

Problem A. Abstract Paintings

Time limit: 2 seconds

After visiting the ancient mosaics in Paphos, an artist decides to create several black-and-white patterns. Each pattern should be a rectangular grid of n rows and m columns, satisfying the following properties.

The top and bottom boundaries of the grid are connected to each other, just as the left and right boundaries are, making a torus shape. Formally, let us index rows from 0 to $n - 1$ top to bottom, and columns from 0 to $m - 1$ left to right. A cell (x, y) is at the intersection of row x and column y . Then:

- cells (x, y) and $((x + 1) \bmod n, y)$ are vertically adjacent,
- cells (x, y) and $(x, (y + 1) \bmod m)$ are horizontally adjacent.

Each cell is painted either black or white. A painting is valid if all of the following conditions hold:

- No two vertically or horizontally adjacent cells are both black.
- For every cell (i, j) , at least one of the three vertically adjacent cells (i, j) , $((i + 1) \bmod n, j)$, and $((i + 2) \bmod n, j)$ is black.
- All white cells form one connected region: from any white cell, it is possible to reach any other by moving between adjacent white cells.

Count the number of different valid paintings modulo 998 244 353. Two paintings are different if the colors of at least one cell differ.

Input

The only line contains two integers n and m ($3 \leq n, m \leq 10^6$), the number of rows and columns, respectively.

Output

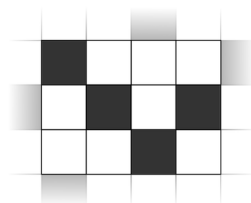
Print the number of valid paintings modulo 998 244 353.

Examples

standard input	standard output
3 3	6
3 4	18
6 6	0

Note

In the second example, there are 18 valid paintings. One of them is shown below.



Problem B. Bus Problem

Time limit: 6 seconds

A tour company in Athens wants to buy the longest possible bus, but they are afraid that the bus will not be able to move around the parked cars.

There are n parallel horizontal lanes, numbered from 1 to n from top to bottom, and m stationary cars. Car i occupies $[l_i, r_i]$ on lane k_i . Cars on the same lane may touch at endpoints, but their interiors do not intersect.

A bus of positive integer length b occupies $[x, x + b]$ on one lane, where x may be any real number. It starts on any lane, entirely to the left of all cars, and must finish entirely to their right.

The bus may move any distance to the right along its lane, or change to an adjacent lane while keeping $[x, x + b]$ unchanged. A lane change is a single move, allowed only if the bus fits on both lanes at that position. The bus cannot move left, turn, leave the highway, or drive horizontally between lanes.

Throughout every move, the bus may touch cars at endpoints, but must not overlap a car by a positive length. Help the tour company find the maximum possible bus length.

Input

The first line contains two integers n and m ($1 \leq n \leq 200\,000$, $0 \leq m \leq 200\,000$), the numbers of lanes and cars.

Each of the next m lines contains three integers k_i , l_i , and r_i ($1 \leq k_i \leq n$, $0 \leq l_i < r_i \leq 10^9$), describing a car.

The interiors of cars on the same lane do not intersect. Cars are given in arbitrary order.

Output

Print **INF** if arbitrarily large bus lengths are possible. Otherwise, print the maximum possible positive integer length, or 0 if no positive length works.

Examples

standard input	standard output
2 2 1 1 3 2 6 8	3
3 2 1 0 100 3 0 100	INF
2 2 1 0 1 2 0 1	0

Note

In the first example, a bus of length 3 starts on lane 2. When it occupies $[3, 6]$, it can change to lane 1, touching both cars at endpoints, and continue right. A longer bus cannot fit between the cars during a lane change.

In the second example, lane 2 is empty, so any bus length works.

In the third example, the cars block both lanes over the same interval, so no bus can pass.

Problem C. Coprime Pairs

Time limit: 2 seconds

You are given a tree with n vertices numbered from 1 to n , rooted at vertex 1. Vertex i has a positive integer a_i written on it.

