Problem 3288: Length of the Longest Increasing Path

Problem Information

Difficulty: Hard
Acceptance Rate: 18.09%
Paid Only: No
Tags: Array, Binary Search, Sorting

Problem Description

You are given a 2D array of integers `coordinates` of length `n` and an integer `k`, where `0 <= k < n`.

`coordinates[i] = [xi, yi]` indicates the point `(xi, yi)` in a 2D plane.

An **increasing path** of length `m` is defined as a list of points `(x1, y1)`, `(x2, y2)`, `(x3, y3)`, ..., `(xm, ym)` such that:

* `xi < xi + 1` and `yi < yi + 1` for all `i` where `1 <= i < m`. * `(xi, yi)` is in the given coordinates for all `i` where `1 <= i <= m`.

Return the **maximum** length of an **increasing path** that contains `coordinates[k]`.

Example 1:

Input: coordinates = [[3,1],[2,2],[4,1],[0,0],[5,3]], k = 1

Output: 3

Explanation:

`(0, 0)`, `(2, 2)`, `(5, 3)` is the longest increasing path that contains `(2, 2)`.

Example 2:

Input: coordinates = [[2,1],[7,0],[5,6]], k = 2

Output: 2

Explanation:

`(2, 1)`, `(5, 6)` is the longest increasing path that contains `(5, 6)`.

Constraints:

* `1 <= n == coordinates.length <= 105` * `coordinates[i].length == 2` * `0 <= coordinates[i][0], coordinates[i][1] <= 109` * All elements in `coordinates` are **distinct**. * `0 <= k <= n - 1`

Code Snippets

C++:

```
class Solution {
public:
int maxPathLength(vector<vector<int>>& coordinates, int k) {

}
};
```

Java:

```
class Solution {
public int maxPathLength(int[][] coordinates, int k) {

}
}
```

Python3:

```
class Solution:
def maxPathLength(self, coordinates: List[List[int]], k: int) -> int:
```