Question 3 – Hard

**The Great Escape**

Your friend, a highly skilled operative, is trapped in enemy territory after a mission gone wrong. This territory, known as **Zone Omega**, is a massive rectangular piece of land used by the enemy for high-security operations. The enemy commander, infamous for his cunning strategies, has fortified Zone Omega with an elaborate defense system to ensure no one can escape alive.

The rectangular territory is surrounded by an **impenetrable, electrified fence**—an advanced security mechanism capable of instantly neutralizing intruders. However, the commander has intentionally left **two small unguarded gaps** at the **bottom-left corner** (entry point) and the **top-right corner** (exit point). These gaps are bait for desperate escape attempts, ensuring that anyone trying to flee will cross paths with the **Guardian Network**.

**The Guardian Network**

The **Guardian Network** consists of **N guard towers**, strategically placed both inside and outside Zone Omega. Each tower is manned by **elite marksmen**, equipped with long-range weaponry capable of covering circular guard areas around their respective towers.

* **Guard areas** are circular zones around the towers where any detected movement will be met with instant termination.
* The radius of each guard area (R) represents the **maximum range** of that tower’s defensive capability.
* Some towers located outside the rectangular boundary can still project their guard areas into Zone Omega, making escape even more perilous.

Your friend must navigate through Zone Omega from the **bottom-left corner (0,0)** to the **top-right corner (X,Y)** without stepping inside any of these guard zones or touching the **impenetrable, electrified fence**

**Challenge**

Your friend’s only hope lies in charting a safe path from the bottom-left to the top-right corner while avoiding all guard zones. Check whether he can safely go from bottom left corner to top right corner.

**Input Format**

You are given two positive integers X and Y, and a 2D array guard\_towers, where guard\_towers[i] = [xi, yi, ri] denotes a circle with center at (xi, yi) and radius ri. The bottom left corner of the rectangular land at the origin and top right corner at the coordinate X and Y.

All inputs will be integers.

Example input:

3 4

[[2,1,1]]

**Output Format**

Your task is to determine if your friend can escape safely.

* Output **"YES"** if there exists a safe path.
* Output **"NO"** if no such path exists.

**Additional Constraints**

* 3 <= X, Y <= 109
* 1 <= guard\_towers.length<= 1000
* guard\_towers [i].length == 3
* 1 <= xi, yi, ri <= 109

**Example 1:**

Input:

3 4

[[2,1,1]]

Output:

YES

**Example 2:**

Input:

3 3

[[1,1,2]]

Output:

NO

**Example 3:**

Input:

3 3

[[2,1,1],[1,2,1]]

Output:

NO

**Example 4:**

Input:

4 4

[[5,5,1]]

Output:

YES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Testcase 1:**

Input:

3 4

[[2,1,1]]

Output:

YES

**Testcase 2:**

Input:

3 3

[[1,1,2]]

Output:

NO

**Testcase 3:**

Input:

3 3

[[2,1,1],[1,2,1]]

Output:

NO

**Testcase 4:**

Input:

4 4

[[5,5,1]]

Output:

YES

**Testcase 5:**

Input:

10 10

[[15, 15, 3], [20, 20, 5]]

Output:

YES

**Testcase 6:**

Input:

10 10

[[5, 5, 3]]

Output:

YES

**Testcase 7:**

Input:

10 10

[[0, 0, 2], [10, 10, 2]]

Output:

NO

**Testcase 8:**

Input:

10 10

[[5, 5, 5]]

Output:

NO

**Testcase 9:**

Input:

20 15

[[30, 30, 5], [-10, -10, 5]]

Output:

YES

**Testcase 10:**

Input:

10 10

[[10, 0, 2]]

Output:

YES

**Testcase 11:**

Input:

10 10

[[3, 3, 1], [7, 7, 1], [3, 7, 1], [7, 3, 1]]

Output:

YES

**Testcase 12:**

Input:

8 8

[[2, 2, 2], [6, 2, 2], [2, 6, 2], [6, 6, 2]]

Output:

NO

**Testcase 13:**

Input:

10 10

[[5, 5, 5], [6, 6, 4]]

Output:

NO

**Testcase 14:**

Input:

50 50

[[10, 10, 5], [30, 30, 5], [40, 40, 6]]

Output:

YES

**Testcase 15:**

Input:

15 15

[[0, 0, 5], [15, 15, 5]]

Output:

NO

**Testcase 16:**

Input:

8 8

[[4, 0, 2], [4, 4, 2], [4, 7, 1]]

Output:

NO

**Testcase 17:**

Input:

8 8

[[0, 4, 3], [4, 4, 2], [8, 4, 3]]

Output:

NO