B. M. S. COLLEGEOF ENGINEERING

(Autonomous college under VTU)

Bull Temple Rd, Basavanagudi, Bengaluru, Karnataka – 560019

Department of Computer Applications

The reportis submittedfor the fulfillmentof the **AAT** work for the course

"Python Programming"
(MMC104)

By

Karthikeya R Hegde: - 1BM25MCA060-T

Navaneeth Gowda G: - 1BM25MCA043-T

Under theGuidanceof

Dr. S. Uma
(Associate Professor, HOD)

Abstract:

This report presents the implementation and analysis of selected algorithmic problems solved using the LeetCode platform as part of the Application Activity Task (AAT) for the Python Programming course. The objective of this work is to enhance problem-solving ability, logical reasoning, and coding proficiency by solving complex programming challenges. Four hard-level problems—***Median of Two Sorted Arrays***, ***Reverse Nodes in K-Group***, ***First Missing Positive***, **and** ***Smallest Range Covering Elements from K Lists***—were implemented using Python. Each problem was analyzed carefully, and efficient algorithms were designed to meet the required time and space constraints. The solutions were tested and verified using LeetCode's automated test cases to ensure correctness. This activity helped in strengthening algorithmic thinking, understanding data structures, and gaining practical experience in writing optimized code using Python. The work also encouraged collaborative learning and systematic problem analysis.

Contents:

1.Introduction

This report presents the problem-solving activities carried out using the **LeetCode platform**. The objective of this work is to improve logical thinking, coding skills, and algorithmic understanding by solving programming problems using an online competitive coding platform.

The problems were solved using **Google Colab** and verified using LeetCode's automated test cases.

2.Objectives

- To improve problem-solving skills
- To understand algorithmic thinking
- To practice coding using LeetCode
- To implement efficient solutions
- To work collaboratively as a team

3.Platform and Tools Used

Tool	**Purpose**
LeetCode	Problem solving and test case validation
Google Colab	Code development and testing
Programming Language	Python

4.Problem Details

▪ Problem 1

Problem Name: Median Of Two Sorted Arrays (From Leetcode Problem Number 4)

Difficulty Level: Hard **Problem Description:**

This problem asks you to find the **median** of two **sorted arrays** nums1 and nums2 of sizes m and n, **without actually merging them**, and to do it in **O(log(m + n))** time.

Approach Used:

- Logical analysis of the problem: -
  The problem requires finding the median of two sorted arrays. Instead of handling them separately, both arrays are combined into a single list so that the overall ordering can be analyzed easily.

- Algorithm implementation:-

  The two input arrays are merged into one array and then sorted. The length of the merged array is calculated to determine whether the total number of elements is even or odd.

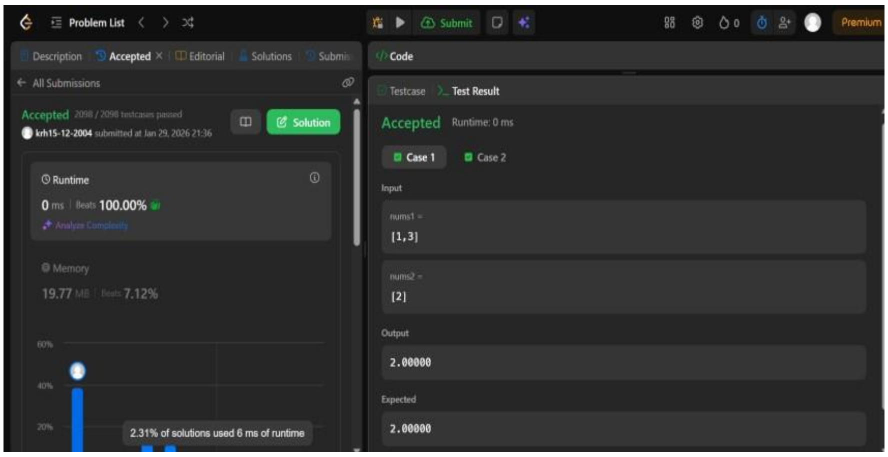
  - If the length is odd, the middle element is returned as the median.
  - If the length is even, the average of the two middle elements is computed and returned.

