Problem A. Social Distancing I

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

A terrible new disease, COWVID-19, has begun to spread among cows worldwide. Farmer John is trying to take as many precautions as possible to protect his herd from infection.

Farmer John's barn is a long narrow building containing N stalls in a row ($2 \leq N \leq 10^5$). Some of these stalls are currently occupied by cows, and some are vacant. Having read about the importance of "social distancing", Farmer John wants to maximize D, where D is the distance between the closest two occupied stalls. For example, if stalls 3 and 8 are the closest that are occupied, then $D = 5$.

Two new cows recently joined Farmer John's herd and he needs to decide to which formerly-unoccupied stalls they should be assigned. Please determine how he can place his two new cows so that the resulting value of D is still as large as possible. Farmer John cannot move any of his existing cows; he only wants to assign stalls to the new cows.

Input

The first line of input contains N. The next line contains a string of length N of 0s and 1s describing the sequence of stalls in the barn. 0s indicate empty stalls and 1s indicate occupied stalls. The string has at least two 0s, so there is at least enough room for two new cows.

Output

Please print the largest value of D (the closest distance between two occupied stalls) that Farmer John can achieve after adding his two new cows in an optimal fashion.

Scoring

- Test cases 2-6 satisfy $N \leq 10$.
- Test cases 7-8 satisfy $N \leq 100$.
- Test cases 9-11 satisfy $N \leq 5000$.
- Test cases 12-15 satisfy no additional constraints.

Example

standard input	standard output
14 10001001000010	2

Note

In this example, Farmer John could add cows to make the occupancy string look like 10x010010x0010, where x's indicate the new cows. In this case $D = 2$. It is impossible to add the new cows to achieve any higher value of D.

Problem credits: Brian Dean

Problem B. A Game with Grundy

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

Grundy is playing his favourite game — hide and seek.

His N friends stand at locations on the x-axis of a two-dimensional plane - the i-th one is at coordinates $(x_i, 0)$. Each friend can see things in a triangular wedge extending vertically upwards from their position — the i-th friend's triangular wedge of vision will be specified by two lines: one with slope of v_i/h_i and the other with slope $-v_i/h_i$. A friend cannot see a point that lies exactly on one of these two lines.

Grundy may choose to hide at any location (a, Y), where a is an integer satisfying $L \leq a \leq R$, and L, R, and Y are given integer constants.

Each possible location may be in view of some of Grundy's friends (namely, strictly within their triangular wedge of vision).

Grundy would like to know in how many different spots he can stand such that he will be in view of at most i of his friends, for every possible value of i from 0 to N.

Input

The first line of input contains the integer N ($1 \leq N \leq 10^5$).

The next line contains three integers: L, R and Y ($-10^9 \leq L \leq R \leq 10^9, 1 \leq Y \leq 10^6$).

Each of the next N lines contain three integers: the i-th such line contains x_i, v_i and h_i ($L \leq x_i \leq R, 1 \leq v_i, h_i \leq 100$). The slopes v_i/h_i and $-v_i/h_i$ define the triangular wedge of vision for friend i.

Output

The output is $N+1$ lines, where each line i ($0 \leq i \leq N$) contains the integer number of positions in which Grundy can stand and be in view of at most i of his friends.

