Problem 396: Rotate Function

Problem Information

Difficulty: Medium
Acceptance Rate: 44.97%
Paid Only: No
Tags: Array, Math, Dynamic Programming

Problem Description

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

Assume `arrk` to be an array obtained by rotating `nums` by `k` positions clock-wise. We define the **rotation function** `F` on `nums` as follow:

* `F(k) = 0 * arrk[0] + 1 * arrk[1] + ... + (n - 1) * arrk[n - 1].`

Return _the maximum value of_ `F(0), F(1), ..., F(n-1)`.

The test cases are generated so that the answer fits in a **32-bit** integer.

Example 1:

Input: nums = [4,3,2,6] **Output:** 26 **Explanation:** F(0) = (0 * 4) + (1 * 3) + (2 * 2) + (3 * 6) = 0 + 3 + 4 + 18 = 25 F(1) = (0 * 6) + (1 * 4) + (2 * 3) + (3 * 2) = 0 + 4 + 6 + 6 = 16 F(2) = (0 * 2) + (1 * 6) + (2 * 4) + (3 * 3) = 0 + 6 + 8 + 9 = 23 F(3) = (0 * 3) + (1 * 2) + (2 * 6) + (3 * 4) = 0 + 2 + 12 + 12 = 26 So the maximum value of F(0), F(1), F(2), F(3) is F(3) = 26.

Example 2:

Input: nums = [100] **Output:** 0

Constraints:

* `n == nums.length` * `1 <= n <= 105` * `-100 <= nums[i] <= 100`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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