❖ Code:-

```
class Solution(object):
    def findMedianSortedArrays(self, nums1, nums2):
        num3 = nums1 + nums2
        num4 = sorted(num3)
        n = len(num4)
        i = n // 2

        if n % 2 == 0:
            return (num4[i] + num4[i - 1]) / 2.0
        else:
            return float(num4[i])
```

❖ Output



Accepted 2098 / 2098 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:36

Runtime
0 ms | Beats 100.00%

Memory
19.77 MB | Beats 7.12%

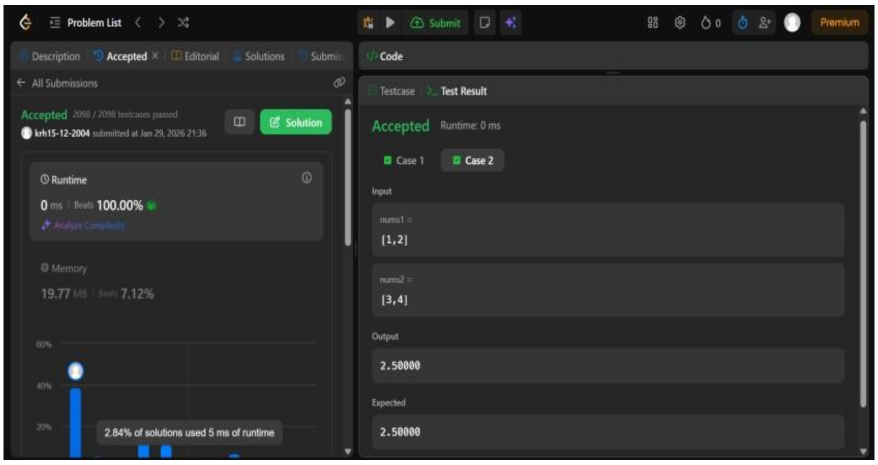
2.31% of solutions used 6 ms of runtime

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums1 =
```
[1,3]
```
nums2 =
```
[2]
```
Output
```
2.00000
```
Expected
```
2.00000
```



Accepted 2098 / 2098 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:36

Runtime
0 ms | Beats 100.00%

Memory
19.77 MB | Beats 7.12%

2.84% of solutions used 5 ms of runtime

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums1 =
```
[1,2]
```
nums2 =
```
[3,4]
```
Output
```
2.50000
```
Expected
```
2.50000
```

Algorithm:

1. Accept two sorted arrays nums1 and nums2.
2. Merge both arrays into a single list.
3. Sort the merged list.
4. Find the total number of elements.
5. If the total count is:
6. Odd → return the middle element.
7. Even → return the average of the two middle elements.
8. Display the median value.

Time Complexity:

O((m + n) log(m + n))

Space Complexity:

O(m + n)

▪ Problem 2

Problem Name: Reverse Nodes in k Group (From LeetCode Problem Number 25)

Difficulty Level: Hard **Problem Description:**

Given the head of a linked list, reverse the nodes of the list k at a time, and return *the modified list*. k is a positive integer and is less than or equal to the length of the linked list. If the number of nodes is not a multiple of k then left-out nodes, in the end, should remain as it is.

Approach Used:

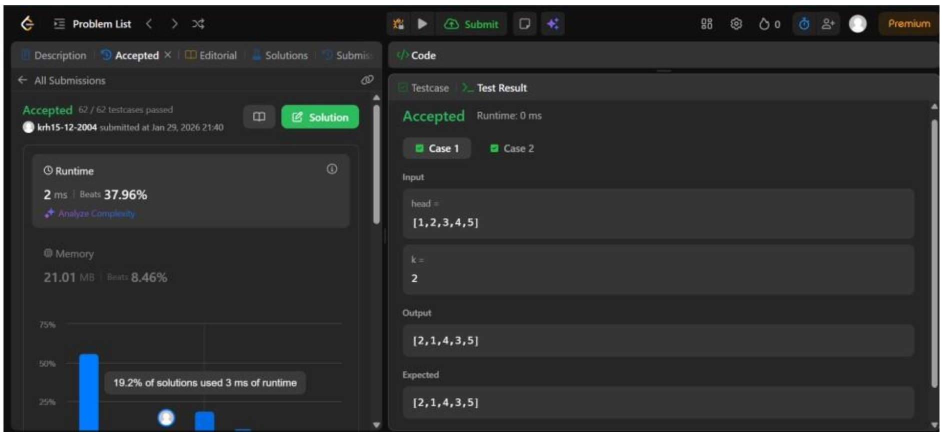
- Step-by-step logic:- It follows a clear **step-by-step logic** by first extracting values, then reversing in groups, and finally rebuilding the list.
- Efficient coding approach:- It uses an **efficient coding approach** by simplifying linked list operations through array manipulation.

