Talent Show

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Faced with the harsh reality of the software engineering job market, you've taken a new path as a primary school teacher. As your first year comes to a close, you decide to host an End-of-Year Talent Show to exhibit the various talents of your students.

There are n students willing to perform, as individual acts only. Each student i has:

- a *skill level* s_i which represents the entertainment value they contribute to the show, and
- a *departure time* d_i which represents that they will have to leave after the first d_i acts.

You will need to schedule some or all of the students to perform.

Your task is to determine the *maximum total skill level* contributed by students who participate in performances that form a valid sequence.

Input

The first line contains a single integer n $(1 \leq n \leq 200,000)$, representing the number of students.

Each of the next n lines contains two space-separated integers s_i and d_i $(1 \leq d_i \leq n)$, representing the skill level and departure time of the ith student respectively.

Output

Output a single integer, the maximum total skill that can be achieved from a valid sequence of performances.

Scoring

For Subtask 1 (50 points):

- $s_i = 1$ for all i.

For Subtask 2 (50 points):

- $1 \leq s_i \leq 10^9$ for all i.

Examples

standard input	standard output

4
1 2
1 2
1 3
1 3
	3

3
4 1
5 2
6 3
	15

5
3 4
4 1
2 2
2 4
5 1
	12

Note

In the first sample case, we can schedule the first two students to complete their acts before leaving, and then either of the remaining students as the third act. Note that this case complies with the constraints of Subtask 1.

In the second sample case, all three students can perform in the order they appear in the input.

In the third sample case, the second and fifth students will both leave after the first act, so at least one of them will not get to perform. One optimal sequence is as follows.

Act	1	2	3	4
Student	5	3	4	1

Enclosure

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

In feudal England, much of the agricultural land in a village would be held in common for peasants to graze livestock and grow crops. However, over several hundred years this system was replaced by the *enclosure movement*, in which the common land was partitioned into fields for the exclusive use of private owners. Backed by Parliament and enforced by erecting fences and walls (and perhaps more importantly armed guards), enclosure was totally cool with all the peasants who no longer had access to land needed for their livelihood, and definitely didn't lead to any riots or anything like that.$^{[citation\ needed]}$

The common land in the village of Townsham forms a grid of dimensions n by m, where each cell of the grid represents one acre of land. Each acre has been assessed for its agricultural yield a_{ij} and **cannot be divided** up.

Townsham starts with a wall around the perimeter of the land. The lord wants to partition the land into fields by erecting new walls through it. The walls must be built **sequentially**, each one **splitting** a rectangular field into two smaller rectangular fields by a straight vertical or horizontal line.

The lord does not care how many fields are made, but fears that if any one field has too much agricultural yield (found by accumulating the yield of each acre within it), then the owner of this field might accumulate enough wealth and power to challenge him. Therefore, he has decreed that no field may have yield greater than a given total T.

Determine the minimum **length** of walls required to divide the land.

Input

The first line of input consists of three space-separated integers n, m ($1 \leq n, m \leq 60$) and T ($1 \leq T \leq 360,000$), representing the number of rows and columns of the grid and the maximum allowed yield in any one field.

n lines follow, the ith of which consists of m space-separated integers $a_{i,1}, a_{i,2}, \ldots, a_{i,m}$, the jth of which represents the agricultural yield of the jth acre in the ith row.

Output

Output a single integer, the minimum length of walls required to complete a valid enclosure.

If a valid enclosure is not possible, output -1 instead.

Scoring

For Subtask 1 (50 points):

- $a_{i,j} = 1$ for all $1 \leq i \leq n$, $1 \leq j \leq m$.

For Subtask 2 (50 points):

- $1 \leq a_{i,j} \leq 100$ for all $1 \leq i \leq n$, $1 \leq j \leq m$.

