Problem 3538: Merge Operations for Minimum Travel Time

Problem Information

Difficulty: Hard
Acceptance Rate: 29.87%
Paid Only: No
Tags: Array, Dynamic Programming, Prefix Sum

Problem Description

You are given a straight road of length `l` km, an integer `n`, an integer `k`**,** and **two** integer arrays, `position` and `time`, each of length `n`.

The array `position` lists the positions (in km) of signs in **strictly** increasing order (with `position[0] = 0` and `position[n - 1] = l`).

Each `time[i]` represents the time (in minutes) required to travel 1 km between `position[i]` and `position[i + 1]`.

You **must** perform **exactly** `k` merge operations. In one merge, you can choose any **two** adjacent signs at indices `i` and `i + 1` (with `i > 0` and `i + 1 < n`) and:

* Update the sign at index `i + 1` so that its time becomes `time[i] + time[i + 1]`. * Remove the sign at index `i`.

Return the **minimum** **total** **travel time** (in minutes) to travel from 0 to `l` after **exactly** `k` merges.

Example 1:

Input: l = 10, n = 4, k = 1, position = [0,3,8,10], time = [5,8,3,6]

Output: 62

Explanation:

* Merge the signs at indices 1 and 2. Remove the sign at index 1, and change the time at index 2 to `8 + 3 = 11`.

* After the merge: * `position` array: `[0, 8, 10]` * `time` array: `[5, 11, 6]` * * Segment | Distance (km) | Time per km (min) | Segment Travel Time (min) ---|---|---|--- 0 -> 8 | 8 | 5 | 8 × 5 = 40 8 -> 10 | 2 | 11 | 2 × 11 = 22 * Total Travel Time: `40 + 22 = 62`, which is the minimum possible time after exactly 1 merge.

Example 2:

Input: l = 5, n = 5, k = 1, position = [0,1,2,3,5], time = [8,3,9,3,3]

Output: 34

Explanation:

* Merge the signs at indices 1 and 2. Remove the sign at index 1, and change the time at index 2 to `3 + 9 = 12`. * After the merge: * `position` array: `[0, 2, 3, 5]` * `time` array: `[8, 12, 3, 3]` * * Segment | Distance (km) | Time per km (min) | Segment Travel Time (min) ---|---|---|--- 0 -> 2 | 2 | 8 | 2 × 8 = 16 2 -> 3 | 1 | 12 | 1 × 12 = 12 3 -> 5 | 2 | 3 | 2 × 3 = 6 * Total Travel Time: `16 + 12 + 6 = 34`**,** which is the minimum possible time after exactly 1 merge.

Constraints:

* `1 <= l <= 105` * `2 <= n <= min(l + 1, 50)` * `0 <= k <= min(n - 2, 10)` * `position.length == n` * `position[0] = 0` and `position[n - 1] = l` * `position` is sorted in strictly increasing order. * `time.length == n` * `1 <= time[i] <= 100■` * `1 <= sum(time) <= 100`

Code Snippets

C++:

```
class Solution {
public:
int minTravelTime(int l, int n, int k, vector<int>& position, vector<int>&
time) {
```

```
}
};
```

Java:

```
class Solution {
public int minTravelTime(int l, int n, int k, int[] position, int[] time) {

}
}
```

Python3:

```
class Solution:
def minTravelTime(self, l: int, n: int, k: int, position: List[int], time:
List[int]) -> int:
```