

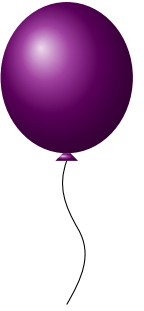
XCPC Training 2026 Lesson 6

10 problems

A	Subsegment sum	2
B	Stogovi	3
C	Windows	5
D	Distinct numbers	7
E	Wall	8
F	Performance Review	11
G	Divine Gifting	13
H	Yaxchilán Maze	15
I	Toll (hard version)	19
J	Crossing the Railways	21

Subsegment sum

TIME LIMIT: 2.0s
MEMORY LIMIT: 256MB



You are given an array containing N elements. Process a series of queries of element modifications and subsegment sum in this array.

INPUT

First line of the input contains two integers N and K — the size of the array and the number of queries ($1 \leq N \leq 100\,000$; $0 \leq K \leq 100\,000$).

The following K lines contain descriptions of queries:

1. **A i x** — assign the value x to the element with index i ($1 \leq i \leq n$; $0 \leq x \leq 10^9$);
2. **Q l r** — find the sum of array elements on positions from l to r ($1 \leq l \leq r \leq n$).

Initially the array consists of zeroes.

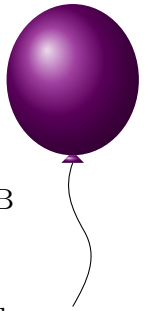
OUTPUT

For every query of the type “Q” print the sum on the given subsegment.

SAMPLES

Sample input 1	Sample output 1
5 9 A 2 2 A 3 1 A 4 2 Q 1 1 Q 2 2 Q 3 3 Q 4 4 Q 5 5 Q 1 5	0 2 1 2 0 5

Stogovi



TIME LIMIT: 2s
MEMORY LIMIT: 1024MB

Mirko is playing with stacks. In the beginning of the game, he has an empty stack denoted with number 0. In the i^{th} step of the game he will choose an existing stack denoted with v , copy it and do one of the following actions:

- place number i on top of the new stack
- remove the number from the top of the new stack
- choose another stack denoted with w and count how many different numbers exist that are in the new stack and in the stack denoted with w

The newly created stack is denoted with i .

Mirko doesn't like to work with stacks so he wants you to write a programme that will do it for him. For each operation of type b output the number removed from stack and for each operation of type c count the required numbers and output how many of them there are.

INPUT

The first line of input contains the integer N ($1 \leq N \leq 300\,000$), the number of steps in Mirko's game.

The steps of the game are chronologically denoted with the first N integers.

The i^{th} of the following N lines contains the description of the i^{th} step of the game in one of the following three forms:

- "a v" for operation of type a .
- "b v" for operation of type b .
- "c v w" for operation of type c .

The first character in the line denotes the type of operation and the following one or two denote the accompanying stack labels that will always be integers from the interval $[0, i - 1]$.

For each operation of type b , the stack we're removing the element from will not be empty.

OUTPUT

For each operation type b or c output the required number, each in their own line, in the order the operations were given in the input.

SAMPLES

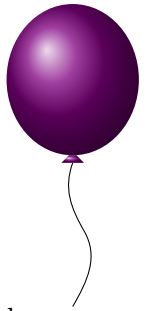
Sample input 1	Sample output 1
5 a 0 a 1 b 2 c 2 3 b 4	2 1 2

Sample input 2	Sample output 2
11 a 0 a 1 a 2 a 3 a 2 c 4 5 a 5 a 6 c 8 7 b 8 b 8	2 2 8 8

Clarification of the first example: In the beginning, we have the stack $S_0 = \{\}$. In the first step, we copy S_0 and place number 1 on top, so $S_1 = \{1\}$. In the second step, we copy S_1 and place 2 on top of it, $S_2 = \{1, 2\}$. In the third step we copy S_2 and remove number 2 from it, $S_3 = \{1\}$. In the fourth step we copy S_2 and denote the copy with S_4 , then count the numbers appearing in the newly created stack S_4 and stack S_3 , the only such number is number 1 so the solution is 1. In the fifth step we copy S_4 and remove number 2 from it, $S_5 = \{1\}$.

Windows

TIME LIMIT: 2.0s
MEMORY LIMIT: 256MB



The screen of your computer is displaying a set of rectangular windows which may intersect and overlap.

Find a point that is covered by the largest number of them.

