VK Cup 2018 - Round 2

A. Mystical Mosaic

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

There is a rectangular grid of n rows of m initially-white cells each.

Arkady performed a certain number (possibly zero) of operations on it. In the i-th operation, a non-empty subset of rows R_i and a non-empty subset of columns C_i are chosen. For each row r in R_i and each column c in C_i, the intersection of row r and column c is coloured black.

There's another constraint: a row or a column can only be chosen at most once among all operations. In other words, it means that no pair of (i, j) $(i < j)$ exists such that $R_i \cap R_j \neq \varnothing$ or $C_i \cap C_j \neq \varnothing$, where $\cap$ denotes intersection of sets, and $\varnothing$ denotes the empty set.

You are to determine whether a valid sequence of operations exists that produces a given final grid.

Input

The first line contains two space-separated integers n and m $(1 \leq n, m \leq 50)$ — the number of rows and columns of the grid, respectively.

Each of the following n lines contains a string of m characters, each being either '`.`' (denoting a white cell) or '`#`' (denoting a black cell), representing the desired setup.

Output

If the given grid can be achieved by any valid sequence of operations, output "`Yes`"; otherwise output "`No`" (both without quotes).

You can print each character in any case (upper or lower).

Examples

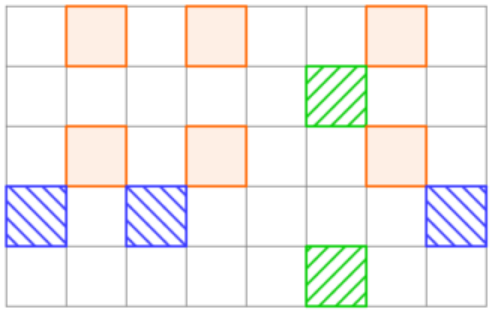
input
5 8 .#.#..#. .....#.. .#.#..#. #.#....# .....#..
output
Yes

input
5 5 ..#.. ..#.. ##### ..#.. ..#..
output
No

input
5 9 ....# #........ ..##.#... ....#. ....#.#.#
output
No

Note

For the first example, the desired setup can be produced by 3 operations, as is shown below.



For the second example, the desired setup cannot be produced, since in order to colour the center row, the third row and all columns must be selected in one operation, but after that no column can be selected again, hence it won't be possible to colour the other cells in the center column.

B. Three-level Laser

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

An atom of element X can exist in n distinct states with energies $E_1 < E_2 < \ldots < E_n$. Arkady wants to build a laser on this element, using a three-level scheme. Here is a simplified description of the scheme.

Three distinct states i, j and k are selected, where $i < j < k$. After that the following process happens:

1. initially the atom is in the state i,
2. we spend E_k - E_i energy to put the atom in the state k,
3. the atom emits a photon with useful energy E_k - E_j and changes its state to the state j,
4. the atom spontaneously changes its state to the state i, losing energy E_j - E_i,
5. the process repeats from step 1.

Let's define the energy conversion efficiency as $\eta = \frac{E_k - E_j}{E_k - E_i}$, i. e. the ration between the useful energy of the photon and spent energy.

Due to some limitations, Arkady can only choose such three states that E_k - $E_i \le U$.

Help Arkady to find such the maximum possible energy conversion efficiency within the above constraints.

Input

The first line contains two integers n and U ($3 \le n \le 10^5$, $1 \le U \le 10^9$) — the number of states and the maximum possible difference between E_k and E_i.

The second line contains a sequence of integers E_1, E_2, ..., E_n ($1 \le E_1 < E_2 ... < E_n \le 10^9$). It is guaranteed that all E_i are given in increasing order.

Output

If it is not possible to choose three states that satisfy all constraints, print `-1`.

Otherwise, print one real number η — the maximum possible energy conversion efficiency. Your answer is considered correct its absolute or relative error does not exceed 10^{-9}.

Formally, let your answer be a, and the jury's answer be b. Your answer is considered correct if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-9}$.