For every vertex x , count the unordered pairs of distinct vertices u and v in the subtree of x such that $\gcd(a_u, a_v) = 1$.

The subtree of x includes x and all its descendants; $\gcd$ denotes the greatest common divisor.

Input

The first line contains an integer n ($1 \leq n \leq 10^5$), the number of vertices.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^5$).

Each of the next $n - 1$ lines contains two integers u_i and v_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$), the endpoints of an edge. The edges form a tree. They may be listed in any order; vertex numbers do not specify the parent-child relationship.

Output

Print the answer for each vertex from 1 to n , one per line.

Examples

standard input	standard output
8 6 1 10 1 15 7 6 14 1 2 1 3 2 4 2 5 3 6 6 7 6 8	18 3 2 0 0 1 0 0
3 6 10 15 1 2 2 3	0 0 0
4 1 1 1 1 1 3 3 2 2 4	6 1 3 0

Note

In the first example, the subtree of vertex 2 has values 1, 1, 15, so all three pairs qualify. The subtrees of vertices 3 and 6 have two and one suitable pairs, respectively.

In the second example, $\gcd(6, 10, 15) = 1$, but no pair of values is coprime.

In the third example, the subtree sizes are 4, 2, 3, 1; every pair of distinct vertices qualifies.

Problem D. Distinct Orders

Time limit: 2 seconds

You regularly visit a taverna in Heraklion for meze, a selection of small dishes. To keep the dinners varied, the cook alternates between two recipes for each dish.

The menu offers n dishes, numbered from 1 to n . Each dish has two versions: the first and the second.

Each day, you choose an integer k from 1 to n and order one portion of each of the first k dishes: dishes $1, 2, \dots, k$. You cannot choose their versions. The first time you order a dish, you receive its first version. On every subsequent occasion, you receive the opposite version from the last time you ordered that dish. Days when you do not order a dish do not affect its version.

Two daily orders are identical if they contain the same number of dishes and the version of each corresponding dish is the same.

Construct a sequence $k_1, k_2, \dots, k_m$ of maximum possible length such that no two daily orders are identical.

Input

The only line contains an integer n ($1 \leq n \leq 20$), the number of dishes.

Output

In the first line, print an integer m , the maximum possible number of days.

In the second line, print m integers $k_1, k_2, \dots, k_m$ ($1 \leq k_i \leq n$), the number of dishes to order on each day.

If there are several optimal sequences, print any of them.

Examples

standard input	standard output
1	2 1 1
2	6 2 2 1 2 2 1

Note

In the first example, order the only dish twice. You receive its first version and then its second version.

In the second example, the sequence $2, 2, 1, 2, 2, 1$ produces the following orders. The numbers in parentheses are the versions of dishes $1, 2, \dots, k$:

$$(1, 1), \quad (2, 2), \quad (1), \quad (2, 1), \quad (1, 2), \quad (2).$$

All six orders are distinct.

Problem E. Eurovision Lights

Time limit: 2 seconds

A Cypriot dance group is rehearsing a routine to *Fuego*, the song performed by Eleni Foureira for Cyprus at Eurovision 2018. For the lighting effect, the stage lamps will be switched on one by one.

There are n lamps in a row, numbered from 1 to n from left to right. Initially, all lamps are off. At each step, exactly one lamp that is still off is switched on. After n steps, all lamps are on.

A lit segment is a maximal nonempty group of consecutive lamps that are on. For example, if lamps 2, 3, and 5 are on in a row of 5 lamps, there are two lit segments: $[2, 3]$ and $[5, 5]$.

For every i from 1 to n , you are given a number a_i : the required number of lit segments immediately after step i . Find any order of switching on the lamps that gives these numbers, or determine that no such order exists.

Input

The first line contains one integer n ($1 \leq n \leq 1000$), the number of lamps.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($0 \leq a_i \leq n$), the required numbers of lit segments.

Output

If no suitable order exists, print -1 .

