



## Experiment 5

**Student Name:** Harsh Mishra

**Branch:** CSE

**Semester:** 6

**Subject Name:** AP LAB-II

**UID:** 22BCS14844

**Section/Group:** 604-B

**Date of Performance:**

**Subject Code:** 22CSP-351

### 1. Aim:

- Top K Frequent Elements
- Sort Colors
- Merge Sorted Array

### 2. Objective:

- Understand the fundamentals Searching and Sorting.
- Writing some good code.
- Understanding some Java concepts.

### 3. Implementation/Code:

```
import java.util.*;

class Main

{

    public int[] topKFrequent(int[] nums, int k)

    {

        final int n = nums.length;

        List<Integer> ans = new ArrayList<>();

        List<Integer>[] bucket = new List[n + 1];

        Map<Integer, Integer> count = new HashMap<>();

        for (final int num : nums)

            count.merge(num, 1, Integer::sum);
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
for (final int num : count.keySet())  
  
    {  
  
        final int freq = count.get(num);  
  
        if (bucket[freq] == null)  
  
            bucket[freq] = new ArrayList<>();  
  
        bucket[freq].add(num);  
  
    }  
  
for (int freq = n; freq > 0; --freq)  
  
    {  
  
        if (bucket[freq] != null)  
  
            ans.addAll(bucket[freq]);  
  
        if (ans.size() == k)  
  
            return ans.stream().mapToInt(Integer::intValue).toArray();  
  
    }  
  
    throw new IllegalArgumentException();  
  
    }  
  
    }
```

class Solution

```
{  
  
    public void sortColors(int[] nums)  
  
    {  
  
        int l = 0;
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int r = nums.length - 1;

for (int i = 0; i <= r;)

    if (nums[i] == 0)

        swap(nums, i++, l++);

    else if (nums[i] == 1)

        ++i;

    else

        swap(nums, i, r--);
}

private void swap(int[] nums, int i, int j)
{
    final int temp = nums[i];
    nums[i] = nums[j];
    nums[j] = temp;
}

}

class Solution
{
    public void merge(int[] nums1, int m, int[] nums2, int n) {

        int i = m - 1;

        int j = n - 1;
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int k = m + n - 1;

while (j >= 0)

    if (i >= 0 && nums1[i] > nums2[j])

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

    else

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

}
```

## 4. Output:

The screenshot shows a test result interface with a dark theme. At the top, there are tabs for 'Testcase' and 'Test Result', with 'Test Result' being the active tab. Below the tabs, the status 'Accepted' is displayed in green, followed by 'Runtime: 0 ms'. There are three tabs for 'Case 1', 'Case 2', and 'Case 3', with 'Case 1' being the active tab. Under the 'Input' section, there are four input fields: 'nums1 =' with the value '[1,2,3,0,0,0]', 'm =' with the value '3', 'nums2 =' with the value '[2,5,6]', and 'n =' with the value '3'. Under the 'Output' section, there is one output field showing the value '[1,2,2,3,5,6]'.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

☒ Testcase | [Test Result](#)

**Accepted** Runtime: 0 ms

• Case 1

• Case 2

Input

nums =  
[2,0,2,1,1,0]

Output

[0,0,1,1,2,2]

Expected

[0,0,1,1,2,2]

[♥ Contribute a testcase](#)

☒ Testcase | [Test Result](#)

**Accepted** Runtime: 1 ms

• Case 1

• Case 2

Input

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

k =  
2

Output

[1,2]

Expected

[1,2]

[♥ Contribute a testcase](#)



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 5. Learning Outcome:

- Learn how to use divide and conquer approach.
- Login to leetcode solution page for submitting solution.