Examples

input
4 4 1 3 5 7
output
0.5

input
10 8 10 13 15 16 17 19 20 22 24 25
output
0.875

input
3 1 2 5 10
output

```
-1
```

Note

In the first example choose states 1, 2 and 3, so that the energy conversion efficiency becomes equal to $\eta = \frac{5-3}{5-1} = 0.5$.

In the second example choose states 4, 5 and 9, so that the energy conversion efficiency becomes equal to $\eta = \frac{24-17}{24-16} = 0.875$.

C. Riverside Curio

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

Arkady decides to observe a river for n consecutive days. The river's water level on each day is equal to some real value.

Arkady goes to the riverside each day and makes a mark on the side of the channel at the height of the water level, but if it coincides with a mark made before, no new mark is created. The water does not wash the marks away. Arkady writes down the number of marks strictly above the water level each day, on the i-th day this value is equal to m_i.

Define d_i as the number of marks strictly under the water level on the i-th day. You are to find out the minimum possible sum of d_i over all days. There are no marks on the channel before the first day.

Input

The first line contains a single positive integer n ($1 \le n \le 10^5$) — the number of days.

The second line contains n space-separated integers $m_1, m_2, \ldots, m_n$ ($0 \le m_i < i$) — the number of marks strictly above the water on each day.

Output

Output one single integer — the minimum possible sum of the number of marks strictly below the water level among all days.

Examples

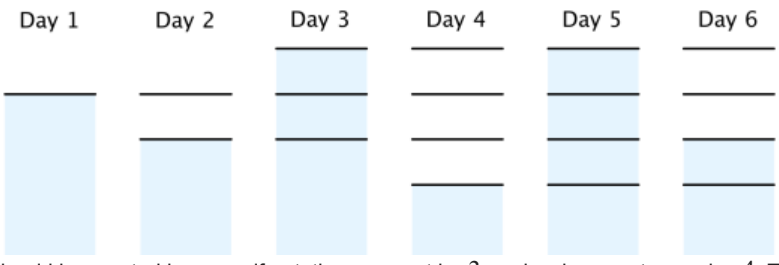
input
6 0 1 0 3 0 2
output
6

input
5 0 1 2 1 2
output
1

input
5 0 1 1 2 2
output
0

Note

In the first example, the following figure shows an optimal case.



Note that on day 3, a new mark should be created because if not, there cannot be 3 marks above water on day 4. The total number of marks underwater is $0 + 0 + 2 + 0 + 3 + 1 = 6$.

In the second example, the following figure shows an optimal case.

D. Contact ATC

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

Arkady the air traffic controller is now working with n planes in the air. All planes move along a straight coordinate axis with Arkady's station being at point 0 on it. The i-th plane, small enough to be represented by a point, currently has a coordinate of x_i and is moving with speed v_i. It's guaranteed that $x_i \cdot v_i < 0$, i.e., all planes are moving towards the station.

Occasionally, the planes are affected by winds. With a wind of speed v_{wind} (not necessarily positive or integral), the speed of the i-th plane becomes $v_i + v_{wind}$.

According to weather report, the current wind has a steady speed falling inside the range $[-w, w]$ (inclusive), but the exact value cannot be measured accurately since this value is rather small — smaller than the absolute value of speed of any plane.

Each plane should contact Arkady at the exact moment it passes above his station. And you are to help Arkady count the number of pairs of planes (i, j) $(i < j)$ there are such that there is a possible value of wind speed, under which planes i and j contact Arkady at the same moment. This value needn't be the same across different pairs.

The wind speed is the same for all planes. You may assume that the wind has a steady speed and lasts arbitrarily long.

Input

The first line contains two integers n and w $(1 \le n \le 100\,000,\ 0 \le w < 10^5)$ — the number of planes and the maximum wind speed.

The i-th of the next n lines contains two integers x_i and v_i $(1 \le |x_i| \le 10^5,\ w + 1 \le |v_i| \le 10^5,\ x_i \cdot v_i < 0)$ — the initial position and speed of the i-th plane.

Planes are pairwise distinct, that is, no pair of (i, j) $(i < j)$ exists such that both $x_i = x_j$ and $v_i = v_j$.