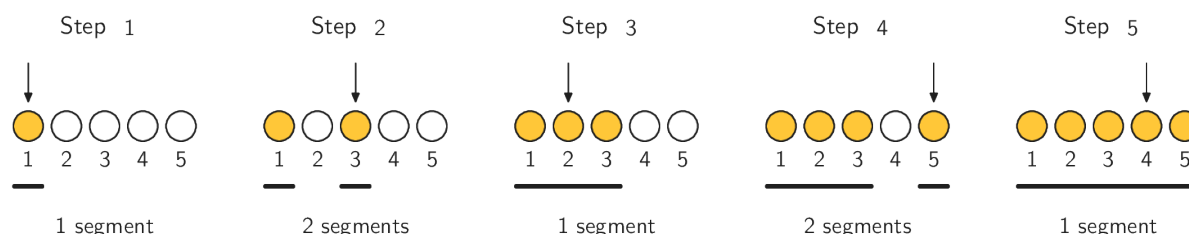
Otherwise, print a permutation $p_1, p_2, \dots, p_n$ of the integers from 1 to n , where p_i is the lamp switched on at step i . If there are several suitable orders, print any of them.

Examples

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

Note

In the first example, lamps are switched on in the order 1, 3, 2, 5, 4. Filled circles are lit lamps, arrows mark the newly lit lamp, and bars show the lit segments.



In the second example, the final number of lit segments must be 1, so the required sequence is impossible. In the third example, the only lamp is switched on.

Problem F. Flamingo Lake

Time limit: 6 seconds

Every winter, flamingos visit Larnaca Salt Lake in Cyprus. As rain fills the lake, more shallow water becomes available for the birds. To study how the flamingos spread out, consider the following simplified model. Initially, a rectangular lake consists of a single cell, and that cell contains a flamingo.

There are n events. In each event, the lake expands either by one row along its entire southern side, or by one column along its entire eastern side.

For each newly added cell, a flamingo arrives if none of the old cells sharing a side or a corner with it contains a flamingo. Here, old cells are the cells that belonged to the lake before this event. All new cells are checked simultaneously: flamingos arriving during the same event do not affect each other. Flamingos that have already arrived stay in their cells.

Help scientists find the total number of flamingos at the lake after all events.

Input

The first line contains an integer n ($1 \leq n \leq 10^6$), the number of events.

The second line contains a string of n characters. The i -th character is **S** if the i -th event adds a row to the south, or **E** if it adds a column to the east.

Output

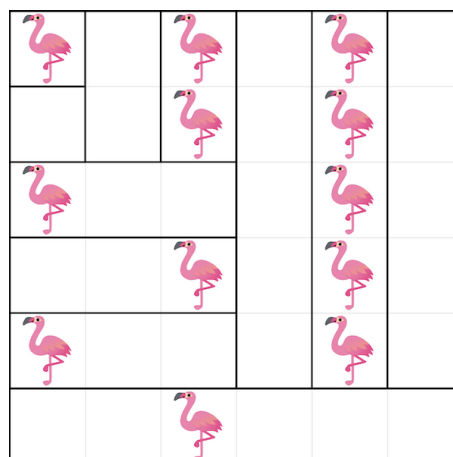
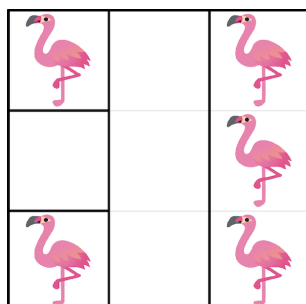
Print one integer: the number of flamingos at the lake after all events.

Examples

standard input	standard output
4 SSEE	5
10 SEESSSEES	12

Note

Illustrations for the examples:



Problem G. Gas Stations

Time limit: 1 second

A group of friends is planning a road trip from Patras to Thessaloniki. They have chosen their main route, but visiting a gas station may require a detour.

Their car travels along a main route from position 0 to position d . On a full tank, it can travel l kilometers. Initially, its tank is full.