Scoring

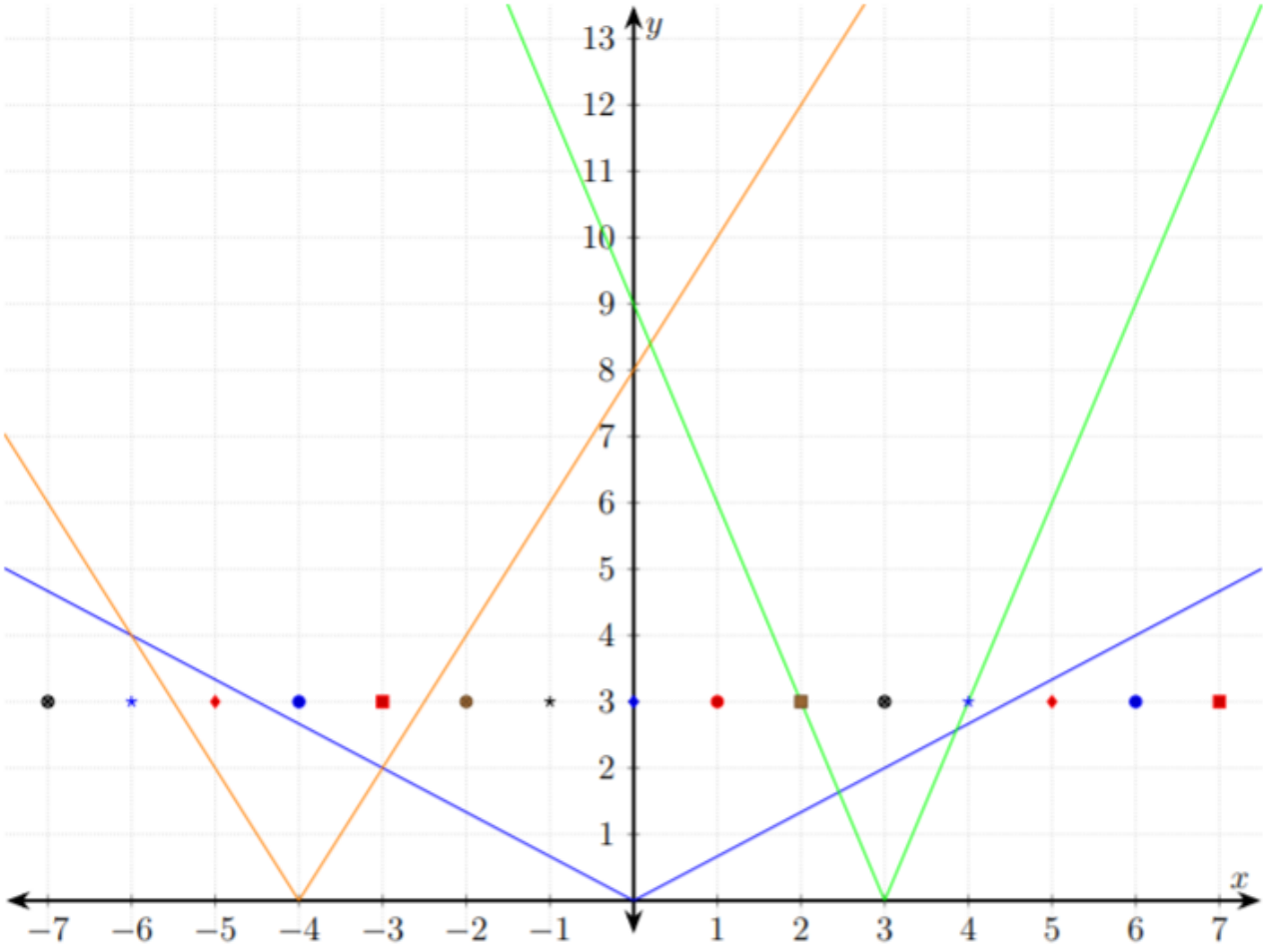
Subtask 1 (60 points): $-10^6 \leq L \leq R \leq 10^6$.
Subtask 2 (40 points): no additional constraints.

Example

standard input	standard output
3 -7 7 3 0 2 3 -4 2 1 3 3 1	5 12 15 15

Note

There are three friends with the following triangular wedges of vision, along with the possible positions that Grundy can be placed, as shown in the diagram below:



Notice the points $(2, 3)$ and $(4, 3)$ are visible only by the friend at position 0, since they lie on the boundary of the triangular wedge of vision for the friend at position 3.

Problem C. Exercise Deadlines

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

Bob has N programming exercises that he needs to complete before their deadlines. Exercise i only takes one time unit to complete, but has a deadline d_i ($1 \le d_i \le N$) time units from now.

