Problem 3196: Maximize Total Cost of Alternating Subarrays

Problem Information

Difficulty: Medium
Acceptance Rate: 29.27%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You are given an integer array `nums` with length `n`.

The **cost** of a subarray `nums[l..r]`, where `0 <= l <= r < n`, is defined as:

`cost(l, r) = nums[l] - nums[l + 1] + ... + nums[r] * (–1)r – l`

Your task is to **split** `nums` into subarrays such that the **total** **cost** of the subarrays is **maximized** , ensuring each element belongs to **exactly one** subarray.

Formally, if `nums` is split into `k` subarrays, where `k > 1`, at indices `i1, i2, ..., ik – 1`, where `0 <= i1 < i2 < ... < ik - 1 < n - 1`, then the total cost will be:

`cost(0, i1) + cost(i1 + 1, i2) + ... + cost(ik – 1 + 1, n – 1)`

Return an integer denoting the _maximum total cost_ of the subarrays after splitting the array optimally.

Note: If `nums` is not split into subarrays, i.e. `k = 1`, the total cost is simply `cost(0, n - 1)`.

Example 1:

Input: nums = [1,-2,3,4]

Output: 10

Explanation:

One way to maximize the total cost is by splitting `[1, -2, 3, 4]` into subarrays `[1, -2, 3]` and `[4]`. The total cost will be `(1 + 2 + 3) + 4 = 10`.

Example 2:

Input: nums = [1,-1,1,-1]

Output: 4

Explanation:

One way to maximize the total cost is by splitting `[1, -1, 1, -1]` into subarrays `[1, -1]` and `[1, -1]`. The total cost will be `(1 + 1) + (1 + 1) = 4`.

Example 3:

Input: nums = [0]

Output: 0

Explanation:

We cannot split the array further, so the answer is 0.

Example 4:

Input: nums = [1,-1]

Output: 2

Explanation:

Selecting the whole array gives a total cost of `1 + 1 = 2`, which is the maximum.

Constraints:

* `1 <= nums.length <= 105` * `-109 <= nums[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
long long maximumTotalCost(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public long maximumTotalCost(int[] nums) {

}
}
```

Python3:

```
class Solution:
def maximumTotalCost(self, nums: List[int]) -> int:
```