Codeforces Round #352 (Div. 1)

A. Recycling Bottles

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It was recycling day in Kekoland. To celebrate it Adil and Bera went to Central Perk where they can take bottles from the ground and put them into a recycling bin.

We can think Central Perk as coordinate plane. There are n bottles on the ground, the i-th bottle is located at position (x_i, y_i). Both Adil and Bera can carry only one bottle at once each.

For both Adil and Bera the process looks as follows:

1. Choose to stop or to continue to collect bottles.
2. If the choice was to continue then choose some bottle and walk towards it.
3. Pick this bottle and walk to the recycling bin.
4. Go to step 1.

Adil and Bera may move independently. They are allowed to pick bottles simultaneously, all bottles may be picked by any of the two, it's allowed that one of them stays still while the other one continues to pick bottles.

They want to organize the process such that the total distance they walk (the sum of distance walked by Adil and distance walked by Bera) is minimum possible. Of course, at the end all bottles should lie in the recycling bin.

Input

First line of the input contains six integers a_x, a_y, b_x, b_y, t_x and t_y $(0 \le a_x, a_y, b_x, b_y, t_x, t_y \le 10^9)$ — initial positions of Adil, Bera and recycling bin respectively.

The second line contains a single integer n $(1 \le n \le 100\,000)$ — the number of bottles on the ground.

Then follow n lines, each of them contains two integers x_i and y_i $(0 \le x_i, y_i \le 10^9)$ — position of the i-th bottle.

It's guaranteed that positions of Adil, Bera, recycling bin and all bottles are distinct.

Output

Print one real number — the minimum possible total distance Adil and Bera need to walk in order to put all bottles into recycling bin. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

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 .

Examples

input
3 1 1 2 0 0 3 1 1 2 1 2 3
output
11.084259940083

input
5 0 4 2 2 0 5 5 2 3 0 5 5 3 5 3 3
output
33.121375178000

Note

Consider the first sample.

Adil will use the following path: .

Bera will use the following path: .

Adil's path will be units long, while Bera's path will be units long.

B. Robin Hood

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

We all know the impressive story of Robin Hood. Robin Hood uses his archery skills and his wits to steal the money from rich, and return it to the poor.

There are n citizens in Kekoland, each person has c_i coins. Each day, Robin Hood will take exactly 1 coin from the richest person in the city and he will give it to the poorest person (poorest person right after taking richest's 1 coin). In case the choice is not unique, he will select one among them at random. Sadly, Robin Hood is old and want to retire in k days. He decided to spend these last days with helping poor people.

After taking his money are taken by Robin Hood richest person may become poorest person as well, and it might even happen that Robin Hood will give his money back. For example if all people have same number of coins, then next day they will have same number of coins too.

Your task is to find the difference between richest and poorest persons wealth after k days. Note that the choosing at random among richest and poorest doesn't affect the answer.

Input

The first line of the input contains two integers n and k ($1 \le n \le 500\,000$, $0 \le k \le 10^9$) — the number of citizens in Kekoland and the number of days left till Robin Hood's retirement.

The second line contains n integers, the i-th of them is c_i ($1 \le c_i \le 10^9$) — initial wealth of the i-th person.

Output

Print a single line containing the difference between richest and poorest peoples wealth.

Examples

input
4 1 1 1 4 2
output
2

input
3 1 2 2 2
output
0

Note

Lets look at how wealth changes through day in the first sample.

1. $[1, 1, 4, 2]$
2. $[2, 1, 3, 2]$ or $[1, 2, 3, 2]$

So the answer is $3 - 1 = 2$

In second sample wealth will remain the same for each person.

C. Ultimate Weirdness of an Array

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Yasin has an array a containing n integers. Yasin is a 5 year old, so he loves ultimate weird things.

Yasin denotes *weirdness* of an array as maximum $gcd(a_i, a_j)$ value among all $1 \le i < j \le n$. For $n \le 1$ weirdness is equal to 0, $gcd(x, y)$ is the greatest common divisor of integers x and y.

He also defines the *ultimate weirdness* of an array. Ultimate weirdness is where $f(i, j)$ is weirdness of the new array a obtained by removing all elements between i and j inclusive, so new array is $[a_1 \ldots a_{i-1}, a_{j+1} \ldots a_n]$.

Since 5 year old boys can't code, Yasin asks for your help to find the value of ultimate weirdness of the given array a!

Input

The first line of the input contains a single integer n $(1 \le n \le 200\,000)$ — the number of elements in a.

The next line contains n integers a_i $(1 \le a_i \le 200\,000)$, where the i-th number is equal to the i-th element of the array a. It is guaranteed that all a_i are distinct.

