Problem 3205: Maximum Array Hopping Score I

Problem Information

Difficulty: Medium
Acceptance Rate: 77.33%
Paid Only: Yes
Tags: Array, Dynamic Programming, Stack, Greedy, Monotonic Stack

Problem Description

Given an array `nums`, you have to get the **maximum** score starting from index 0 and **hopping** until you reach the last element of the array.

In each **hop** , you can jump from index `i` to an index `j > i`, and you get a **score** of `(j - i) * nums[j]`.

Return the _maximum score_ you can get.

Example 1:

Input: nums = [1,5,8]

Output: 16

Explanation:

There are two possible ways to reach the last element:

* `0 -> 1 -> 2` with a score of `(1 - 0) * 5 + (2 - 1) * 8 = 13`. * `0 -> 2` with a score of `(2 - 0) * 8 = 16`.

Example 2:

Input: nums = [4,5,2,8,9,1,3]

Output: 42

Explanation:

We can do the hopping `0 -> 4 -> 6` with a score of `(4 - 0) * 9 + (6 - 4) * 3 = 42`.

Constraints:

* `2 <= nums.length <= 103` * `1 <= nums[i] <= 105`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public int maxScore(int[] nums) {

}
}
```

Python3:

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