❖ Code:-

```
class Solution(object):
    def reverseKGroup(self, head, k):
        arr = []
        curr = head
        while curr:
            arr.append(curr.val)
            curr = curr.next
        for i in range(0, len(arr), k):
            if i + k <= len(arr):
                arr[i:i + k] = arr[i:i + k][::-1]
        dummy = ListNode(0)
        curr = dummy

        for val in arr:
            curr.next = ListNode(val)
            curr = curr.next
        return dummy.next
```

❖ **Output:-**



Accepted 62 / 62 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:40

Runtime
2 ms | Beats 37.96%

Memory
21.01 MB | Beats 8.46%

19.2% of solutions used 3 ms of runtime

Accepted Runtime: 0 ms
Case 1 Case 2

Input

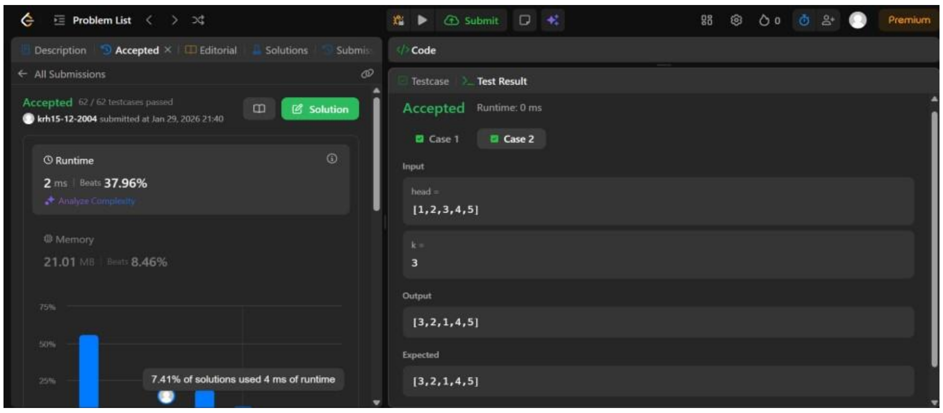
```
head =
[1,2,3,4,5]
k =
2
```

Output

```
[2,1,4,3,5]
```

Expected

```
[2,1,4,3,5]
```



Accepted 62 / 62 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:40

Runtime
2 ms | Beats 37.96%

Memory
21.01 MB | Beats 8.46%

7.41% of solutions used 4 ms of runtime

Accepted Runtime: 0 ms
Case 1 Case 2

Input

```
head =
[1,2,3,4,5]
k =
3
```

Output

```
[3,2,1,4,5]
```

Expected

```
[3,2,1,4,5]
```

Algorithm:

1. Accept the head of the linked list and an integer k.
2. Traverse the linked list and store all node values in an array.
3. Traverse the array in steps of size k.
4. For each group of size k, reverse the elements.
5. Leave the remaining elements unchanged if they are less than k.
6. Create a new linked list using the modified array.
7. Return the head of the updated linked list.

Time Complexity:

O(n)

Space Complexity:

O(n)

▪ Problem 3:-

Problem Name: First Missing Positive (From LeetCode Problem Number 41)

Difficulty Level: Hard

Problem Description:-

Given an unsorted integer array **nums**, find the smallest missing positive integer. The array may contain **negative numbers**, **zeros**, and **duplicates**. We have to implement an algorithm that runs in **O(n)** time and uses **constant extra space**.

Approach Used:

- **Swapping Technique:** A while loop is used to repeatedly swap elements into their correct positions until all possible numbers are in place.
- **Minimal space usage:**
  It avoids using extra data structures by reusing the original array to track the presence of values.