Bob will solve the exercises in an order described by a sequence $a_1, a_2, \ldots, a_N$, such that a_1 is the first exercise he solves, a_2 is the second exercise he solves, and so on. Bob's original plan is described by the sequence $1, 2, \ldots, N$. With one *swap* operation, Bob can exchange two adjacent numbers in this sequence. What is the minimum number of swaps required to change this sequence into one that completes all exercises on time?

Input

The first line consists of a single integer N ($1 \le N \le 200\,000$). The next line contains N space-separated integers $d_1, d_2, \ldots, d_N$ ($1 \le d_i \le N$).

Output

Output a single integer, the minimum number of swaps required for Bob to solve all exercises on time, or "`-1`" if this is impossible.

Scoring

Subtask 1 (68 points): $N \le 5000$.
Subtask 2 (32 points): no additional constraints.

Examples

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

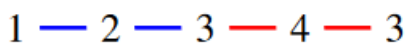
Note

In the first example, one valid sequence is $(1, 4, 3, 2)$, which can be obtained from $(1, 2, 3, 4)$ by three swaps.

Problem D. Travelling Salesperson

Input file:	`standard input`
Output file:	`standard output`
Time limit:	7 seconds
Memory limit:	256 megabytes

In the city of Newpolis, every pair of buildings is connected by a road, either red or blue. To switch from travelling along red roads to blue roads or vice versa costs one ticket. The length of a route is the number of buildings that are visited. For example, the following route has a length of five and costs one ticket:



If we wanted to travel on a blue road again after visiting vertex 3 for the second time, we would need another ticket, for a total of two tickets:

You are a travelling salesperson visiting the city of Newpolis, and you wish to visit each building at least once, while minimizing repeated visits of the same buildings. You have not yet decided which building you are starting your route from, so you would like to plan out all possible routes. Furthermore, you only have access to one ticket. For each building, you would like to find a route of minimum length that begins at that building, visits all the buildings at least once, and uses at most one ticket.

Input

The first line will contain a single integer n ($2 \le n \le 2000$), the number of buildings in Newpolis. Lines 2 to n each contain a string, with line i containing the string s_i, representing the colours of the roads connected to building i. The string $s_i = c_{i,1}c_{i,2}\dots c_{i,i-1}$ has a length of $i-1$ and consists only of the characters R and B. If $c_{i,j} = \text{R}$ then the road between buildings i and j is red. Otherwise, it is blue.

Output

Output $2n$ lines. Lines $2i-1$ for $1 \le i \le n$ should contain single integer m_i, representing the length of the travel plan starting at building i. Lines $2i$ from $1 \le i \le n$ should each contain m_i space separated integers, describing the order in which you visit the buildings, starting at building i.

Scoring

For every one of your travel plans, a score is computed. Let k_i be the length of the optimal route starting at each building, and let m_i be the length of your route. If $m_i > 2k_i$, then your score will be 0, and you will receive a verdict of Wrong Answer. If $m_i = k_i$, then your score will be 100. Otherwise you will a receive a score of $\lfloor 32 + 32 \cdot \frac{2k_i - m_i}{k_i - 1} \rfloor$. Your score for the test case is the minimum score for each travel plan.

If any of your plans are invalid, your score will be 0, and you will receive a verdict of Wrong Answer.

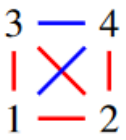
Your submission's score is the minimum score over all test cases.

Example

standard input	standard output
4 R RR BRB	4 1 2 3 4 4 2 1 3 4 4 3 2 1 4 4 4 3 2 1

Note

In the sample test Newpolis looks like this:



The route starting from building 3 has an optimal length of 4 by visiting the buildings in the order $3, 2, 1, 4$. The solution's route has a length of 5, meaning the score is equal to $\lfloor 32 + 32 \cdot \frac{2 \cdot 4 - 5}{4 - 1} \rfloor = 64$

Problem E. Interval collection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3.5 seconds
Memory limit:	512 megabytes

Alina is starting an interval collection. An interval is defined as two positive integers $[l, r]$ such that $l < r$. We say that the length of this interval is $r - l$. Additionally, we say that an interval $[l, r]$ contains another interval $[x, y]$ if $l \leq x$ and $y \leq r$. In particular, each interval contains itself.

