Warehouse Network Optimization – Coding Problem

In a large-scale warehouse distribution network, minimizing operational overhead and optimizing package routing is essential to ensure efficient delivery across various regions.

The network consists of n warehouses, numbered from 1 to n, each strategically positioned at its corresponding index. Each warehouse has a specific storage capacity, represented by the array warehouseCapacity, where warehouseCapacity[i] indicates the capacity of the warehouse located at position i (using 1-based indexing).

These warehouses are arranged in non-decreasing order of their storage capacities, meaning:
warehouseCapacity[i] ≤ warehouseCapacity[i+1] for all 0 ≤ i < n-1

Each warehouse must establish a connection to a distribution hub positioned at a location greater than or equal to its own index. Thus, a warehouse at position i can only connect to a hub at position j where j ≥ i.

To ensure connectivity, a primary central hub is placed at the last warehouse, i.e., position n. This hub acts as the default connection point for all warehouses if no other hubs are available.

To improve routing efficiency, multiple routing plans (queries) are considered. Each query introduces two high-capacity hubs at given positions hubA and hubB. For each routing plan, your task is to compute the minimum total connection cost for all warehouses, by always choosing the nearest available hub at or beyond each warehouse's position.

The cost to connect from warehouse i to hub j is defined as:
warehouseCapacity[j] - warehouseCapacity[i]

Function Signature

def getMinConnectionCost(warehouseCapacity: List[int], additionalHubs: List[List[int]]) -> List[int]:

Input

- warehouseCapacity: List of n integers, the storage capacities of warehouses in non-decreasing order.
- additionalHubs: A list of q queries, where each query contains two integers [hubA, hubB], denoting the positions (1-based) of additional hubs.

Output

- A list of q integers, where each value is the total minimum connection cost for that query.

Constraints

- $3 \le n \le 2.5 * 10^5$
- $1 \le q \le 2.5 * 10^5$
- $0 \le warehouseCapacity[i] \le 10^9$
- $warehouseCapacity[i] \le warehouseCapacity[i+1]$
- $1 \le additionalHubs[i][0] < additionalHubs[i][1] < n$

Example

Input:
warehouseCapacity = [0, 2, 5, 9, 12, 18]
additionalHubs = [[2, 5], [1, 3]]

Output:
[12, 18]

Explanation:

Query 1:
- Additional hubs at positions 2 and 5 (0-based → 1 and 4)
- Total cost = 2 + 0 + 7 + 3 + 0 + 0 = 12

Query 2:
- Additional hubs at positions 1 and 3 (0-based → 0 and 2)
- Total cost = 0 + 3 + 0 + 9 + 6 + 0 = 18