There are n refueling stations. Station i is connected to the main route at position p_i . To visit it, the car leaves the route, travels $a_i/2$ kilometers to the station, fills its tank completely, and travels $a_i/2$ kilometers back to the same position on the main route. Thus, visiting this station adds a_i kilometers to the journey. Fuel is consumed on both parts of the detour, at the same constant rate as on the main route.

The car moves only forward along the main route. The positions of the stations it visits must strictly increase. The car may arrive at a station or at the destination with an empty tank. The distance traveled between consecutive refuelings, as well as before the first and after the last refueling, must not exceed l .

Find the minimum total distance the car must travel to reach position d , including all detours. If reaching the destination is impossible, report this.

Input

The first line contains an integer d ($1 \leq d \leq 10^9$), the length of the main route in kilometers.

The second line contains an integer l ($1 \leq l \leq 10^6$), the distance in kilometers that the car can travel on a full tank.

The third line contains an integer n ($0 \leq n \leq 10^5$), the number of stations.

Each of the next n lines contains two integers p_i and a_i ($1 \leq p_i \leq d - 1$, $0 \leq a_i \leq 10^6$), the position of station i on the main route and the total length of its detour. The stations are given in strictly increasing order of position: $p_1 < p_2 < \dots < p_n$. If $d = 1$, then $n = 0$.

Output

Print one integer: the minimum total distance traveled, or -1 if the destination cannot be reached.

Examples

standard input	standard output
300 150 4 100 2 130 5 230 30 270 5	310
20 10 1 12 0	-1
10 10 0	10

Problem H. Hacking the Traffic Lights

Time limit: 4 seconds

A student team in Limassol is tired of being late because the traffic lights on their way to the university are poorly synchronized.

There are n traffic lights along their path. The car starts at an integer time $x \geq 0$ and reaches the i -th light exactly d_i seconds later, without stopping.

All lights have the same integer period c . Originally, light i is green at an integer time t if $(t - s_i) \bmod c$ belongs to $\{0, 1, \dots, g_i - 1\}$. Remainders are taken between 0 and $c - 1$: the green interval may wrap around the period, and $g_i = c$ means the light is always green.

The team has found a backdoor in the road department's server. They can shift each light's phase independently in either direction. Choose a nonnegative integer shift magnitude $\delta_i \in \{0, 1, \dots, c - 1\}$ for each light, and replace s_i with either $(s_i + \delta_i) \bmod c$ or $(s_i - \delta_i) \bmod c$.

Choose the starting time and phase shifts so that the car passes every light on green. Minimize $\sum_{i=1}^n \delta_i$, the sum of the shift magnitudes.

Input

The first line contains two integers n and c ($1 \leq n \leq 200\,000$, $1 \leq c \leq 10^9$), the number of traffic lights and their common period.

Each of the next n lines contains three integers d_i , s_i , and g_i ($1 \leq d_i \leq 10^9$, $0 \leq s_i < c$, $1 \leq g_i \leq c$), describing the i -th light. The travel times satisfy $d_1 < d_2 < \dots < d_n$.

Output

Print one integer: the minimum sum of the shift magnitudes.

Examples

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

Note

In the first example, start at $x = 0$ and shift the second light by 5 seconds. Both lights are then green at the arrival times, 1 and 2. The total cost is 5.

In the second example, start at $x = 6$. The arrival remainders are 0, 3, and 7, so no shifts are needed.

In the third example, start at $x = 4$, shift the first light by 2 seconds and the third by -1 second, and leave the second light unchanged. The total cost is 3.

Problem I. Improvised Handles

Time limit: 2 seconds

A craft shop in Ioannina keeps n boxes tightly packed in a single row inside an open-top organizer. The boxes have the same width, and box i has height h_i .

A box can only be removed by lifting it straight up. To lift a box without a handle, you must grip both of its sides near the top. You can do this if and only if each of its two immediate neighbors in the original row has already been removed or is strictly shorter than the chosen box. There is no obstacle beyond either end of the row.

After a box is removed, its place stays empty. The other boxes do not move, and boxes separated by an empty place do not become neighbors. You cannot tilt the boxes or slide them sideways.

