2020 Canadian Computing Olympiad
Day 1, Problem 1
A Game with Grundy

Time Limit: 1 second

Problem Description
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 Specification
The first line of input contains the integer N ($1 \leq N \leq 100\,000$).

The next line contains three integers: L, R and Y ($-1\,000\,000\,000 \leq L \leq R \leq 1\,000\,000\,000, 1 \leq Y \leq 1\,000\,000$).

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

For 15 of the 25 marks available, $-1\,000\,000 \leq L \leq R \leq 1\,000\,000$.

Output Specification
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.

Sample Input
```
3
-7 7 3
0 2 3
```

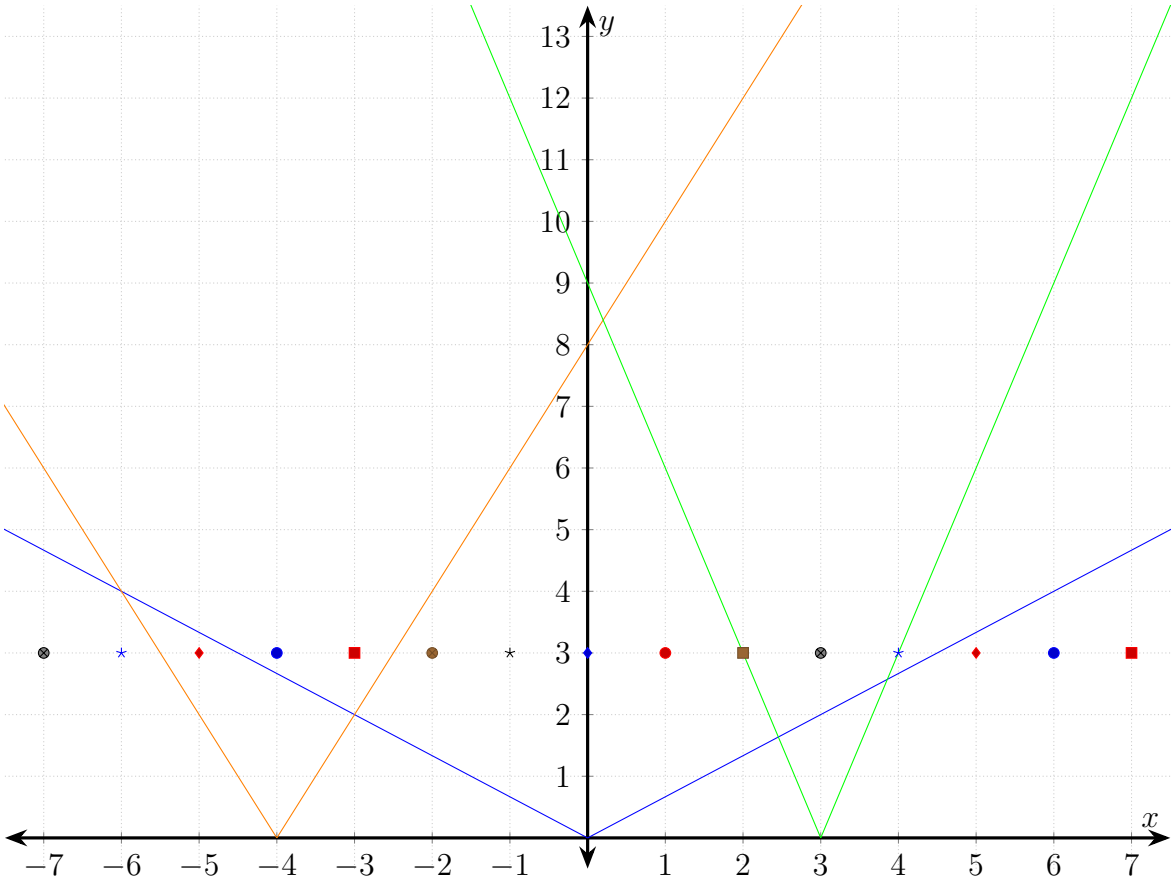
```
-4 2 1
3 3 1
```

Output for Sample Input

```
5
12
15
15
```

Explanation of Output for Sample Input

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.