INPUT

First line of the input contains one integer n — the number of windows ($1 \leq n \leq 50\,000$).

Next n lines contain coordinates of windows $x_{(1,i)} \ y_{(1,i)} \ x_{(2,i)} \ y_{(2,i)}$, where $(x_{(1,i)}, y_{(1,i)})$ are coordinates of the top left corner of the window number i and $(x_{(2,i)}, y_{(2,i)})$ are coordinates of its bottom right corner.

On the computer screen y coordinates increase from top to bottom and x coordinates increase from left to right.

All the coordinates are integers and their absolute value does not increase 10^6 .

OUTPUT

In the first line output the maximum number of windows that covers some point of the screen in the given configuration. In the second line output two integers separated by a space — coordinates of a point covered by the maximum number of windows. Windows cover their boundaries as well.

SAMPLES

Sample input 1	Sample output 1
2 0 0 3 3 1 1 4 4	2 1 3

Sample input 2	Sample output 2
1 0 0 1 1	1 0 1

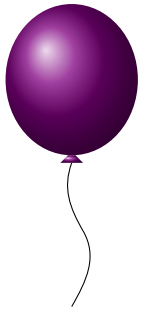
WINDOWS

Sample input 3	Sample output 3
4 0 0 1 1 0 1 1 2 1 0 2 1 1 1 2 2	4 1 1

Sample input 4	Sample output 4
5 0 0 1 1 0 1 1 2 0 0 2 2 1 0 2 1 1 1 2 2	5 1 1

Distinct numbers

TIME LIMIT: 3.0s
MEMORY LIMIT: 256MB



How many distinct numbers does a subsegment of the array contain?

INPUT

First line of input contains the length of the array n ($1 \leq n \leq 300\,000$). Second line of input contains n integers from 0 to 10^9-1 — array elements.

Third line contains the number of queries q ($1 \leq q \leq 300\,000$). Next q lines contain descriptions of queries, one per line. Each query is described by two integers l and r ($1 \leq l \leq r \leq n$).

OUTPUT

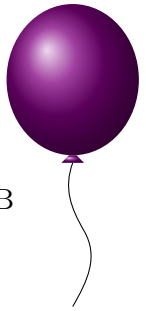
For each query output the number of distinct numbers on the corresponding subsegment.

SAMPLES

Sample input 1	Sample output 1
5	3
1 1 2 1 3	2
3	3
1 5	
2 4	
3 5	

Wall

TIME LIMIT: 3s
MEMORY LIMIT: 2048MB



Jian-Jia is building a wall by stacking bricks of the same size together. This wall consists of n columns of bricks, which are numbered 0 to $n - 1$ from left to right. The columns may have different heights. The height of a column is the number of bricks in it.

Jian-Jia builds the wall as follows. Initially there are no bricks in any column. Then, Jian-Jia goes through k phases of *adding* or *removing* bricks. The building process completes when all k phases are finished. In each phase Jian-Jia is given a range of consecutive brick columns and a height h , and he does the following procedure:

- In an *adding* phase, Jian-Jia adds bricks to those columns in the given range that have less than h bricks, so that they have exactly h bricks. He does nothing on the columns having h or more bricks.
- In a *removing* phase, Jian-Jia removes bricks from those columns in the given range that have more than h bricks, so that they have exactly h bricks. He does nothing on the columns having h bricks or less.

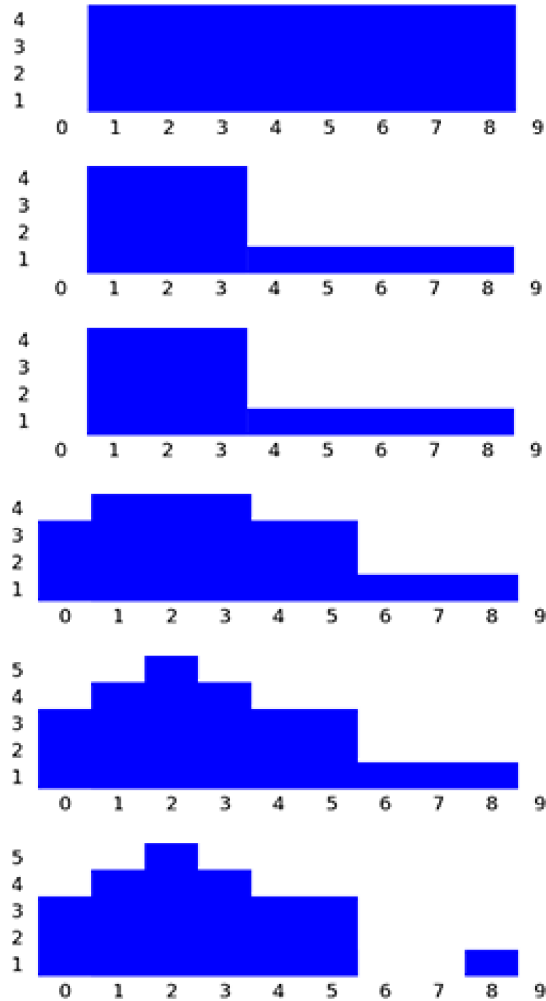
Your task is to determine the final shape of the wall.