For a non-empty set S of intervals, we define the set of common intervals as all the intervals that are contained within every interval $[l, r]$ in S. If the set of common intervals is non-empty, then we say the greatest common interval of S is equal to the common interval with the largest length. Notice that the common interval does not required to be in the set S.

For the same set S, we define the set of enclosing intervals as all the intervals that contain every interval $[l, r]$ in S. Note that this set is always non-empty, so we say the least enclosing interval of S is equal to the enclosing interval with the smallest length. Notice that the enclosing interval also does not required to be in the set S.

Initially, Alina owns no intervals in her collection. There are q events that change the set of intervals she owns.

The first type of event is when Alina adds an interval $[l, r]$ to her collection. Note that this interval could have the same l and r as another interval in her collection. They should be treated as separate intervals.

The second type of event is when Alina removes an existing interval $[l, r]$ from her collection. Note that if Alina has more than one interval with the same $[l, r]$, she removes exactly one of them.

After each event, Alina chooses an non-empty subset S of intervals she owns in her collection that satisfy the following conditions:

- Among all sets S Alina could choose, she chooses one that has no greatest common interval, if possible. If this is impossible, then she chooses one which has the length of its greatest common interval as small as possible.
- Among all sets S that satisfy the previous condition, she chooses one which has the length of its least enclosing interval as small as possible.

Your task is to determine the length of the least enclosing interval of the set S Alina chose after each event.

Input

The first line of input contains q ($1 \leq q \leq 500\,000$), the number of add and remove operations in total. The next q lines are in one of the following forms:

- A l r: add the interval $[l, r]$ to Alina's collection.
- R l r: remove one of the instances of the interval $[l, r]$ from Altina's collection. It is guaranteed the interval to be removed exists and that the collection will be non-empty after the interval is removed.

For all queries, $1 \leq l, r \leq 1\,000\,000$

Output

The output consists of q lines, each line containing the length of the least enclosing interval for Altina's choice of S as described in the problem description.

Scoring

Subtask 1 (12 points): $q \leq 500$.
Subtask 2 (32 points): $q \leq 12\,000$.
Subtask 3 (28 points): $q \leq 50\,000$.
Subtask 4 (16 points): At any time for any two intervals $[l_1, r_1]$ and $[l_2, r_2]$ in collection either $r_1 < l_2$ or $r_2 < l_1$.
Subtask 5 (12 points): no additional constraints.

Example

standard input	standard output
5 A 1 5 A 2 7 A 4 6 A 6 8 R 4 6	4 6 5 4 7

Note

After the interval $[1, 5]$ is added, there is only one interval, so $S = ([1, 5])$ is the only valid choice and the least enclosing interval is $[1, 5]$.

After the interval $[2, 7]$ is added, $S = ([1, 5], [2, 7])$ has the greatest common interval $[2, 5]$ and least enclosing interval $[1, 7]$.

After the interval $[4, 6]$ is added, $S = ([1, 5], [4, 6])$ has the greatest common interval $[4, 5]$ and least enclosing interval $[1, 6]$.

After the interval $[6, 8]$ is added, $S = ([4, 6], [6, 8])$ has no greatest common interval and its least enclosing interval $[4, 8]$. Note that $S = ([1, 5], [6, 8])$ also has no greatest common interval but its least enclosing interval $[1, 8]$ has a greater length than $[4, 8]$.

After the interval $[4, 6]$ is removed, $S = ([1, 5], [6, 8])$ has no greatest common interval and least enclosing interval $[1, 8]$.