated as a remainder of the division of the sum $x_a + y_b$ by number p.

Output

Print on the first line the only number: the maximal number of points Gerald can earn. Print on the second line a sting consisting of $n+m-2$ characters, each of which is either a "C" or "S", the i-th character should be "C" if Gerald's i-th move should be eating a candy and "S" if he should eat a stone.

Examples

candies.in	candies.out

2 2 10
0 0
0 1
	2
	SC

3 3 10
0 2 0
0 0 2
	10
	CSSC

3 3 2
0 1 1
1 1 0
	4
	SCSC

Note

In the first test if Gerald's first move is eating a stone, he will receive a point for it and if he eats a candy, he will get zero pints. In any way Gerald will get 0 points before his first move, and 1 after his second one. This, the maximum number of points Gerald can get equals to 2, and for that he should first eat a stone, then a candy.

Problem J. ACM and ICPC

Input file:	acm.in
Output file:	acm.out
Time limit:	2 seconds
Memory limit:	256 megabytes

Consider a ACM trees with next features:

- A non-negative integer value is assigned to each vertex;
- Each vertex has no more than two children.

Lets call the ACM tree ICPC, if it has the following features:

- Value of each vertex is the sum of values of its children;
- Value of each leaf is not greater than 1.

Given an ACM tree. We can do the following operations:

- add 1 to the value of some vertex;
- subtract 1 from the value of some vertex;

Find out the minimal number of operations needed to transform given ACM tree to the ICPC tree.

Input

The first line of the input contains one integer — number of vertices in the given ACM tree ($1 \le n \le 5000$). Second line contains two integers a_i ($0 \le a_i \le 5000$); i-th of these integers denotes the value of i-th vertex. i-th of the next $n-1$ lines contains two integers a and b ($1 \le a, b \le n$) denoting that vertices a and b are connected by an edge. Root of the ACM tree is placed at the vertex with number 1.

Output

Print minimum number of operations, needed to transform given ACM tree into ICPC tree.

Examples

acm.in	acm.out

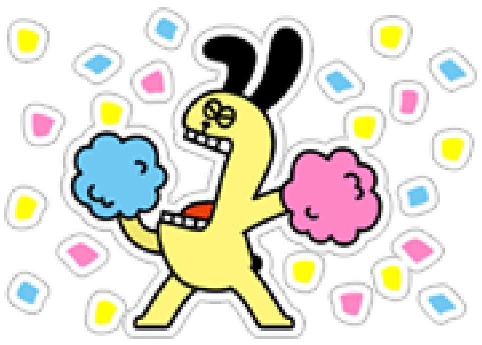
2
1 0
1 2
	1

5
5 1 3 0 1
1 2
1 3
3 4
3 5
	4

Problem K. Tavas and Pashmaks

Input file:	tavas.in
Output file:	tavas.out
Time limit:	1 second
Memory limit:	256 megabytes

Tavas is a cheerleader in the new sports competition named "Pashmaks".



This competition consists of two part: swimming and then running. People will immediately start running R meters after they finished swimming exactly S meters. A winner is a such person that nobody else finishes running before him/her (there may be more than one winner).

Before the match starts, Tavas knows that there are n competitors registered for the match. Also, he knows that i-th person's swimming speed is s_i meters per second and his/her running speed is r_i meters per second. Unfortunately, he doesn't know the values of R and S, but he knows that they are real numbers greater than 0.

As a cheerleader, Tavas wants to know who to cheer up. So, he wants to know all people that might win. We consider a competitor might win if and only if there are some values of R and S such that with these values, (s)he will be a winner.

Tavas isn't really familiar with programming, so he asked you to help him.

Input

The first line of input contains a single integer n ($1 \le n \le 2 \times 10^5$).

The next n lines contain the details of competitors. i-th line contains two integers s_i and r_i ($1 \le s_i, r_i \le 10^4$).

Output

In the first and the only line of output, print a sequence of numbers of possible winners in increasing order.

Examples

tavas.in	tavas.out

3
1 3
2 2
3 1
	1 2 3

3
1 2
1 1
2 1
	1 3

Problem L. Igloo Skyscraper

Input file:	`igloo.in`
Output file:	`igloo.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Today the North Pole hosts an Olympiad in a sport called. . . toy igloo skyscrapers' building!

There are n walruses taking part in the contest. Each walrus is given a unique number from 1 to n. After start each walrus begins to build his own igloo skyscraper. Initially, at the moment of time equal to 0, the height of the skyscraper i-th walrus is equal to a_i. Each minute the i-th walrus finishes building b_i floors.

The journalists that are reporting from the spot where the Olympiad is taking place, make q queries to the organizers. Each query is characterized by a group of three numbers l_i, r_i, t_i. The organizers respond to each query with a number x, such that:

1. Number x lies on the interval from l_i to r_i inclusive ($l_i \le x \le r_i$).

2. The skyscraper of the walrus number x possesses the maximum height among the skyscrapers of all walruses from the interval $[l_i, r_i]$ at the moment of time t_i.

For each journalists' query print the number of the walrus x that meets the above-given criteria. If there are several possible answers, print any of them.

Input

The first line contains numbers n and q ($1 \le n, q \le 10^5$). Next n lines contain pairs of numbers a_i, b_i ($1 \le a_i, b_i \le 10^9$). Then follow q queries i the following format l_i, r_i, t_i, one per each line ($1 \le l_i \le r_i \le n$, $0 \le t_i \le 10^6$). All input numbers are integers.

Output

For each journalists' query print the number of the walrus x that meets the criteria, given in the statement. Print one number per line.

Examples

igloo.in	igloo.out

5 4
4 1
3 5
6 2
3 5
6 5
1 5 2
1 3 5
1 1 0
1 5 0
	5
	2
	1
	5

5 4
6 1
5 1
2 5
4 3
6 1
2 4 1
3 4 5
1 4 5
1 2 0
	3
	3
	3
	1