Problem 813: Largest Sum of Averages

Problem Information

Difficulty: Medium
Acceptance Rate: 54.57%
Paid Only: No
Tags: Array, Dynamic Programming, Prefix Sum

Problem Description

You are given an integer array `nums` and an integer `k`. You can partition the array into **at most** `k` non-empty adjacent subarrays. The **score** of a partition is the sum of the averages of each subarray.

Note that the partition must use every integer in `nums`, and that the score is not necessarily an integer.

Return _the maximum**score** you can achieve of all the possible partitions_. Answers within `10-6` of the actual answer will be accepted.

Example 1:

Input: nums = [9,1,2,3,9], k = 3 **Output:** 20.00000 **Explanation:** The best choice is to partition nums into [9], [1, 2, 3], [9]. The answer is 9 + (1 + 2 + 3) / 3 + 9 = 20. We could have also partitioned nums into [9, 1], [2], [3, 9], for example. That partition would lead to a score of 5 + 2 + 6 = 13, which is worse.

Example 2:

Input: nums = [1,2,3,4,5,6,7], k = 4 **Output:** 20.50000

Constraints:

* `1 <= nums.length <= 100` * `1 <= nums[i] <= 104` * `1 <= k <= nums.length`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public double largestSumOfAverages(int[] nums, int k) {

}
}
```

Python3:

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