EXAMPLE

We assume that there are 10 brick columns and 6 wall building phases. All ranges in the following table are inclusive. Diagrams of the wall after each phase are shown below.

phase	type	range	height
0	add	columns 1 to 8	4
1	remove	columns 4 to 9	1
2	remove	columns 3 to 6	5
3	add	columns 0 to 5	3
4	add	column 2	5
5	remove	columns 6 to 7	0

Since all columns are initially empty, after phase 0 each of the columns 1 to 8 will have 4 bricks. Columns 0 and 9 remain empty. In phase 1, the bricks are removed from columns 4 to 8 until each of them has 1 brick, and column 9 remains empty. Columns 0 to 3, which are out of the given range, remain unchanged. Phase 2 makes no change since columns 3 to 6 do not have more than 5 bricks. After phase 3 the numbers of bricks in columns 0, 4, and 5 increase to 3. There are 5 bricks in column 2 after phase 4. Phase 5 removes all bricks from columns 6 and 7.



TASK

Given the description of the k phases, please calculate the number of bricks in each column after all phases are finished.

INPUT

The first line contains two integers n and k : the number of columns of the wall and the number of phases ($1 \leq n \leq 2,000,000$; $1 \leq k \leq 500,000$).

Line $2 + i$ ($0 \leq i \leq k - 1$) contains four integers `op[i]`, `left[i]`, `right[i]` and `height[i]`:

- `op[i]` is the type of phase i : 1 for an adding phase and 2 for a removing phase.
- the range of columns in phase i starts with column `left[i]` and ends with column `right[i]` (including both endpoints `left[i]` and `right[i]`). You will always have `left[i] ≤ right[i]`.
- `height[i]` is the height parameter of phase i .

The height parameters of all phases are nonnegative integers less or equal to 100,000.

OUTPUT

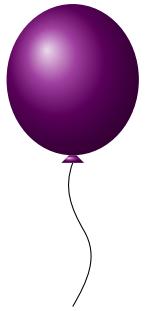
Output n lines: line $1 + i$ ($0 \leq i \leq n - 1$) contains the final number of bricks in column i .

SAMPLES

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

Performance Review

TIME LIMIT: 2s
MEMORY LIMIT: 256MB



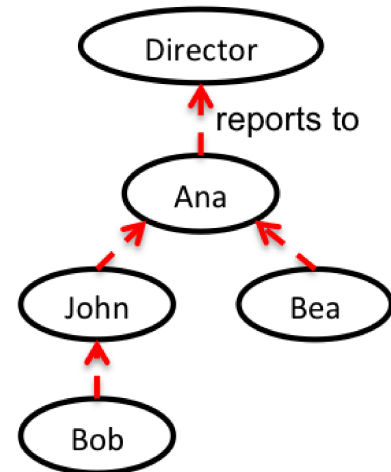
Employee performance reviews are a necessary evil in any company. In a performance review, employees give written feedback about each other on the work done recently. This feedback is passed up to their managers which then decide promotions based on the feedback received.

Maria is in charge of the performance review system in the engineering division of a famous company. The division follows a typical structure. Each employee (except the engineering director) reports to one manager and every employee reports directly or indirectly to the director.

Having the managers assessing the performance of their direct reports has not worked very well. After thorough research, Maria came up with a new performance review system. The main idea is to complement the existing corporate structure with a technical rank for each employee. An employee should give feedback only about subordinates with lower technical level.

Hence, the performance review will work as follows. Employees prepare a summary of their work, estimate how much time it takes to review it, and then request their superiors with higher technical rank to review their work.

