Problem 1: Cow Lineup

Farmer John's N cows (1 <= N <= 100,000) are lined up in a row. Each cow is identified by an integer "breed ID" in the range 0...1,000,000,000; the breed ID of the ith cow in the lineup is B(i). Multiple cows can share the same breed ID.

FJ thinks that his line of cows will look much more impressive if there is a large contiguous block of cows that all have the same breed ID. In order to create such a block, FJ chooses up to K breed IDs and removes from his lineup all the cows having those IDs. Please help FJ figure out the length of the largest consecutive block of cows with the same breed ID that he can create by doing this.

INPUT FORMAT:

* Line 1: Two space-separated integers: N and K.

* Lines 2..1+N: Line i+1 contains the breed ID B(i).

SAMPLE INPUT:

```
9 1
2
7
3
7
7
3
7
5
7
```

INPUT DETAILS:

There are 9 cows in the lineup, with breed IDs 2, 7, 3, 7, 7, 3, 7, 5, 7. FJ would like to remove up to 1 breed ID from this lineup.

OUTPUT FORMAT:

* Line 1: The largest size of a contiguous block of cows with identical breed IDs that FJ can create.

SAMPLE OUTPUT:

```
4
```

OUTPUT DETAILS:

By removing all cows with breed ID 3, the lineup reduces to 2, 7, 7, 7, 7, 5, 7.
In this new lineup, there is a contiguous block of 4 cows with the same breed ID (7).

Problem 2: Painting the Fence

Farmer John has devised a brilliant method to paint the long fence next to his barn (think of the fence as a one-dimensional number line). He simply attaches a paint brush to his favorite cow Bessie, and then retires to drink a cold glass of water as Bessie walks back and forth across the fence, applying paint to any segment of the fence that she walks past.

Bessie starts at position 0 on the fence and follows a sequence of N moves (1 <= N <= 100,000). Example moves might be "10 L", meaning Bessie moves 10 units to the left, or "15 R", meaning Bessie moves 15 units to the right. Given a list of all of Bessie's moves, FJ would like to know what area of the fence gets painted with at least K coats of paint. Bessie will move at most 1,000,000,000 units away from the origin during her walk.

INPUT FORMAT:

* Line 1: Two space-separated integers: N and K.

* Lines 2..1+N: Each line describes one of Bessie's N moves (e.g., "15 L").

SAMPLE INPUT:

```
6 2
2 R
6 L
1 R
8 L
1 R
2 R
```

INPUT DETAILS:

Bessie starts at position 0 and moves 2 units to the right, then 6 to the left, 1 to the right, 8 to the left, and finally 3 to the right. FJ wants to know the area covered by at least 2 coats of paint.

OUTPUT FORMAT:

* Line 1: The total area covered by at least K coats of paint.

SAMPLE OUTPUT:

```
6
```

OUTPUT DETAILS:

6 units of area are covered by at least 2 coats of paint. This includes the intervals [-11,-8], [-4,-3], and [0,2].

Problem 3: Square Overlap

Farmer John is planning to build N (2 <= N <= 50,000) square fenced-in pastures on his farm, each of size exactly K x K (1 <= K <= 1,000,000). Pasture i is centered at point (x_i, y_i) with integer coordinates in the range -1,000,000...1,000,000. However, in his haste to complete his plans, FJ realizes that he may have accidentally placed two pastures in locations that overlap (by overlap, this means the two pastures share a positive area in common). No two pastures share the exact same center point.

Given the locations of each of the planned square pastures, please help FJ compute the area shared by the two overlapping pastures. Output zero if no two squares overlap, and -1 if overlap occurs between more than a single pair of pastures.

INPUT FORMAT:

* Line 1: Two space-separated integers, N and K. K is guaranteed to be even.

* Lines 2..1+N: Line i+1 contains the integers x_i and y_i, describing the center of the ith pasture.

SAMPLE INPUT (file squares.in):

```
4 6
0 0
8 4
-2 1
0 7
```

INPUT DETAILS:

There are 4 squares, each of size 6 x 6. The first square is centered at
(0,0), and so on.

OUTPUT FORMAT:

* Line 1: The area shared by the two overlapping squares. Output zero if no two squares overlap, and -1 if overlap occurs between more than a single pair of pastures.

SAMPLE OUTPUT:

```
20
```

OUTPUT DETAILS:

Pastures #1 and #3 overlap in 20 units of area.

Problem 4: Mirrors

Farmer John's cows have been causing too much trouble around the farm, and FJ therefore wants to keep a more watchful eye on them. By installing N reflective fences (1 <= N <= 200) at various locations on the farm, he hopes to be able to see from his house at location (0,0) to the barn at location (a,b).

On a 2D map of FJ's farm, fence i appears as a short line segment centered at integer location (x_i, y_i) and tilted 45 degrees (either like '/' or like '\'). For example, a fence oriented like '/' at position (3,5) could be described as a line segment from (2.9,4.9) to (3.1,5.1). Each fence (and also the location of the barn) lies at a different position with integer coordinates in the range -1,000,000...1,000,000. No fence lies at (0,0) or (a,b).

FJ plans to sit at his house at position (0,0) and look directly to the right (in the +x direction). With his gaze bouncing off some of the reflective fences on his farm, he hopes to be able to see the point (a,b). Unfortunately, FJ thinks he oriented one of his fences incorrectly (e.g., '\' instead of '/'). Please output the index of the first fence in FJ's list such that by toggling its direction (between '/' and '\' or vice versa), FJ will be able to see the point (a,b).

If FJ can already see the point (a,b) without toggling any fence, please output 0. If it is still impossible for him to see (a,b) even after toggling up to a single fence, output -1.

INPUT FORMAT:

* Line 1: Three space-separated integers, N, a, and b.

* Lines 2..1+N: Line i+1 describes fence i and contains either "x_i y_i /" or "x_i y_i \", where (x_i, y_i) is the location of the center of the fence, and \ or / refers to its orientation.

SAMPLE INPUT:

```
5 6 2
3 0 /
0 2 /
1 2 /
3 2 \
1 3 \
```

INPUT DETAILS:

A map of the farm looks like this (with H denoting FJ's house and B
denoting the barn):

```
3 .\.....
2 //.\..B
1 .......
```

```
0 H../...
  0123456
```

OUTPUT FORMAT:

* Line 1: The index of the first fence for which toggling that fence allows FJ to see the point (a,b). If FJ can already see the point (a,b), please output 0, or if there is no way he can see (a,b) even after toggling up to one fence, please output -1.

SAMPLE OUTPUT:

4

OUTPUT DETAILS:

By toggling the fence at position (3,2), FJ can see the point (a,b). On the map:

```
3 .\.....
2 //./--B
1 ...|...
0 H--/...
  0123456
```