Examples

standard input	standard output

3 3 2
1 1 1
1 1 1
1 1 1
	8

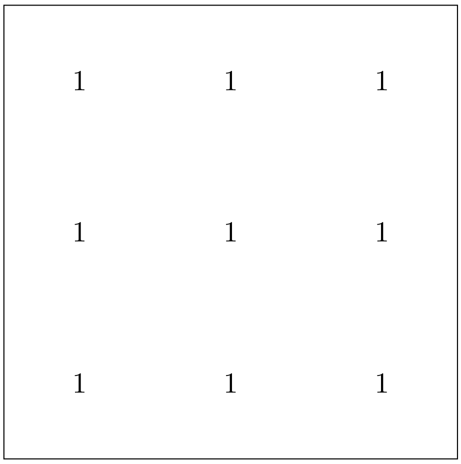
2 3 5
1 1 1
5 1 1
	3

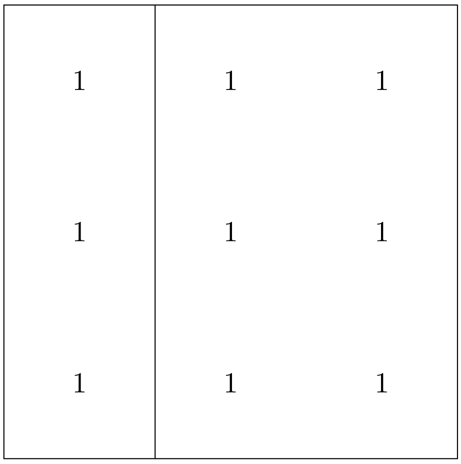
Note

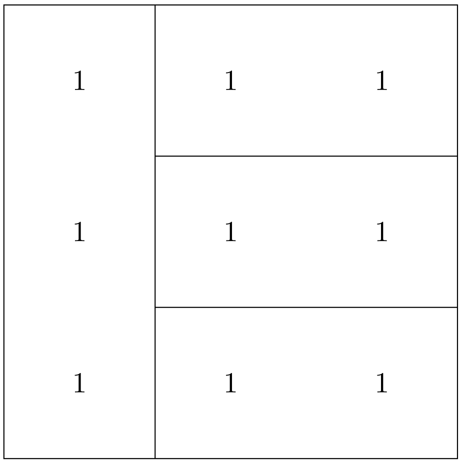
In the first sample test case, the fields will be split by 4 walls:

- a 3-length vertical wall splitting off the first column,
- two 2-length horizontal walls, splitting the right half into three pieces, and
- a 1-length horizontal wall splitting the bottom left acre,

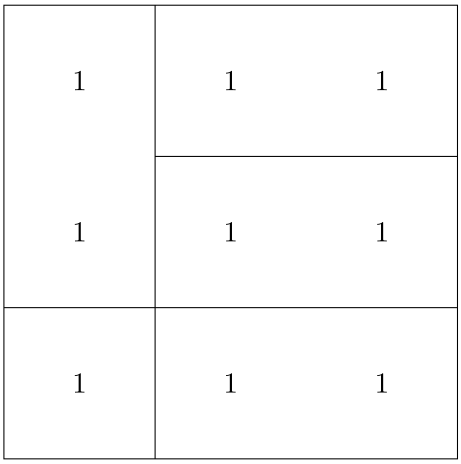
for a total wall length of 8.







1	1	1
1	1	1
1	1	1

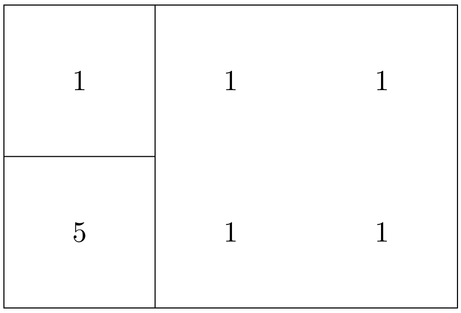


1	1	1
1	1	1
1	1	1

In the second sample test case, it is necessary to isolate the bottom left corner as this acre alone reaches the maximum yield. Since our cuts must cross the entire field, the cheapest approach is:

- a 2-length vertical wall splitting off the first column and
- a 1-length horizontal wall splitting the bottom left acre,

for a total wall length of 3.



1	1	1
5	1	1

LinkedList

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 128 megabytes

Note that the memory limit is 128MB rather than the usual 256MB.

Anubhav Ammangi · 1 connection
4d · Edited · 🌐

Feeling truly inspired to share this as LinkedList's FIRST post.

The state of online professional networking has prompted the development of LinkedList, a platform where not every minor inconvenience/success is a 'life-changing leadership lesson' or a humblebrag about 'resilience, grit, and unicorn-level growth mindset.'

We could not be prouder of what we've achieved!

LinkedList is trialed by N users, numbered from 1 to N. The users have established two-way 'connections'; however, to maintain professionalism, all of these connections are between a user and their boss. Each user i has exactly *one* boss, user b_i, except for the Bossboss (user 1), who answers to no one but can still have connections with direct subordinates.

We're honoured to share that our team has made a feature that enables any user to find the 'connection distance' between two users. The connection distance between two users is defined as the fewest number of direct LinkedList connections required to get from one user to the other.

Unfortunately, we are not yet able to analyse another form of user interactions — group chats. On LinkedList, each user begins in their own group chat, and at any time can only belong to one group chat. Each user has access to the 'invite link' for the group they are in.

To help us innovate further with the amazing platform we've developed, we'd like you to process Q operations of the following three types:

- **1** x_i y_i: User x_i sends an invite directly to user y_i. User y_i sends this invite in their own group chat, and everyone in user y_i's group chat hops into user x_i's, leaving their own.
- **2** x_i y_i: User x_i shares the invite with a direct connection who has the lowest 'connection distance' to user y_i. This user then repeats the process, sharing the invite with a direct connection of theirs with the lowest 'connection distance' to user y_i, until user y_i finally receives the invite. Note that all users who receive the invite (including user y_i) will send it in their own group chat, prompting all users there to leave and join x_i's.
- **3** x_i y_i: Determine whether user x_i and user y_i are part of the same group chat.

Feel free to connect if you're able to achieve this for us. Good luck!

