



Experiment-4

Student Name: Mohammad Saiful Haque

UID: 22BCS15656

Branch: BE-CSE

Section/Group: KPIT-901/B

Semester: 6th

Date of Performance: 18/01/25

Subject Name: Advanced Programming Lab - 2

Subject Code: 22CSP-351

1. Aim: Sorting and Searching.

1. Problem: 88. Merge Sorted Array.
2. Problem: 347. Top K Frequent Elements.

2. Objective:

1. Merge two sorted arrays nums1 and nums2 in-place without using extra space.
2. Find the k most frequent elements in an array using an optimized approach.

3. Implementation/Code:

1.)

```
class Solution {
```

```
public:
```

```
void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
```

```
    int i = m - 1, j = n - 1, k = m + n - 1;
```

```
    while (i >= 0 && j >= 0) {
```

```
        if (nums1[i] > nums2[j]) {
```

```
            nums1[k--] = nums1[i--];
```

```
        } else {
```

```
            nums1[k--] = nums2[j--];
```

```
        }
```

```
    }
```

```
    while (j >= 0) {
```

```
        nums1[k--] = nums2[j--];
```

```
    }
```

```
}  
};
```

2.)

```
class Solution {  
public:  
    vector<int> topKFrequent(vector<int>& nums, int k) {  
        unordered_map<int, int> freq;  
        for (int num : nums) {  
            freq[num]++;  
        }  
  
        vector<vector<int>> buckets(nums.size() + 1);  
        for (auto& [num, count] : freq) {  
            buckets[count].push_back(num);  
        }  
  
        vector<int> result;  
        for (int i = nums.size(); i >= 0 && result.size() < k; --i) {  
            for (int num : buckets[i]) {  
                result.push_back(num);  
                if (result.size() == k) return result;  
            }  
        }  
        return result;  
    }  
};
```

4. Output:

1.

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums1 =
[1,2,3,0,0,0]

m =
3

nums2 =
[2,5,6]

n =
3

Output

[1,2,2,3,5,6]

Expected

[1,2,2,3,5,6]

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 59 / 59 testcases passed

Mohammad Saiful Haque submitted at Mar 05, 2025 16:12

Editorial Solution

Runtime 0 ms | Beats 100.00% Memory 12.36 MB | Beats 39.43%

Analyze Complexity

100% 50% 0% 0% 1ms 2ms 3ms 4ms

Code | C++

```
class Solution {
public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
        int i = m - 1, j = n - 1, k = m + n - 1;
        while (i >= 0 && j >= 0) {
            if (nums1[i] > nums2[j]) {
                nums1[k--] = nums1[i--];
            } else {

```

View more

2.

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums =
[1,1,1,2,2,3]

k =
2

Output

[1,2]

Expected

[1,2]

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 21 / 21 testcases passed

Mohammad Saiful Haque submitted at Mar 05, 2025 16:13

Editorial Solution

Runtime 8 ms | Beats 10.67% Memory 19.77 MB | Beats 15.48%

Analyze Complexity

40% 20% 0% 0% 2ms 4ms 6ms 8ms 10ms

5. Time Complexity:

1. $O(m+n)$
2. $O(n)$

6. Space Complexity:

1. $O(1)$
2. $O(n)$

7. Learning Outcome:

1. Efficient merging using the two-pointer technique.
2. Modifying arrays in-place to optimize space.
3. Working with sorted arrays efficiently.
4. Hash maps for frequency counting.
5. Bucket sort for frequency-based grouping.
6. Optimized $O(n)$ approach instead of sorting ($O(n \log n)$).