Before removing any boxes, you may attach a paper tab to the top of any box. A box with a tab can be lifted out at any time, regardless of its neighbors' heights. Each tab belongs to one box and cannot be moved to another.

Find the minimum number of tabs that allow you to remove all boxes, choosing the removal order yourself.

Input

The first line contains an integer n ($1 \leq n \leq 200\,000$), the number of boxes.

The second line contains n integers $h_1, h_2, \dots, h_n$ ($1 \leq h_i \leq 10^9$), the heights of the boxes. Heights are not necessarily distinct.

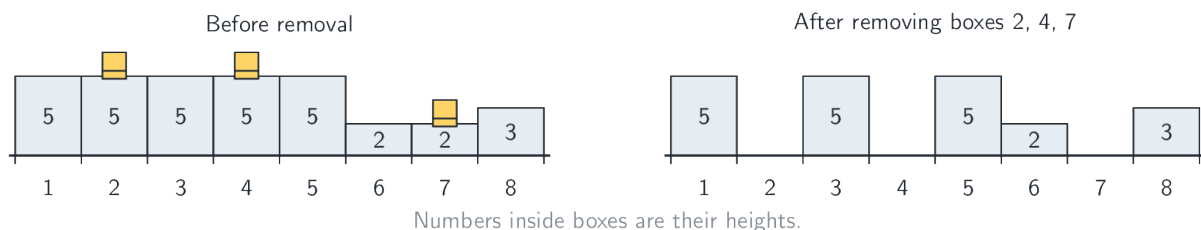
Output

Print one integer: the minimum number of paper tabs needed.

Examples

standard input	standard output
8 5 5 5 5 5 2 2 3	3
3 1 2 1	0
1 1000000000	0

Note



First example: attach tabs to boxes 2, 4, and 7, and remove these boxes.
Then remove boxes 1, 3, 5, 8, 6, in this order. Three tabs are necessary.

In the second example, no tabs are needed: remove boxes 2, 1, and 3, in this order. After box 2 is removed, boxes 1 and 3 do not become neighbors.

Problem J. Janitor's Cart

Time limit: 2 seconds

The organizers of the programming championship have printed the problem statements in several university offices. The university janitor must collect the statements and bring them to the staircase.

A staircase is at one end of a straight corridor with n offices. Office i is located i meters along the corridor from the staircase and contains a_i printed statements.

You need to move all the statements to the staircase using a cart that can carry at most c statements at a time. Each trip starts at the staircase with an empty cart and ends at the staircase, where all collected statements are unloaded.

During a trip, you may collect statements from several offices. Statements from the same office may be transported on different trips.

Find the minimum total distance traveled by the cart.

Input

The first line contains two integers n and c ($1 \leq n \leq 200\,000$, $1 \leq c \leq 10^9$): the number of offices and the capacity of the cart.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($0 \leq a_i \leq 10^6$): the numbers of statements in the offices.

Output

Print one integer: the minimum total distance, in meters, needed to move all the statements to the staircase.

Examples

standard input	standard output
3 3 2 0 2	8
2 2 0 5	12
4 10 0 0 0 0	0

Note

In the first example, one trip collects two statements from office 3 and one from office 1. Its length is 6 meters. Another trip collects the remaining statement from office 1 and has length 2 meters. The total distance is 8 meters.

In the second example, office 2 has five statements. Three trips of length 4 meters each are needed, so the answer is 12.

In the third example, there are no statements, so no trips are needed.

Problem K. Kavala Olives

Time limit: 4 seconds

At a taverna in Kavala, a cook is arranging olives on small plates for dinner. The cook wants to distribute the olives so that no plate contains more than one.

There are n plates arranged in a circle and numbered from 1 to n in clockwise order. Plate i initially contains a_i olives. The total number of olives does not exceed n .

In one move, you may take one olive from a plate and put it onto the next plate clockwise or counter-clockwise. In particular, plates 1 and n are adjacent. There is no limit on the number of olives on a plate during the process.

Find the minimum number of moves needed to make every plate contain at most one olive.