Maria is very proud of this new system, but she is unsure if it will be feasible in practice. She wonders how much time each employee will waste writing reviews. Can you help her out?



TASK

Given the corporate structure of the engineering division, determine how much time each employee will spend writing performance reviews.

INPUT

The first line of input has one integer E , the number of employees, who are conveniently numbered between 1 and E . The next E lines describe all the employees, starting at employee 1 until employee E . Each line contains three space-separated integers m_i r_i t_i , the manager, the technical rank and the expected time to perform the review of employee i . The engineering director has no manager, represented with $m_i = -1$. The other employees have m_i between 1 and E .

CONSTRAINTS

- $1 \leq E \leq 100\,000$ Number of employees
- $1 \leq r_i \leq 100\,000$ Technical rank of each employee
- $1 \leq t_i \leq 100\,000$ Expected time to perform each review

PERFORMANCE REVIEW

OUTPUT

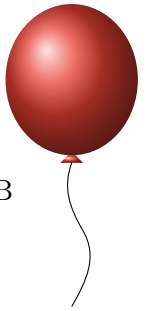
The output contains E lines. Line i has the time employee i will spend writing reviews.

SAMPLES

Sample input 1	Sample output 1
5 4 4 80 1 1 40 -1 10 60 3 5 50 4 8 70	40 0 240 120 0

Divine Gifting

TIME LIMIT: 5.0s
MEMORY LIMIT: 2048MB



A celestial scroll, detailing Zeus's latest whim unfurls before Hermes: the next few millenia will be a period of divine gifting for mortals. Hermes, the messenger god, is tasked with delivering these gifts. Not just any gifts, mind you, but exquisitely crafted items from the Olympian workshops: a lyre that plays melodies of pure joy, a quill that writes words of profound wisdom, and so on. Each of the N gifts is unique, and, to complicate matters, each has an optimal delivery date, a day when its magic would be most potent. But a divine law forbids delivering gifts before their optimal delivery day, lest mortals become complacent and entitled. Of course, all gifts must be delivered.

Adding to the challenge, Hermes, despite being the fastest god on Olympus, is always extremely busy. Between managing the celestial postal service and refereeing chariot races, he knows that he can only dedicate at most K days for the deliveries (each of those days, Hermes can deliver any number of gifts). Furthermore, late deliveries incur penalties: for each gift, the penalty is the square of the difference between its actual delivery day and its optimal delivery day. If a lyre is delivered a day or two late, a village might experience a few hours of slightly off-key music. A minor inconvenience, to be sure. However, if the lyre is delivered a month or a year late, the consequences are far more dire: a full year of discordant melodies, enough to drive even the most stoic musician to madness. The potential for chaos is immense.

And this is where you come in, mortal friend. Hermes, with his myriad responsibilities, could use a helping hand. Can you help him plan his divine calendar by determining the best days for delivering the gifts, so as to minimize the sum of late delivery penalties?

INPUT

On the first line of the input, two space-separated integers:

- N , the number of gifts;
- K , the maximum number of days Hermes dedicates to gifting.

On the second line, N space-separated integers d_i , representing the optimal delivery date of each gift.

OUTPUT

A single line of space-separated integers, where the i -th element is the day when Hermes should deliver the i -th gift. If there are multiple optimal solutions, they will all be accepted.

LIMITS

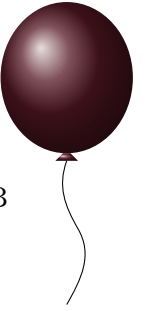
- $1 \leq N \leq 5,000$
- $1 \leq K \leq 20$
- $0 \leq d_i \leq 1,000,000$ for all $i \leq N$

SAMPLES

Sample input 1	Sample output 1
5 2 50 0 51 10 50	51 10 51 10 51

Yaxchilán Maze

TIME LIMIT: 5.0s
MEMORY LIMIT: 2048MB



Years spent deciphering Mayan glyphs had led Dr Wood, a famous archeologist, to this very moment: inside a chamber of the Yaxchilán maze, in the middle of the Lacandon Jungle, the last undiscovered Mayan codex appeared before her. It was beautiful, protected by jaguar pelt and adorned with jade. However, as soon as she touched it, all the corridors of the maze closed. She was now a prisoner of the maze. Worse, she was not alone, but had a full team of archeologists with her that were also held prisoner in different chambers of the maze.

