Problem 1057: Campus Bikes

Problem Information

Difficulty: Medium
Acceptance Rate: 59.07%
Paid Only: Yes
Tags: Array, Sorting, Heap (Priority Queue)

Problem Description

On a campus represented on the X-Y plane, there are `n` workers and `m` bikes, with `n <= m`.

You are given an array `workers` of length `n` where `workers[i] = [xi, yi]` is the position of the `ith` worker. You are also given an array `bikes` of length `m` where `bikes[j] = [xj, yj]` is the position of the `jth` bike. All the given positions are **unique**.

Assign a bike to each worker. Among the available bikes and workers, we choose the `(workeri, bikej)` pair with the shortest **Manhattan distance** between each other and assign the bike to that worker.

If there are multiple `(workeri, bikej)` pairs with the same shortest **Manhattan distance** , we choose the pair with **the smallest worker index**. If there are multiple ways to do that, we choose the pair with **the smallest bike index**. Repeat this process until there are no available workers.

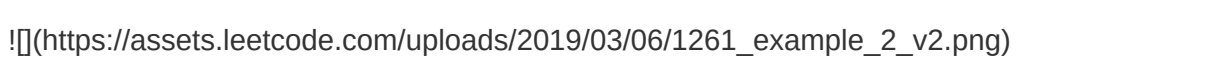
Return _an array_`answer` _of length_`n` _, where_`answer[i]`_is the index (**0-indexed**) of the bike that the _`ith` _worker is assigned to_.

The **Manhattan distance** between two points `p1` and `p2` is `Manhattan(p1, p2) = |p1.x - p2.x| + |p1.y - p2.y|`.

Example 1:

Input: workers = [[0,0],[2,1]], bikes = [[1,2],[3,3]] **Output:** [1,0] **Explanation:** Worker 1 grabs Bike 0 as they are closest (without ties), and Worker 0 is assigned Bike 1. So the output is [1, 0].

Example 2:



Input: workers = [[0,0],[1,1],[2,0]], bikes = [[1,0],[2,2],[2,1]] **Output:** [0,2,1] **Explanation:** Worker 0 grabs Bike 0 at first. Worker 1 and Worker 2 share the same distance to Bike 2, thus Worker 1 is assigned to Bike 2, and Worker 2 will take Bike 1. So the output is [0,2,1].

Constraints:

* `n == workers.length` * `m == bikes.length` * `1 <= n <= m <= 1000` * `workers[i].length == bikes[j].length == 2` * `0 <= xi, yi < 1000` * `0 <= xj, yj < 1000` * All worker and bike locations are **unique**.

Code Snippets

C++:

```
class Solution {
public:
vector<int> assignBikes(vector<vector<int>>& workers, vector<vector<int>>&
bikes) {

}
};
```

Java:

```
class Solution {
public int[] assignBikes(int[][] workers, int[][] bikes) {

}
}
```

Python3:

```
class Solution:
def assignBikes(self, workers: List[List[int]], bikes: List[List[int]]) ->
List[int]:
```