Output

Print a single line containing the value of ultimate weirdness of the array a.

Example

input
3 2 6 3
output
6

Note

Consider the first sample.

- $f(1, 1)$ is equal to 3.
- $f(2, 2)$ is equal to 1.
- $f(3, 3)$ is equal to 2.
- $f(1, 2), f(1, 3)$ and $f(2, 3)$ are equal to 0.

Thus the answer is $3 + 0 + 0 + 1 + 0 + 2 = 6$.

D. Roads in Yusland

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Mayor of Yusland just won the lottery and decided to spent money on something good for town. For example, repair all the roads in the town.

Yusland consists of n intersections connected by $n - 1$ bidirectional roads. One can travel from any intersection to any other intersection using only these roads.

There is only one road repairing company in town, named "RC company". Company's center is located at the intersection 1. RC company doesn't repair roads you tell them. Instead, they have workers at some intersections, who can repair only some specific paths. The i-th worker can be paid c_i coins and then he repairs **all roads** on a path from u_i to some v_i that **lies on the path** from u_i to intersection 1.

Mayor asks you to choose the cheapest way to hire some subset of workers in order to repair all the roads in Yusland. It's allowed that some roads will be repaired more than once.

If it's impossible to repair all roads print -1.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 300\,000$) — the number of cities in Yusland and the number of workers respectively.

Then follow $n-1$ line, each of them contains two integers x_i and y_i ($1 \le x_i, y_i \le n$) — indices of intersections connected by the i-th road.

Last m lines provide the description of workers, each line containing three integers u_i, v_i and c_i ($1 \le u_i, v_i \le n$, $1 \le c_i \le 10^9$). This means that the i-th worker can repair all roads on the path from v_i to u_i for c_i coins. It's guaranteed that v_i lies on the path from u_i to 1. Note that v_i and u_i may coincide.

Output

If it's impossible to repair all roads then print -1. Otherwise print a single integer — minimum cost required to repair all roads using "RC company" workers.

Example

input
6 5 1 2 1 3 3 4 4 5 4 6 2 1 2 3 1 4 4 1 3 5 3 1 6 3 2
output
8

Note

In the first sample, we should choose workers with indices 1, 3, 4 and 5, some roads will be repaired more than once but it is OK. The cost will be equal to $2 + 3 + 1 + 2 = 8$ coins.

E. Organizing a Race

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Kekoland is a country with n beautiful cities numbered from left to right and connected by n - 1 roads. The i-th road connects cities i and $i + 1$ and length of this road is w_i kilometers.

When you drive in Kekoland, each time you arrive in city i by car you immediately receive g_i liters of gas. There is no other way to get gas in Kekoland.

You were hired by the Kekoland president Keko to organize the most beautiful race Kekoland has ever seen. Let race be between cities l and r ($l \le r$). Race will consist of two stages. On the first stage cars will go from city l to city r. After completing first stage, next day second stage will be held, now racers will go from r to l with their cars. Of course, as it is a race, racers drive directly from start city to finish city. It means that at the first stage they will go only right, and at the second stage they go only left. Beauty of the race between l and r is equal to r - $l + 1$ since racers will see r - $l + 1$ beautiful cities of Kekoland. Cars have infinite tank so racers will take all the gas given to them.

At the beginning of **each stage** racers start the race with empty tank (0 liters of gasoline). They will immediately take their gasoline in start cities (l for the first stage and r for the second stage) right after the race starts.

It may not be possible to organize a race between l and r if cars will run out of gas before they reach finish.

You have k presents. Each time you give a present to city i its value g_i increases by 1. You may distribute presents among cities in any way (also give many presents to one city, each time increasing g_i by 1). What is the most beautiful race you can organize?

Each car consumes 1 liter of gas per one kilometer.

Input

The first line of the input contains two integers n and k ($2 \le n \le 100\,000$, $0 \le k \le 10^9$) — the number of cities in Kekoland and the number of presents you have, respectively.

Next line contains n - 1 integers. The i-th of them is w_i ($1 \le w_i \le 10^9$) — the length of the road between city i and $i + 1$.

Next line contains n integers. The i-th of them is g_i ($0 \le g_i \le 10^9$) — the amount of gas you receive every time you enter city i.

Output

Print a single line — the beauty of the most beautiful race you can organize.

Examples

input
4 4 2 2 2 1 1 1 1
output
4

input
8 5 2 2 2 3 7 3 1 1 3 1 5 4 0 2 5
output
7

Note

In first sample if you give one present to each city then it will be possible to make a race between city 1 and city 4.

In second sample you should add 1 to g_5 and 4 to g_6, then it will be possible to make a race between cities 2 and 8.