Problem 3092: Most Frequent IDs

Problem Information

Difficulty: Medium
Acceptance Rate: 42.13%
Paid Only: No
Tags: Array, Hash Table, Heap (Priority Queue), Ordered Set

Problem Description

The problem involves tracking the frequency of IDs in a collection that changes over time. You have two integer arrays, `nums` and `freq`, of equal length `n`. Each element in `nums` represents an ID, and the corresponding element in `freq` indicates how many times that ID should be added to or removed from the collection at each step.

* **Addition of IDs:** If `freq[i]` is positive, it means `freq[i]` IDs with the value `nums[i]` are added to the collection at step `i`. * **Removal of IDs:** If `freq[i]` is negative, it means `-freq[i]` IDs with the value `nums[i]` are removed from the collection at step `i`.

Return an array `ans` of length `n`, where `ans[i]` represents the **count** of the _most frequent ID_ in the collection after the `ith` step. If the collection is empty at any step, `ans[i]` should be 0 for that step.

Example 1:

Input: nums = [2,3,2,1], freq = [3,2,-3,1]

Output: [3,3,2,2]

Explanation:

After step 0, we have 3 IDs with the value of 2. So `ans[0] = 3`. After step 1, we have 3 IDs with the value of 2 and 2 IDs with the value of 3. So `ans[1] = 3`. After step 2, we have 2 IDs with the value of 3. So `ans[2] = 2`. After step 3, we have 2 IDs with the value of 3 and 1 ID with the value of 1. So `ans[3] = 2`.

Example 2:

Input: nums = [5,5,3], freq = [2,-2,1]

Output: [2,0,1]

Explanation:

After step 0, we have 2 IDs with the value of 5. So `ans[0] = 2`. After step 1, there are no IDs. So `ans[1] = 0`. After step 2, we have 1 ID with the value of 3. So `ans[2] = 1`.

Constraints:

* `1 <= nums.length == freq.length <= 105` * `1 <= nums[i] <= 105` * `-105 <= freq[i] <= 105` * `freq[i] != 0` * The input is generated such that the occurrences of an ID will not be negative in any step.

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public long[] mostFrequentIDs(int[] nums, int[] freq) {

}
}
```

Python3:

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