2020 Canadian Computing Olympiad
Day 1, Problem 2
Exercise Deadlines

Time Limit: 1 second

Problem Description

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 ($1 \leq d_i \leq N$) 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 Specification

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

For 17 of the 25 marks available, $N \leq 5000$.

Output Specification

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

Sample Input 1

```
4
4 4 3 2
```

Output for Sample Input 1

```
3
```

Explanation of Output for Sample Input 1

One valid sequence is $(1, 4, 3, 2)$, which can be obtained from $(1, 2, 3, 4)$ by three swaps.

Sample Input 2

```
3
1 1 3
```

Output for Sample Input 2

```
-1
```

Explanation of Output for Sample Input 2

There are two exercises that are due at time 1, but only one exercise can be solved by this time.

2020 Canadian Computing Olympiad
Day 1, Problem 3
Mountains and Valleys

Time Limit: 7 seconds

Problem Description

You are planning a long hiking trip through some interesting, but well-known terrain. There are N interesting sites you would like to visit and M trails connecting pairs of sites. Each trail has a difficulty level indicated as a positive integer.

The trail system is a bit special, however. There are exactly $N-1$ trails with difficulty level 1 (these are completely flat trails), and the rest of the trails all have a difficulty level of at least $\left\lceil \frac{N}{3} \right\rceil$ (these are very mountainous trails). (The ceiling of x, denoted as $\lceil x \rceil$, is the smallest integer greater than or equal to x.)

Additionally, it is possible to travel between any two sites using only the trails with difficulty level 1.

You would like to visit every site, starting your walk from any site of your choice and finishing at some other site, such that you visit each site at least once and the total sum of difficulty levels is minimum among all walks. Note that walking a trail k times with difficulty level d contributes a value of $k \cdot d$ to the sum of difficulty levels.

Input Specification

The first line of input contains two space-separated integers N ($4 \leq N \leq 500\,000$) and M ($N - 1 \leq M \leq 2\,000\,000$).

The next M lines contain three space-separated integers x_i, y_i, and w_i describing the trail between sites x_i and y_i with difficulty level w_i ($1 \leq i \leq M$; $0 \leq x_i, y_i \leq N - 1$; $x_i \neq y_i$). Note that there is at most one trail between every pair of sites, and that $w_i = 1$ or $\left\lceil \frac{N}{3} \right\rceil \leq w_i \leq N$.

For 1 of the 25 marks available, $N \leq 6$ and $M \leq 10$.

For an additional 2 of the 25 marks available, $N \leq 20$ and $M \leq 40$.

For an additional 2 of the 25 marks available, $N \leq 5\,000$, $M \leq 10\,000$, and either $w_i = 1$ or $\left\lceil \frac{N}{2} \right\rceil \leq w_i \leq N$.

For an additional 6 of the 25 marks available, $N \leq 100$ and $M \leq 200$.

For an additional 2 of the 25 marks available, $N \leq 500$ and $M \leq 1\,000$.

For an additional 3 of the 25 marks available, $N \leq 5\,000$ and $M \leq 10\,000$.

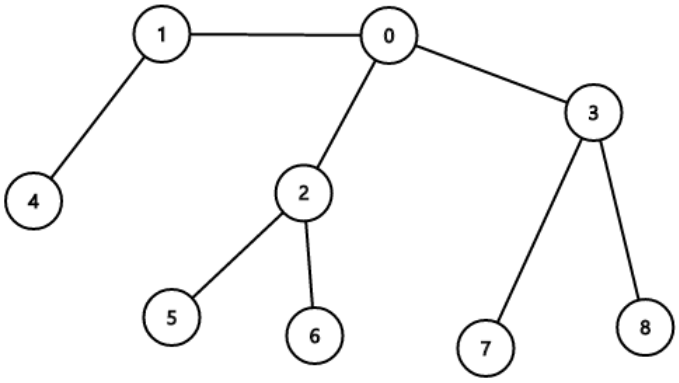
For an additional 5 of the 25 marks available, $N \leq 80\,000$ and $M \leq 160\,000$.

