Problem 3346: Maximum Frequency of an Element After Performing Operations I

Problem Information

Difficulty: Medium
Acceptance Rate: 40.00%
Paid Only: No
Tags: Array, Binary Search, Sliding Window, Sorting, Prefix Sum

Problem Description

You are given an integer array `nums` and two integers `k` and `numOperations`.

You must perform an **operation** `numOperations` times on `nums`, where in each operation you:

* Select an index `i` that was **not** selected in any previous operations. * Add an integer in the range `[-k, k]` to `nums[i]`.

Return the **maximum** possible frequency of any element in `nums` after performing the **operations**.

Example 1:

Input: nums = [1,4,5], k = 1, numOperations = 2

Output: 2

Explanation:

We can achieve a maximum frequency of two by:

* Adding 0 to `nums[1]`. `nums` becomes `[1, 4, 5]`. * Adding -1 to `nums[2]`. `nums` becomes `[1, 4, 4]`.

Example 2:

Input: nums = [5,11,20,20], k = 5, numOperations = 1

Output: 2

Explanation:

We can achieve a maximum frequency of two by:

* Adding 0 to `nums[1]`.

Constraints:

* `1 <= nums.length <= 105` * `1 <= nums[i] <= 105` * `0 <= k <= 105` * `0 <= numOperations <= nums.length`

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public int maxFrequency(int[] nums, int k, int numOperations) {

}
}
```

Python3:

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