Codeforces Round #165 (Div. 2)

A. Fancy Fence

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

Emuskald needs a fence around his farm, but he is too lazy to build it himself. So he purchased a fence-building robot.

He wants the fence to be a regular polygon. The robot builds the fence along a single path, but it can only make fence corners at a single angle a.

Will the robot be able to build the fence Emuskald wants? In other words, is there a regular polygon which angles are equal to a?

Input
The first line of input contains an integer t ($0 < t < 180$) — the number of tests. Each of the following t lines contains a single integer a ($0 < a < 180$) — the angle the robot can make corners at measured in degrees.

Output
For each test, output on a single line "YES" (without quotes), if the robot can build a fence Emuskald wants, and "NO" (without quotes), if it is impossible.

Examples

input
3 30 60 90
output
NO YES YES

Note
In the first test case, it is impossible to build the fence, since there is no regular polygon with angle 30°.

In the second test case, the fence is a regular triangle, and in the last test case — a square.

B. Multithreading

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

Emuskald is addicted to Codeforces, and keeps refreshing the main page not to miss any changes in the "recent actions" list. He likes to read thread conversations where each thread consists of multiple messages.

Recent actions shows a list of n different threads ordered by the time of the latest message in the thread. When a new message is posted in a thread that thread jumps on the top of the list. No two messages of different threads are ever posted at the same time.

Emuskald has just finished reading all his opened threads and refreshes the main page for some more messages to feed his addiction. He notices that no new threads have appeared in the list and at the i-th place in the list there is a thread that was at the a_i-th place before the refresh. He doesn't want to waste any time reading old messages so he wants to open only threads with new messages.

Help Emuskald find out the number of threads that **surely** have new messages. A thread x surely has a new message if there is no such sequence of thread updates (posting messages) that both conditions hold:

1. thread x is not updated (it has no new messages);
2. the list order 1, 2, ..., n changes to a_1, a_2, ..., a_n.

Input

The first line of input contains an integer n, the number of threads ($1 \le n \le 10^5$). The next line contains a list of n space-separated integers a_1, a_2, ..., a_n where a_i ($1 \le a_i \le n$) is the old position of the i-th thread in the new list. It is guaranteed that all of the a_i are distinct.

Output

Output a single integer — the number of threads that surely contain a new message.

Examples

input
5 5 2 1 3 4
output
2

input
3 1 2 3
output
0

input
4 4 3 2 1
output
3

Note

In the first test case, threads 2 and 5 are placed before the thread 1, so these threads must contain new messages. Threads 1, 3 and 4 may contain no new messages, if only threads 2 and 5 have new messages.

In the second test case, there may be no new messages at all, since the thread order hasn't changed.

In the third test case, only thread 1 can contain no new messages.

C. Magical Boxes

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

Emuskald is a well-known illusionist. One of his trademark tricks involves a set of magical boxes. The essence of the trick is in packing the boxes inside other boxes.

From the top view each magical box looks like a square with side length equal to 2^k (k is an integer, $k \geq 0$) units. A magical box v can be put inside a magical box u, if side length of v is strictly less than the side length of u. In particular, Emuskald can put 4 boxes of side length 2^{k-1} into one box of side length 2^k, or as in the following figure:

Emuskald is about to go on tour performing around the world, and needs to pack his magical boxes for the trip. He has decided that the best way to pack them would be inside another magical box, but magical boxes are quite expensive to make. Help him find the smallest magical box that can fit all his boxes.

Input

The first line of input contains an integer n ($1 \leq n \leq 10^5$), the number of different sizes of boxes Emuskald has. Each of following n lines contains two integers k_i and a_i ($0 \leq k_i \leq 10^9$, $1 \leq a_i \leq 10^9$), which means that Emuskald has a_i boxes with side length 2^{k_i}. It is guaranteed that all of k_i are distinct.

Output

Output a single integer p, such that the smallest magical box that can contain all of Emuskald's boxes has side length 2^p.

Examples

input
2 0 3 1 5
output
3

input
1 0 4
output
1

input
2 1 10 2 2
output
3

Note

Picture explanation. If we have 3 boxes with side length 2 and 5 boxes with side length 1, then we can put all these boxes inside a box with side length 4, for example, as shown in the picture.