Output Specification
Output one integer, which is the minimum sum of difficulty levels taken for all trails walked to visit each site at least once.

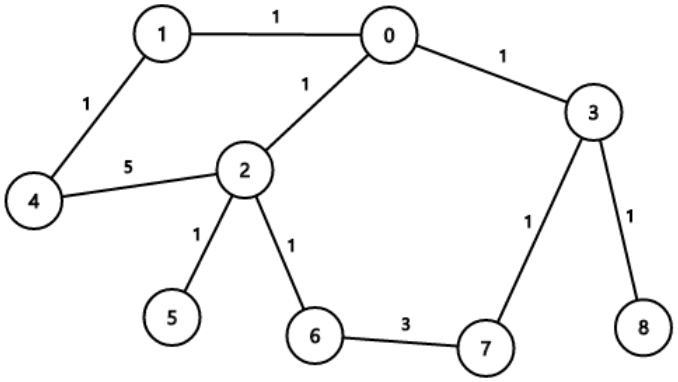
Sample Input
```
9 10
0 1 1
0 2 1
0 3 1
1 4 1
2 5 1
2 6 1
3 7 1
3 8 1
2 4 5
6 7 3
```

Output for Sample Input
```
11
```

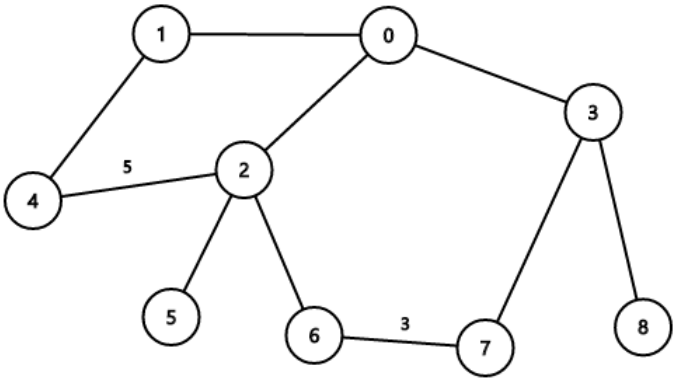
Explanation of Output for Sample Input
This is the set of flat trails:



This is the entire set of trails with all the difficulty levels.



This is the entire set of trails, with trails with difficulty level 1 being omitted.

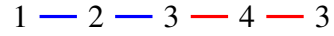


An optimal walk for this set of trails is $4 \to 1 \to 0 \to 2 \to 5 \to 2 \to 6 \to 7 \to 3 \to 8$ with a total cost of $1 + 1 + 1 + 1 + 1 + 1 + 3 + 1 + 1 = 11$. There is no way to make a walk that visits all the sites at least once with a lower total difficulty level cost.

2020 Canadian Computing Olympiad
Day 2, Problem 1
Travelling Salesperson

Time Limit: 7 seconds

Problem Description
In the city of RedBlue, 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 RedBlue, 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 Specification
The first line will contain a single integer N ($2 \leq N \leq 2\,000$), the number of buildings in RedBlue.

Lines 2 to N each contain a string, with line i containing the string C_i, representing the colours of the roads connected to building i. The string $C_i = C_{i,1}C_{i,2}...C_{i,i-1}$ has a length of $i - 1$ and consists only of the characters `R` and `B`. If $C_{i,j}$ is `R`, then the road between buildings i and j is red. Otherwise, it is blue.

Output Specification
Output $2N$ lines. Lines $2i - 1$ for $1 \leq i \leq N$ should contain a single integer M_i, representing the length of the travel plan starting at building i. Lines $2i$ for $1 \leq i \leq N$ 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 is greater than $2K_i$, then your score will be 0, and you will receive a verdict of Wrong Answer. If M_i is equal to K_i, then your score will be 25. Otherwise you will a receive a score of $\lfloor 8 + 8 \times \frac{2K_i - M_i}{K_i - 1} \rfloor$. 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.