❖ Code:-

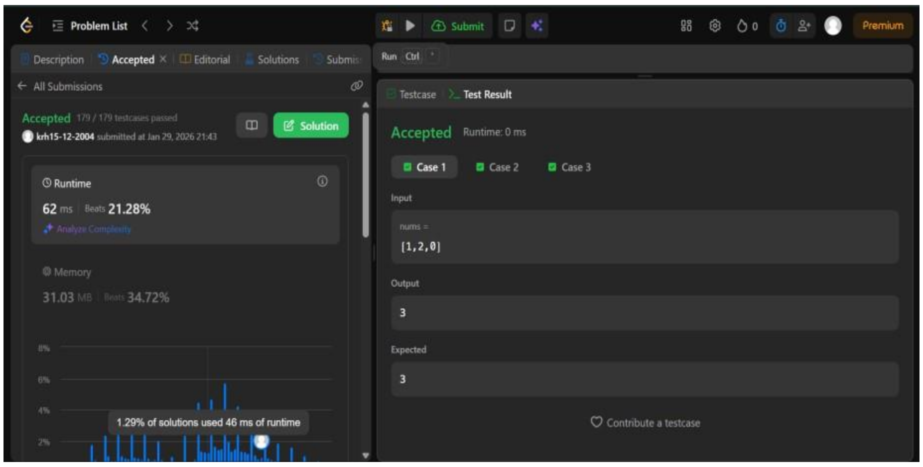
```
class Solution:
    def firstMissingPositive(self, nums):
        n = len(nums)

        for i in range(n):
            while 1 <= nums[i] <= n and nums[nums[i] - 1] != nums[i]:
                nums[nums[i] - 1], nums[i] = nums[i], nums[nums[i] - 1]

        for i in range(n):
            if nums[i] != i + 1:
                return i + 1

        return n + 1
```

❖ **<u>Output:-</u>**



Accepted 179 / 179 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:43

Runtime
62 ms | Beats 21.28%

Memory
31.03 MB | Beats 34.72%

1.29% of solutions used 46 ms of runtime

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

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

Output
```
3
```

Expected
```
3
```

Algorithm:

1. Accept the integer array nums.
2. Traverse the array and place each number x (where $1 \le x \le n$) at index $x - 1$.
3. Continue swapping elements until every valid number is in its correct position.
4. Traverse the array again.
5. If an index i is found such that $nums[i] \neq i + 1$, return $i + 1$.
6. If all elements are correctly placed, return $n + 1$.

Time Complexity:

O(n)

Space Complexity:

O(1)

▪ Problem 4:-

Problem Name: Smallest Range Covering Elements From K Lists(From LeetCode Problem Number 632)

Difficulty Level: Hard

Problem Description:-

We have k lists of sorted integers in non-decreasing order. Find the smallest range that includes at least one number from each of the k lists is included in the range

Approach Used:

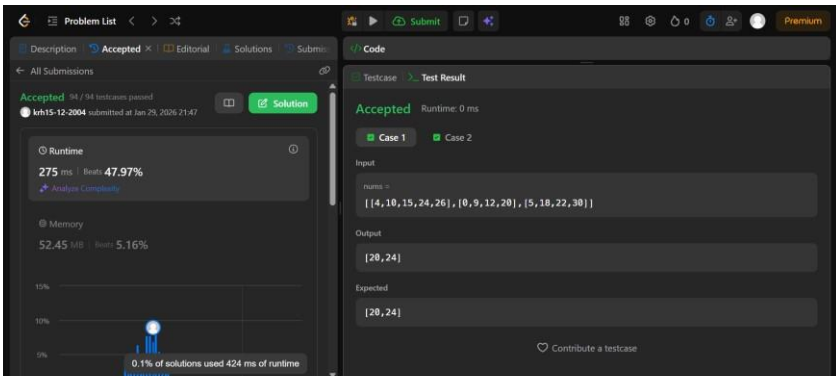
- **Cover Tracking:** Keep count of covered lists to know when the window is valid.
- **Sliding Window (Two-Pointer Technique):** Expand and shrink a window to find minimum range.