In the second test case, we can put all four small boxes into a box with side length 2.

D. Greenhouse Effect

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

Emuskald is an avid horticulturist and owns the world's longest greenhouse — it is effectively infinite in length.

Over the years Emuskald has cultivated n plants in his greenhouse, of m different plant species numbered from 1 to m. His greenhouse is very narrow and can be viewed as an infinite line, with each plant occupying a single point on that line.

Emuskald has discovered that each species thrives at a different temperature, so he wants to arrange $m - 1$ borders that would divide the greenhouse into m sections numbered from 1 to m from left to right with each section housing a single species. He is free to place the borders, but in the end all of the i-th species plants must reside in i-th section from the left.

Of course, it is not always possible to place the borders in such way, so Emuskald needs to replant some of his plants. He can remove each plant from its position and place it anywhere in the greenhouse (at **any** real coordinate) with no plant already in it. Since replanting is a lot of stress for the plants, help Emuskald find the minimum number of plants he has to replant to be able to place the borders.

Input

The first line of input contains two space-separated integers n and m ($1 \le n, m \le 5000$, $n \ge m$), the number of plants and the number of different species. Each of the following n lines contain two space-separated numbers: one integer number s_i ($1 \le s_i \le m$), and one real number x_i ($0 \le x_i \le 10^9$), the species and position of the i-th plant. Each x_i will contain no more than 6 digits after the decimal point.

It is guaranteed that all x_i are different; there is at least one plant of each species; the plants are given in order "from left to the right", that is in the ascending order of their x_i coordinates ($x_i < x_{i+1}, 1 \le i < n$).

Output

Output a single integer — the minimum number of plants to be replanted.

Examples

input
3 2 2 1 1 2.0 1 3.100
output
1

input
3 3 1 5.0 2 5.5 3 6.0
output
0

input
6 3 1 14.284235 2 17.921382 1 20.328172 3 20.842331 1 25.790145 1 27.204125
output
2

Note

In the first test case, Emuskald can replant the first plant to the right of the last plant, so the answer is 1.

In the second test case, the species are already in the correct order, so no replanting is needed.

E. Flawed Flow

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

Emuskald considers himself a master of flow algorithms. Now he has completed his most ingenious program yet — it calculates the maximum flow in an undirected graph. The graph consists of n vertices and m edges. Vertices are numbered from 1 to n. Vertices 1 and n being the source and the sink respectively.

However, his max-flow algorithm seems to have a little flaw — it only finds the flow volume for each edge, but not its direction. Help him find for each edge the direction of the flow through this edges. Note, that the resulting flow should be correct maximum flow.

More formally. You are given an undirected graph. For each it's undirected edge (a_i, b_i) you are given the flow volume c_i. You should direct all edges in such way that the following conditions hold:

1. for each vertex v $(1 < v < n)$, sum of c_i of incoming edges is equal to the sum of c_i of outcoming edges;
2. vertex with number 1 has no incoming edges;
3. the obtained directed graph **does not have cycles**.

Input

The first line of input contains two space-separated integers n and m ($2 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$), the number of vertices and edges in the graph. The following m lines contain three space-separated integers a_i, b_i and c_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$, $1 \le c_i \le 10^4$), which means that there is an undirected edge from a_i to b_i with flow volume c_i.

It is guaranteed that there are no two edges connecting the same vertices; the given graph is connected; a solution always exists.

Output

Output m lines, each containing one integer d_i, which should be 0 if the direction of the i-th edge is $a_i \to b_i$ (the flow goes from vertex a_i to vertex b_i) and should be 1 otherwise. The edges are numbered from 1 to m in the order they are given in the input.

If there are several solutions you can print any of them.

Examples

input
3 3 3 2 10 1 2 10 3 1 5

output
1 0 1

input
4 5 1 2 10 1 3 10 2 3 5 4 2 15 3 4 5

output
0 0 1 1 0

Note

In the first test case, 10 flow units pass through path $1 \to 2 \to 3$, and 5 flow units pass directly from source to sink: $1 \to 3$.