Sample Input

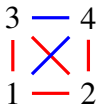
```
4
R
RR
BRB
```

Possible Output for Sample Input

```
5
1 4 2 1 3
6
2 3 1 2 3 4
5
3 1 2 3 4
4
4 3 1 2
```

Explanation of Possible Output for Sample Input

RedBlue 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 $\lfloor 8 + 8 \times \frac{2 \times 4 - 5}{4 - 1} \rfloor = 16$.

2020 Canadian Computing Olympiad
Day 2, Problem 2
Interval Collection

Time Limit: 3.5 seconds

Problem Description

Altina 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 $[x, y]$ 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.

For the same set S, we define the set of enclosing intervals as all the intervals $[x, y]$ 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.

Initially, Altina 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 Altina adds an interval $[l, r]$ to her collection. Note that this interval could have the same $[l, r]$ as another interval in her collection. They should be treated as separate intervals.

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

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

- Among all sets S Altina 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 Altina chose after each event.

Input Specification
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 Altina'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 subtasks, $1 \leq l < r \leq 1\,000\,000$.

For 3 of the 25 marks available, $Q \leq 500$.

For an additional 8 of the 25 marks available, $Q \leq 12\,000$.

For an additional 7 of the 25 marks available, $Q \leq 50\,000$.

For an additional 4 of the 25 marks available, the following condition holds after each event: for every two separate intervals $[l_1, r_1]$ and $[l_2, r_2]$ in Altina's collection, either $r_1 < l_2$ or $r_2 < l_1$.

Output Specification
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.

Sample Input
```
5
A 1 5
A 2 7
A 4 6
A 6 8
R 4 6
```

Output for Sample Input
```
4
6
5
4
7
```

Explanation of Output for Sample Input
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 $[1, 8]$.

2020 Canadian Computing Olympiad
Day 2, Problem 3
Shopping Plans

Time Limit: 2 seconds

Problem Description

You are shopping from a store that sells a total of N items. The i-th item has a *type* a_i which is an integer between 1 and M. A feasible shopping plan is a subset of these items such that for all types j, the number of items of type j is in the interval $[x_j, y_j]$.

The i-th item in the store has a cost of c_i, and the cost of a shopping plan is the sum of the costs of items in the plan. You are interested in the possible costs of feasible shopping plans. Find the costs of the K cheapest feasible shopping plans. Note that if there are two different shopping plans with the same cost, they should be counted separately in the output.

Input Specification

The first line consists of three space-separated integers N, M, and K ($1 \leq N, M, K \leq 200\,000$). N lines follow, the i-th of which contains two space-separated integers a_i and c_i ($1 \leq a_i \leq M$, $1 \leq c_i \leq 10^9$). M lines follow, the j-th of which contains two space-separated integers x_j and y_j ($0 \leq x_j \leq y_j \leq N$).

For 5 of the 25 marks available, $x_j = y_j = 1$ and $N, M, K \leq 4000$.

For an additional 5 of the 25 marks available, $x_j = y_j = 1$ and $N, M, c_i \leq 4000$.

For an additional 5 of the 25 marks available, $x_j = y_j = 1$.

For an additional 5 of the 25 marks available, $x_j = 0$.

Output Specification

Output K lines. On the i-th line, output the cost of the i-th cheapest feasible shopping plan, if one exists, or `-1` if there are fewer than i feasible shopping plans.

Sample Input 1

```
5 2 7
1 5
1 3
2 3
1 6
2 1
1 1
1 1
```

Output for Sample Input 1

```
4
6
6
7
8
9
-1
```

Explanation of Output for Sample Input 1

A feasible shopping plan must combine exactly one item with a cost in $\{5, 3, 6\}$ with exactly one item with a cost in $\{3, 1\}$.