❖ Code:-

```
class Solution:
    def smallestRange(self, nums):
        arr = sorted((v, i) for i in range(len(nums)) for v in nums[i])
        cnt = {}
        left = 0
        cover = 0
        ans = [-10**9, 10**9]
        for r in range(len(arr)):
            cnt[arr[r][1]] = cnt.get(arr[r][1], 0) + 1

            if cnt[arr[r][1]] == 1:
                cover += 1
            while cover == len(nums):
                if arr[r][0] - arr[left][0] < ans[1] - ans[0]:
                    ans = [arr[left][0], arr[r][0]]
                cnt[arr[left][1]] -= 1
                if cnt[arr[left][1]] == 0:
                    cover -= 1
                left += 1
        return ans
```

❖ Output:-



Accepted 94 / 94 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:47

Runtime
275 ms | Beats 47.97%

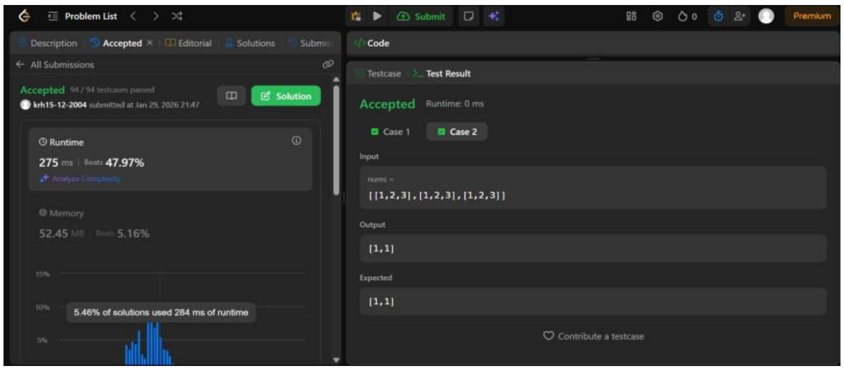
Memory
52.45 MB | Beats 5.16%

0.1% of solutions used 424 ms of runtime

Accepted Runtime: 0 ms

Case 1 Case 2

Input
```
nums =
[[4,10,15,24,26],[0,9,12,20],[5,18,22,30]]
```
Output
```
[20,24]
```
Expected
```
[20,24]
```



Accepted 94 / 94 testcases passed
krh15-12-2004 submitted at Jan 29, 2026 21:47

Runtime
275 ms | Beats 47.97%

Memory
52.45 MB | Beats 5.16%

5.46% of solutions used 284 ms of runtime

Accepted Runtime: 0 ms

Case 1 Case 2

Input
```
nums =
[[1,2,3],[1,2,3],[1,2,3]]
```
Output
```
[1,1]
```
Expected
```
[1,1]
```

Algorithm:

1. Combine all elements from the given k lists into a single list along with their list index.
2. Sort the combined list based on values.
3. Use two pointers (sliding window technique) to find the smallest range.
4. Maintain a count of elements from each list in the current window.
5. Expand the right pointer until all lists are covered.
6. Shrink the left pointer to minimize the range.
7. Return the smallest valid range.

Time Complexity:
O(N log N)

Space Complexity:
O(N)

5.Individual Contribution

Team

Member	**Contribution**
Navaneeth Gowda G	Problem analysis, algorithm design, and coding of **“Reverse Nodes in k-Group”** and **“Median Of Two Sorted Arrays”**. Responsible for understanding the problems, devising efficient approaches, and implementing initial solutions.
Karthikeya R Hegde	Testing, code optimization, validation, and documentation for **“Smallest Range Covering Elements From K Lists”** and **“First Missing Positive”**. Ensured correctness of solutions by running multiple test cases, improving performance, and compiling the final report.

6.Learning Outcomes

- After completing this Application Activity Task, the following learning outcomes were achieved:
- Gained a clear understanding of problem-solving techniques using Python.
- Improved logical thinking and analytical skills by solving complex algorithmic problems.
- Learned how to apply data structures such as arrays and linked lists effectively.
- Understood the importance of time and space complexity in designing efficient algorithms.
- Gained hands-on experience using the LeetCode platform for competitive programming.
- Enhanced debugging and code optimization skills.
- Developed teamwork and collaborative problem-solving abilities.
- Improved confidence in implementing real-world algorithmic solutions.

7.Conclusion

This problem-solving activity helped in strengthening programming concepts and logical thinking. Using LeetCode and Google Colab provided hands-on experience in solving realworld coding problems. Teamwork played an important role in completing the tasks effectively.

8.References

LeetCode – Online Coding Platform
https://leetcode.com/
Python Official Documentation
https://docs.python.org/3/
GeeksforGeeks – Data Structures and Algorithms
https://www.geeksforgeeks.org/
Introduction to Algorithms – Thomas H. Cormen et al.
Google Colab Documentation
https://colab.research.google.com/