As she examined the codex, a series of intricate diagrams caught her eye. They depicted the maze, its corridors morphing into different configurations, each aligned with a specific position of the sun. Another set of diagrams showed intricately carved holes in the walls together with a chilling depiction of a giant wasp. A realization dawned on her – the Maya had designed this maze to change with the sun's position, and some chambers were booby-trapped with a giant nest of deadly wasps.

Namely, there are N chambers in the maze numbered $0, \dots, N - 1$. Dr Wood and the members of her team start in chambers $0, 1, \dots, A - 1$ (one person per chamber). There are E exits: chambers $N - E, \dots, N - 1$. Initially, the maze contains no open corridors. At each hour (00:00, 01:00, 02:00 and so on, represented by integers $0, 1, 2$ and so on), a new corridor between two chambers opens. This corridor stays open for exactly M hours minus one minute. Corridors are bi-directional.

Among the N chambers, B are booby-trapped. The trap of a chamber triggers as soon as it is connected to at least K other chambers. When triggered, a gigantic swarm of deadly wasps appears in the chamber and immediately spreads to all chambers connected to the booby-trapped chamber. Furthermore, as time progresses, the wasps never disappear from a chamber, and worse, they continue to spread instantaneously from chambers that contain wasps to newly connected chambers. Two chambers are considered connected at a given time when there exists a path of one or multiple open corridors that allows going from one chamber to the other.

Thanks to the codex and to her knowledge of the maze (and smartphones!), Dr Wood and each member of her team have full information on the maze at hour 0, including all future events: in particular, they each know where they are in the maze, where corridors lead to, the exact time, when corridors open and close and which chambers they connect, the location of the booby-traps, exits, K , and can deduce which chambers are or will be filled with deadly wasps. All the archeologists can move freely and independently from each other using the open corridors at all times. They run fast and can move to any reachable chamber in less than 59 minutes. If an archeologist ends up in a chamber filled with deadly wasps, he or she dies and cannot exit the maze.

The exit chambers behave as the other chambers: they can be filled with wasps and be booby-trapped.

As soon as an archeologist reaches any exit chamber not filled with wasps, she or he exits the maze and its dangers. Can you tell the earliest time when each archeologist can exit the maze?

INPUT

- The first line contains the integer A , the number of archeologists.
- The second line contains the integer N .
- The third line contains the integer M .
- The fourth line contains the integer E .
- The fifth line contains the integer T , the number of hours at which corridors open.
- The sixth line contains B , followed by the list of the B space-separated booby-trapped chambers b_i .
- The seventh line contains K .
- In the next T lines, line t contains two space-separated integers u_t, v_t representing a corridor that opens at hour t and connects chambers u_t and v_t . t goes from 0 to $T - 1$ (inclusive). Multiple corridors can connect the same two chambers. A corridor can connect a chamber to itself.

OUTPUT

The output should contain A lines. The i -th line represents the i -th archeologist. If the i -th archeologist can exit the maze, it should be the integer representing the earliest hour at which she or he can exit. Otherwise, it should be “IMPOSSIBLE” (without quotes).

LIMITS

- $1 \leq A \leq 50$;
- $2 \leq N \leq 50\,000$;
- $1 \leq M \leq 100\,000$;
- $1 \leq E \leq 100$;
- $A + E \leq N$ (no chamber is both a starting chamber and an exit chamber);
- $1 \leq T \leq 500\,000$;
- $0 \leq B \leq N$;
- $0 \leq b_i < N$ for $i = 0, \dots, B - 1$;
- The b_i 's are unique;
- $0 \leq K < N$;

- $0 \leq u_t < N$ for $t = 0, \dots, T - 1$;
- $0 \leq v_t < N$ for $t = 0, \dots, T - 1$;

SAMPLES

Sample input 1	Sample output 1
1 5 1 1 5 0 0 0 1 0 2 2 3 2 4 0 4	3

Explanation of sample 1.

There are no booby-traps. Dr Wood (the only archeologist) starts at 0 and the exit is 4. After the corridor 0 2 opens, she goes from chamber 0 to chamber 2. She stays there until the corridor 2 4 opens, which allows her to reach chamber 4 and exit at hour 3.

Sample input 2	Sample output 2
1 4 1 1 3 0 0 0 1 2 3 1 2	IMPOSSIBLE

Explanation of sample 2.