Output

Output a single integer — the number of unordered pairs of planes that can contact Arkady at the same moment.

Examples

input
5 1 -3 2 -3 3 -1 2 1 -3 3 -5
output
3

input
6 1 -3 2 -2 2 -1 2 1 -2 2 -2 3 -2
output
9

Note

In the first example, the following 3 pairs of planes satisfy the requirements:

- **(2, 5)** passes the station at time $3 / 4$ with $v_{wind} = 1$;
- **(3, 4)** passes the station at time $2 / 5$ with $v_{wind} = 1 / 2$;
- **(3, 5)** passes the station at time $4 / 7$ with $v_{wind} = -1 / 4$.

In the second example, each of the 3 planes with negative coordinates can form a valid pair with each of the other 3, totaling 9 pairs.

E. Wardrobe

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

Olya wants to buy a custom wardrobe. It should have n boxes with heights $a_1, a_2, \ldots, a_n$, stacked one on another in some order. In other words, we can represent each box as a vertical segment of length a_i, and all these segments should form a single segment from 0 to $\sum_{i=1}^{n} a_i$ without any overlaps.

Some of the boxes are important (in this case $b_i = 1$), others are not (then $b_i = 0$). Olya defines the *convenience* of the wardrobe as the number of important boxes such that their bottom edge is located between the heights l and r, inclusive.

You are given information about heights of the boxes and their importance. Compute the maximum possible convenience of the wardrobe if you can reorder the boxes arbitrarily.

Input

The first line contains three integers n, l and r ($1 \le n \le 10\,000$, $0 \le l \le r \le 10\,000$) — the number of boxes, the lowest and the highest heights for a bottom edge of an important box to be counted in convenience.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10\,000$) — the heights of the boxes. It is guaranteed that the sum of height of all boxes (i. e. the height of the wardrobe) does not exceed $10\,000$: Olya is not very tall and will not be able to reach any higher.

The second line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 1$), where b_i equals 1 if the i-th box is important, and 0 otherwise.

Output

Print a single integer — the maximum possible convenience of the wardrobe.

Examples

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

input
2 2 5 3 6 1 1
output
1

Note

In the first example you can, for example, first put an unimportant box of height 2, then put an important boxes of sizes 1, 3 and 2, in this order, and then the remaining unimportant boxes. The convenience is equal to 2, because the bottom edges of important boxes of sizes 3 and 2 fall into the range $[3, 6]$.

In the second example you have to put the short box under the tall box.

F. Minimal Subset Difference

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

We call a positive integer x a k-beautiful integer if and only if it is possible to split the multiset of its digits in the decimal representation into two subsets such that the difference between the sum of digits in one subset and the sum of digits in the other subset is **less than or equal to** k. Each digit should belong to exactly one subset after the split.

There are n queries for you. Each query is described with three integers l, r and k, which mean that you are asked how many integers x between l and r (inclusive) are k-beautiful.

Input

The first line contains a single integer n ($1 \le n \le 5 \cdot 10^4$), indicating the number of queries.

Each of the next n lines describes a query, containing three integers l, r and k ($1 \le l \le r \le 10^{18}$, $0 \le k \le 9$).

Output

For each query print a single number — the answer to the query.

Examples

input
10 1 100 0 1 100 1 1 100 2 1 100 3 1 100 4 1 100 5 1 100 6 1 100 7 1 100 8 1 100 9

output
9 28 44 58 70 80 88 94 98 100

input
10 1 1000 0 1 1000 1 1 1000 2 1 1000 3 1 1000 4 1 1000 5 1 1000 6 1 1000 7 1 1000 8 1 1000 9

output
135 380 573 721 830 906 955 983 996 1000

Note

If $1 \le x \le 9$, integer x is k-beautiful if and only if $x \le k$.

If $10 \le x \le 99$, integer $x = 10a + b$ is k-beautiful if and only if $|a - b| \le k$, where a and b are integers between 0 and 9, inclusive.

100 is k-beautiful if and only if $k \ge 1$.