Problem 3633: Earliest Finish Time for Land and Water Rides I

Problem Information

Difficulty: Easy
Acceptance Rate: 61.32%
Paid Only: No
Tags: Array, Two Pointers, Binary Search, Greedy, Sorting

Problem Description

You are given two categories of theme park attractions: **land rides** and **water rides**.

* **Land rides** * `landStartTime[i]` - the earliest time the `ith` land ride can be boarded. * `landDuration[i]` - how long the `ith` land ride lasts. * **Water rides** * `waterStartTime[j]` - the earliest time the `jth` water ride can be boarded. * `waterDuration[j]` - how long the `jth` water ride lasts.

A tourist must experience **exactly one** ride from **each** category, in **either order**.

* A ride may be started at its opening time or **any later moment**. * If a ride is started at time `t`, it finishes at time `t + duration`. * Immediately after finishing one ride the tourist may board the other (if it is already open) or wait until it opens.

Return the **earliest possible time** at which the tourist can finish both rides.

Example 1:

Input: landStartTime = [2,8], landDuration = [4,1], waterStartTime = [6], waterDuration = [3]

Output: 9

Explanation:

* Plan A (land ride 0 -> water ride 0): * Start land ride 0 at time `landStartTime[0] = 2`. Finish at `2 + landDuration[0] = 6`. * Water ride 0 opens at time `waterStartTime[0] = 6`. Start immediately at `6`, finish at `6 + waterDuration[0] = 9`. * Plan B (water ride 0 -> land ride 1): * Start water ride 0 at time `waterStartTime[0] = 6`. Finish at `6 + waterDuration[0] = 9`. * Land ride 1 opens at `landStartTime[1] = 8`. Start at time `9`, finish at `9 + landDuration[1] = 10`. * Plan C (land ride 1 -> water ride 0): * Start land ride 1 at time `landStartTime[1] = 8`. Finish at `8 + landDuration[1] = 9`. * Water ride 0 opened at `waterStartTime[0] = 6`. Start at time `9`, finish at `9 + waterDuration[0] = 12`. * Plan D (water ride 0 -> land ride 0): * Start water ride 0 at time `waterStartTime[0] = 6`. Finish at `6 + waterDuration[0] = 9`. * Land ride 0 opened at `landStartTime[0] = 2`. Start at time `9`, finish at `9 + landDuration[0] = 13`.

Plan A gives the earliest finish time of 9.

Example 2:

Input: landStartTime = [5], landDuration = [3], waterStartTime = [1], waterDuration = [10]

Output: 14

Explanation:

* Plan A (water ride 0 -> land ride 0): * Start water ride 0 at time `waterStartTime[0] = 1`. Finish at `1 + waterDuration[0] = 11`. * Land ride 0 opened at `landStartTime[0] = 5`. Start immediately at `11` and finish at `11 + landDuration[0] = 14`. * Plan B (land ride 0 -> water ride 0): * Start land ride 0 at time `landStartTime[0] = 5`. Finish at `5 + landDuration[0] = 8`. * Water ride 0 opened at `waterStartTime[0] = 1`. Start immediately at `8` and finish at `8 + waterDuration[0] = 18`.

Plan A provides the earliest finish time of 14.**■■■■■■■**

Constraints:

* `1 <= n, m <= 100` * `landStartTime.length == landDuration.length == n` * `waterStartTime.length == waterDuration.length == m` * `1 <= landStartTime[i], landDuration[i], waterStartTime[j], waterDuration[j] <= 1000`

Code Snippets

C++:

```
class Solution {
public:
int earliestFinishTime(vector<int>& landStartTime, vector<int>& landDuration,
vector<int>& waterStartTime, vector<int>& waterDuration) {

}
};
```

Java:

```
class Solution {
public int earliestFinishTime(int[] landStartTime, int[] landDuration, int[]
waterStartTime, int[] waterDuration) {

}
}
```

Python3:

```
class Solution:
def earliestFinishTime(self, landStartTime: List[int], landDuration:
List[int], waterStartTime: List[int], waterDuration: List[int]) -> int:
```