Input

The first line contains an integer n ($1 \leq n \leq 200\,000$), the number of plates.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($0 \leq a_i \leq n$), the initial numbers of olives. It is guaranteed that $a_1 + a_2 + \dots + a_n \leq n$.

Output

Print one integer: the minimum number of moves.

Examples

standard input	standard output
6 3 0 0 0 0 0	2
5 1 0 1 0 1	0
5 0 3 2 0 0	4

Note

In the first example, move one olive from plate 1 to plate 2, and another olive from plate 1 to plate 6. Two moves are necessary because two olives must leave plate 1.

In the second example, every plate already contains at most one olive.

In the third example, move two olives from plate 2 to plate 1, one at a time. Then move one olive from plate 1 to plate 5, and one from plate 3 to plate 4. This takes four moves; a plate may temporarily contain more than one olive.

Problem L. Lost Cargo

Time limit: 2 seconds

This is an interactive problem.

A team of divers is surveying the cargo of the *Zenobia*, a shipwreck off the coast of Larnaca. In one section of its cargo deck, rectangular containers completely cover an $n \times m$ grid. Each container occupies exactly two cells sharing a side, and each cell belongs to exactly one container.

The boundaries between the containers are hidden by sediment. The divers can inspect individual cells to identify the containers. Each inspection takes time, so the number of inspections is limited.

Rows are numbered from 1 to n from top to bottom, and columns from 1 to m from left to right. You can inspect a cell (x, y) . The reply describes which half of its container occupies the cell:

- U: the upper half of a vertical container; its other half is at $(x + 1, y)$;
- D: the lower half of a vertical container; its other half is at $(x - 1, y)$;
- L: the left half of a horizontal container; its other half is at $(x, y + 1)$;
- R: the right half of a horizontal container; its other half is at $(x, y - 1)$.

Reconstruct the arrangement of all containers using at most $\lfloor nm/4 \rfloor$ queries. The arrangement is fixed before the interaction starts and does not change in response to your queries.

Interaction Protocol

First you need to read two integers n and m ($1 \leq n, m \leq 100$), the number of rows and columns. It is guaranteed that nm is even.

To inspect a cell, print a line of the form `? x y`, where $1 \leq x \leq n$ and $1 \leq y \leq m$. The interactor replies with one of U, D, L, or R on a separate line. Repeated queries are allowed, but each counts toward the limit.

When you have reconstructed the tiling, print a line containing `!`, followed by n lines of m characters each. Character y of line x must be the half occupying cell (x, y) , using the same U, D, L, and R notation. Your answer must match the hidden tiling exactly. Printing the final answer does not count as a query. Then terminate your program.

After printing each query and after printing the final answer, flush the output buffer. For example, use `cout.flush()` in C++ or `sys.stdout.flush()` in Python.

If you make an invalid query or exceed the query limit, your solution will be judged incorrect. If you read `-1` or reach the end of input while expecting a reply, terminate immediately.

Example

standard input	standard output
3 2	? 2 2
D	!
	UU
	DD
	LR

Problem M. Museum Tickets

Time limit: 2 seconds

A group of n friends wants to visit the Acropolis Museum in Athens.

The museum offers two types of tickets. An individual ticket costs a euros and admits one person. A group ticket costs b euros and admits all n friends.

What is the minimum total cost of tickets that allow all the friends to enter the museum?

Input

The first line contains an integer n ($1 \leq n \leq 1000$): the number of friends.

The second line contains an integer a ($1 \leq a \leq 1000$): the price of an individual ticket, in euros.

The third line contains an integer b ($1 \leq b \leq 1000$): the price of a group ticket, in euros.

Output

Print one integer: the minimum total cost, in euros.

Examples

standard input	standard output
4 8 25	25
3 7 30	21
5 6 30	30

Note

In the first example, four individual tickets cost 32 euros, while a group ticket costs 25 euros. Buy a group ticket.

In the second example, three individual tickets cost 21 euros, which is cheaper than a group ticket.

In the third example, both options cost 30 euros.