There are no booby-traps. Dr Wood (the only archeologist) starts at 0 and the exit is 3. Unfortunately, Dr Wood can only reach chamber 2 after the corridor 2 3 leading to the exit closed. She cannot exit the maze.

Sample input 3	Sample output 3
4 10 2 2 11 2 3 7 2 0 1 1 2 2 3 1 4 4 9 5 6 0 6 5 7 1 6 7 8 6 8	9 9 9 IMPOSSIBLE

Explanation of sample 3.

The fastest path to an exit for the first archeologist starting at 0 is to go to chamber 5 at hour 6 using corridors 0 6 and 5 6, then to chamber 7 at hour 7 using the corridor 5 7, and finally reach chamber 8 (an exit) at hour 9 using the corridor 7 8. Note that chamber 7 is booby-trapped, but the wasps only appear at hour 10, after the first archeologist exited the maze.

The second archeologist starting at 1 can move to chamber 0 at hour 0 and then follow the path of the first archeologist.

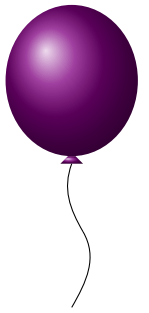
The third archeologist starting at 2 can move to chamber 0 at hour 1 and then follow the first two archeologists.

The last archeologist starting at 3 is killed by the wasps at hour 2.

At the end, chambers 1, 2, 3, 4, 6, 7, 8, 9 are filled with wasps.

Toll (hard version)

TIME LIMIT: 0.5s
MEMORY LIMIT: 256MB



A trucking company wants to optimize its internal processes—which mainly means saving money. The company serves a region where a toll must be paid for every single street. Each street connects two places (cities, villages etc.) directly. The company serves a set of orders; each order tells them to carry goods from one place to another. When serving an order, the company wants to pay the minimum overall toll. As the region's street network can be modeled by a graph where each edge has a specific cost (the toll for the respective street), the company actually wants to know (the cost of) the cheapest path between two nodes in this graph. However, the region's street network graph has an interesting property: it is directed (i.e. all streets are oneway), and there can only be an edge from a to b if $\lfloor b/K \rfloor = 1 + \lfloor a/K \rfloor$ (for some constant K). Write a program that for each of a given list of orders outputs the minimum toll the company has to pay to serve the respective order.

INPUT

The first line contains four integers: K (with the meaning described above), N (the number of places), M (the number of streets), and O (the number of orders). Each of the next M lines contains three integers a, b, t ($0 \leq a, b < N$). This means that there is a (oneway) street from a to b with toll t . You are guaranteed that $\lfloor b/K \rfloor = 1 + \lfloor a/K \rfloor$ is satisfied, and that no two locations are connected by more than one street. Finally O lines follow, each containing two integers a, b : this means that there is an order to carry goods from place a to place b .

Constraints are the following: $1 \leq N \leq 50\,000$, $1 \leq O \leq 10\,000$ and $K \leq 5$. Moreover, we have $0 \leq a < b < N$ for all orders a, b and $1 \leq t \leq 10\,000$ for all tolls t .

OUTPUT

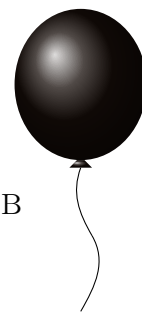
Your output should consist of O lines, each with one integer. The i -th line should contain the toll on a cheapest path between the two places in order i . If no such path exists, output -1 in this line.

SAMPLES

Sample input 1	Sample output 1
5 14 5 5 0 5 9 5 12 10 0 7 7 7 12 8 4 7 10 0 12 0 5 0 7 7 12 0 13	15 9 7 8 -1

D Crossing the Railways

TIME LIMIT: 4.0s
MEMORY LIMIT: 2048MB

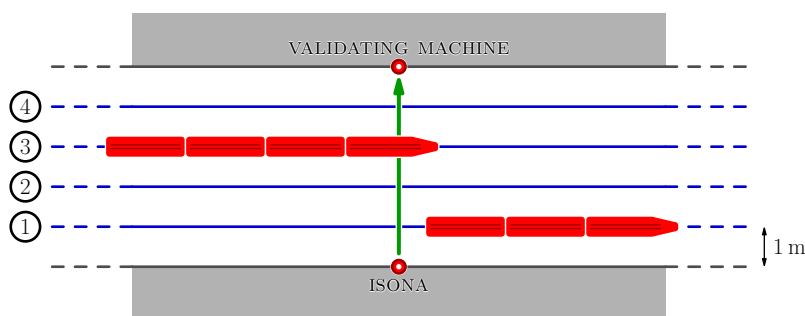


Isona is in a train station. This station has two platforms, and between them there are m parallel railways that can be viewed as infinite straight lines. Each railway is identified with an integer from 1 to m , railway 1 being the closest to the first platform and railway m being the farthest. There is a 1 meter distance between consecutive railways, as well as between each platform and its closest railway.

