Problem 1838: Frequency of the Most Frequent Element

Problem Information

Difficulty: Medium
Acceptance Rate: 44.47%
Paid Only: No
Tags: Array, Binary Search, Greedy, Sliding Window, Sorting, Prefix Sum

Problem Description

The **frequency** of an element is the number of times it occurs in an array.

You are given an integer array `nums` and an integer `k`. In one operation, you can choose an index of `nums` and increment the element at that index by `1`.

Return _the**maximum possible frequency** of an element after performing **at most** _`k` _operations_.

Example 1:

Input: nums = [1,2,4], k = 5 **Output:** 3**Explanation:** Increment the first element three times and the second element two times to make nums = [4,4,4]. 4 has a frequency of 3.

Example 2:

Input: nums = [1,4,8,13], k = 5 **Output:** 2 **Explanation:** There are multiple optimal solutions: - Increment the first element three times to make nums = [4,4,8,13]. 4 has a frequency of 2. - Increment the second element four times to make nums = [1,8,8,13]. 8 has a frequency of 2. - Increment the third element five times to make nums = [1,4,13,13]. 13 has a frequency of 2.

Example 3:

Input: nums = [3,9,6], k = 2 **Output:** 1

Constraints:

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

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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