Cross Count 2

🔒 locked

You're back at the same problem, but this time the limits are high as sky.

There is a Cartesian plane with *N* vertical lines (**infinitely long**), and *M* line segments (**finite**).
Your task is to count the total number of crossings made by the finite line segments, with the infinitely long vertical lines.

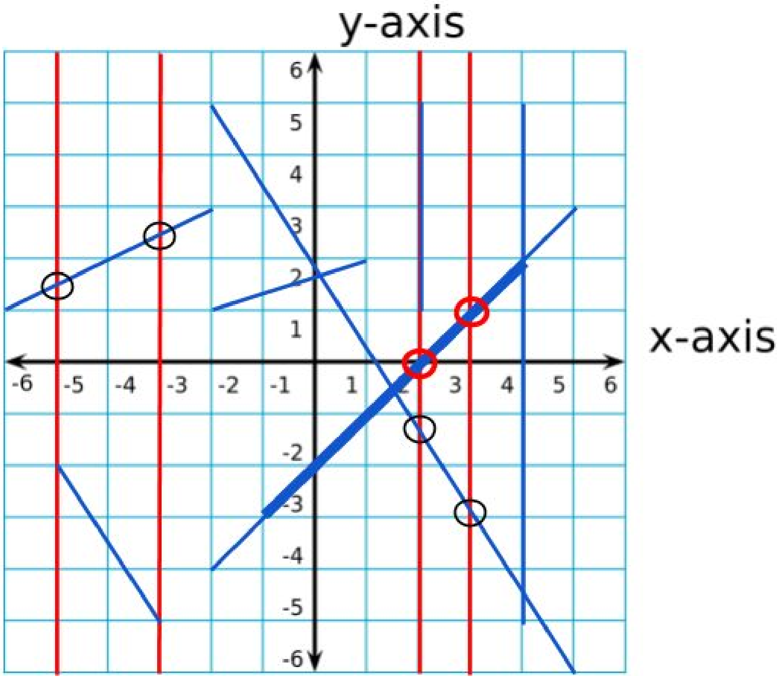
Example: There are *N = 4* infinitely long vertical lines, at

$$x = -5, -3, \quad 2, \quad 4$$

There are *M = 8* finite line segments:

$$\begin{array}{l}
(-2, \quad 5), (\quad 5, -6) \\
(-5, -2), (-3, -5) \\
(-2, \quad 3), (-6, \quad 1) \\
(-1, -3), (\quad 4, \quad 2) \\
(\quad 2, \quad 5), (\quad 2, \quad 1) \\
(\quad 4, \quad 5), (\quad 4, -5) \\
(-2, -4), (\quad 5, \quad 3) \\
(\quad 1, \quad 2), (-2, \quad 1)
\end{array}$$

After marking the infinitely long vertical lines and the line segments, the Cartesian plane looks like this.



The **circles** denote the crossings. **Black** circles denote **1** crossing. **Red** circles denote **2** crossings. So the answer is **8**.

Input Format

First line contains two integers, N and M.
Second line contains N space separated integers, with i^th^ of them indicating X_i, the x coordinate of the i^th^ infinitely long vertical line.
M lines follow, each containing 4 space separated integers, x_1, y_1, x_2, y_2. The start and end points of the line segments.

Constraints

- $1 \leq n, m \leq 10^6$
- $-10^{16} \leq X_i, x_1, y_1, x_2, y_2 \leq 10^{16}$

Limits

- **Time Limit:** 2s
- **Memory Limit:** 256MB

Output Format

A single **integer**, denoting the total number of crossings between the infinitely long vertical lines and the line segments.

Notes

- The output might not fit into Integer data type.
- For contestants using C++ or Java, you might need to use faster I/O techniques. (Refer: C++ / Java)

Sample Input 0

```
4 8
-5 -3 2 3
-2 5 5 -6
-5 -2 -3 -5
-2 3 -6 1
-1 -3 4 2
2 5 2 1
4 5 4 -5
-2 -4 5 3
1 2 -2 1
```

Sample Output 0

```
8
```

Explanation 0

Explained above.