Problem 2736: Maximum Sum Queries

Problem Information

Difficulty: Hard
Acceptance Rate: 29.45%
Paid Only: No
Tags: Array, Binary Search, Stack, Binary Indexed Tree, Segment Tree, Sorting, Monotonic Stack

Problem Description

You are given two **0-indexed** integer arrays `nums1` and `nums2`, each of length `n`, and a **1-indexed 2D array** `queries` where `queries[i] = [xi, yi]`.

For the `ith` query, find the **maximum value** of `nums1[j] + nums2[j]` among all indices `j` `(0 <= j < n)`, where `nums1[j] >= xi` and `nums2[j] >= yi`, or **-1** if there is no `j` satisfying the constraints.

Return _an array_`answer` _where_`answer[i]`_is the answer to the_`ith` _query._

Example 1:

Input: nums1 = [4,3,1,2], nums2 = [2,4,9,5], queries = [[4,1],[1,3],[2,5]] **Output:** [6,10,7] **Explanation:** For the 1st query xi = 4 and yi = 1, we can select index j = 0 since nums1[j] >= 4 and nums2[j] >= 1. The sum nums1[j] + nums2[j] is 6, and we can show that 6 is the maximum we can obtain. For the 2nd query xi = 1 and yi = 3, we can select index j = 2 since nums1[j] >= 1 and nums2[j] >= 3. The sum nums1[j] + nums2[j] is 10, and we can show that 10 is the maximum we can obtain. For the 3rd query xi = 2 and yi = 5, we can select index j = 3 since nums1[j] >= 2 and nums2[j] >= 5. The sum nums1[j] + nums2[j] is 7, and we can show that 7 is the maximum we can obtain. Therefore, we return [6,10,7].

Example 2:

Input: nums1 = [3,2,5], nums2 = [2,3,4], queries = [[4,4],[3,2],[1,1]] **Output:** [9,9,9] **Explanation:** For this example, we can use index j = 2 for all the queries since it satisfies the constraints for each query.

Example 3:

Input: nums1 = [2,1], nums2 = [2,3], queries = [[3,3]] **Output:** [-1] **Explanation:** There is one query in this example with xi = 3 and yi = 3. For every index, j, either nums1[j] < xi or nums2[j] < yi. Hence, there is no solution.

Constraints:

* `nums1.length == nums2.length` * `n == nums1.length ` * `1 <= n <= 105` * `1 <= nums1[i], nums2[i] <= 109 ` * `1 <= queries.length <= 105` * `queries[i].length == 2` * `xi == queries[i][1]` * `yi == queries[i][2]` * `1 <= xi, yi <= 109`

Code Snippets

C++:

```
class Solution {
public:
vector<int> maximumSumQueries(vector<int>& nums1, vector<int>& nums2,
vector<vector<int>>& queries) {

}
};
```

Java:

```
class Solution {
public int[] maximumSumQueries(int[] nums1, int[] nums2, int[][] queries) {

}
}
```

Python3:

```
class Solution:
def maximumSumQueries(self, nums1: List[int], nums2: List[int], queries:
List[List[int]]) -> List[int]:
```