

Problem 3743: Maximize Cyclic Partition Score

Problem Information

Difficulty: Hard

Acceptance Rate: 9.47%

Paid Only: No

Tags: Array, Dynamic Programming

Problem Description

You are given a **cyclic** array `nums`` and an integer `k``.

Partition `nums`` into **at most** `k`` subarrays. As `nums`` is cyclic, these subarrays may wrap around from the end of the array back to the beginning.

The **range** of a subarray is the difference between its **maximum** and **minimum** values. The **score** of a partition is the sum of subarray **ranges**.

Return the **maximum** possible **score** among all cyclic partitions.

Example 1:

Input: `nums = [1,2,3,3]`, `k = 2`

Output: 3

Explanation:

* Partition `nums`` into `[2, 3]` and `[3, 1]` (wrapped around). * The range of `[2, 3]` is `max(2, 3) - min(2, 3) = 3 - 2 = 1`. * The range of `[3, 1]` is `max(3, 1) - min(3, 1) = 3 - 1 = 2`. * The score is `1 + 2 = 3`.

Example 2:

Input: `nums = [1,2,3,3]`, `k = 1`

****Output:**** 2

****Explanation:****

* Partition `nums` into `[1, 2, 3, 3]`. * The range of `[1, 2, 3, 3]` is $\max(1, 2, 3, 3) - \min(1, 2, 3, 3) = 3 - 1 = 2$. * The score is 2.

****Example 3:****

****Input:**** nums = [1,2,3,3], k = 4

****Output:**** 3

****Explanation:****

Identical to Example 1, we partition `nums` into `[2, 3]` and `[3, 1]`. Note that `nums` may be partitioned into fewer than `k` subarrays.

****Constraints:****

* $1 \leq \text{nums.length} \leq 1000$ * $1 \leq \text{nums}[i] \leq 109$ * $1 \leq k \leq \text{nums.length}$

Code Snippets

C++:

```
class Solution {
public:
    long long maximumScore(vector<int>& nums, int k) {

    }
};
```

Java:

```
class Solution {
    public long maximumScore(int[] nums, int k) {

    }
}
```

```
}
```

Python3:

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