Isona is standing on the inner border of the first platform, when she realizes that she forgot to validate her ticket! There is a validating machine on the second platform, exactly opposite her current position (thus, the distance between Isona and the validating machine is $m + 1$ meters). There are only s seconds left to validate the ticket and the bridge designated to cross the railways is too far from the validating machine. Therefore, Isona (who is very brave and a little bit careless) will cross the railways running in a straight line perpendicular to the railways themselves. Isona can only run forward (not backward) and **she can stay still**. When she runs at maximum speed, she needs v seconds to traverse 1 meter. She can run at any speed less than or equal to her maximum speed.

There is only one problem: n trains are programmed to transit through the railways. The i -th train will use the railway r_i . It will start crossing the straight line between Isona and the validating machine a_i seconds from now and it will end b_i seconds from now. Of course, Isona cannot cross a railway when a train is passing. Formally, for every $i = 1, 2, \dots, n$, Isona is not allowed to be on railway r_i at any time t with $a_i < t < b_i$ (but she is allowed to cross at times a_i or b_i).

The following picture summarizes the situation. In the picture there are $m = 4$ railways and two trains are visible; the train going through railway 3 is currently crossing the line between Isona and the validating machine.



Isona is a really good runner, but she gets tired every time she has to change her running speed. What is the minimum number of speed changes she has to perform to get to the validating machine on the other platform within s seconds from now? Note that at the beginning Isona is not running. She can start to run anytime. The instant she starts to run (i.e. her speed becomes positive) is **not** counted as a speed change.

INPUT

The first line of the input contains four integers n, m, s, v ($1 \leq n \leq 500, 1 \leq m \leq 10, 1 \leq s, v \leq 10^9$) — the number of trains, the number of railways, the maximum time in seconds Isona can spend crossing the railways, and the number of seconds she needs to traverse 1 meter at maximum speed.

Each of the next n lines contains three integers a_i, b_i, r_i ($1 \leq a_i < b_i \leq 10^9, 1 \leq r_i \leq m$) — the start and end times of the i -th train crossing the straight line between Isona and the validating machine, and the railway it will be using.

It is guaranteed that, for any two trains i and j that go through the same railway (i.e. $r_i = r_j$), there is at least 1 second between them (that is, either $a_j \geq b_i + 1$ or $a_i \geq b_j + 1$).

OUTPUT

Print the minimum number of speed changes Isona has to perform to get to the validating machine in time. If this is impossible, print -1 .

SAMPLES

Sample input 1	Sample output 1
4 3 5 1 1 2 1 3 4 1 2 3 2 3 4 3	0

Explanation of sample 1.

If Isona starts running at time $t = 0$ at maximum speed (1 m/s), she will cross each railway just when a train is about to traverse it, and she will arrive at the other platform at time $4 = s - 1$ without changing speed.

Sample input 2	Sample output 2
3 3 12 2 2 10 1 1 6 2 8 12 3	2

Explanation of sample 2.

A possible solution with 2 speed changes is the following: for the first 2 seconds Isona goes at maximum speed (0.5 m/s), then she slows down to 0.25 m/s for 4 seconds until she reaches the second railway. At that point, she goes at maximum speed again until she reaches the other platform.

Sample input 3	Sample output 3
8 4 13 2 1 4 1 5 13 1 1 5 2 6 13 2 1 9 3 10 13 3 1 10 4 11 13 4	2

Explanation of sample 3.

Isona can wait 2 seconds before starting running. She then runs for 5 seconds at maximum speed (0.5 m/s). After that, she waits for 1 second not running (or running at 0 m/s), and finally she runs again at maximum speed for the last 5 seconds. Overall, she changes speed twice.

Sample input 4	Sample output 4
1 1 2 2 1 2 1	-1