To assist you, we've provided the problem specifications and some samples below — reply below if you're able to explain them!

#AlgoDesign #LeadershipThroughCode #TechPhilosophy #ComplexityWithPurpose #GrowthMindset #BelongingInTech #ConnectednessMatters #ComputerScienceWisdom

Input

The first line contains two space-separated integers N and Q ($1 \le N, Q \le 500{,}000$) — the number of LinkedList users and the operations respectively.

The second line contains $n - 1$ space-separated integers $b_2, \ldots, b_N$, where b_i ($1 \le b_i \le N$) represents the boss of user i. Recall that user 1, being the Bossboss, has no boss.

The next Q lines describe the operations. Each line begins with either 1, 2, or 3, followed by x_i and y_i ($1 \leq x_i, y_i \leq N$).

IMPORTANT: This problem has a lot of input data. It is possible to receive a 'Time Limit Exceeded' verdict by using `cin/cout` incorrectly. If you are using `cin/cout` you need to ensure that:

- You include the line `std::ios_base::sync_with_stdio(0); std::cin.tie(0);` at the beginning of your `main` function.
- You *do not* use `endl`. Use `'\n'` instead.
- You *do not* use `scanf/printf` simultaneously with `cin/cout`.

If you are exclusively using `scanf/printf` rather than `cin/cout`, then you do not need to worry about any of this.

Output

Output one line for each operation of type 3: `YES` if user x_i and user y_i are in the same group chat and `NO` if otherwise.

Scoring

For Subtask 1 (50 points):

- For operations of type 2, all users only wish to invite the Bossboss (user 1) to their group chat. The Bossboss (user 1) remains free to invite any user to their group chat for operations of type 2.

  That is, for operations of type 2, either $x_i = 1$ or $y_i = 1$.
- No additional constraints for operations of type 1 or type 3.

For Subtask 2 (50 points):

- No additional constraints.

Examples

standard input	standard output

4 3
1 2 2
2 3 1
3 1 2
3 3 4
	YES
	NO

4 4
1 2 2
1 1 2
3 1 3
2 3 4
3 1 3
	NO
	YES

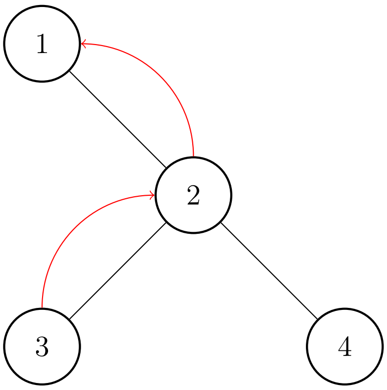
Note

Yiheng You · 2 connections
3d · Edited ·

Congratulations on the wonderful achievement LinkedList team! Thanks for sharing your insights.

I am delighted to announce that I have SOLVED this problem, and am a HUGE fan of LinkedList as well as this problem! For those of you still attempting this problem, here's the explanation for sample 1.

Initially, every user is in their own group chat. The first operation is of type 2 from user 3 to user 1. The invite link first reaches user 2, before being passed to user 1.



This leaves group chat compositions as $\{1,2,3\}$ and $\{4\}$.

Obviously, user 1 and user 2 are part of the same group chat so `YES` is outputted. Obviously, user 3 and user 4 are not part of the same group chat so `NO` is outputted.

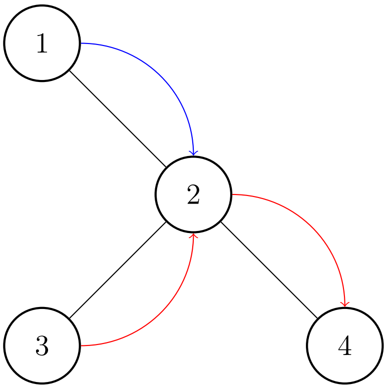
As a bonus, I'm happy to share that I noticed this case conforms with the restrictions of Subtask 1.

Yiheng You · 2 connections
2d · Edited · 🌍

I am beyond thrilled to announce that I have started a new position as Algorithm Designer and Developer at LinkedList! It truly is an INCREDIBLE honour; thanks to everyone who supported me on this journey.

My first task is to explain the remaining sample. Here's how it works.

As with the first sample, initially, every user is in their own group chat. The first operation is of type 1 from user 1 to user 2. The invite link is sent directly to user 2 from user 1.



This leaves group chat compositions as $\{1,2\}$, $\{3\}$ and $\{4\}$.

Obviously, user 1 and user 3 are not part of the same group chat so `NO` is outputted.

Then, the next operation is of type 2 from user 3 to user 4. The invite link first reaches user 2, who is already in a group chat with user 1, so user 2 posts the link in their group chat, causing both user 1 and user 2 to join user 3's group chat. User 2 then passes on the link to user 4, who also joins user 3's group chat.

This leaves the group chat composition as $\{1,2,3,4\}$.

Obviously, user 1 and user 3 are part of the same group chat so `YES` is outputted.

As a final note, if this problem resonated with you, like it did with me, do consider a career at LinkedList!