Fast Learner Assignment

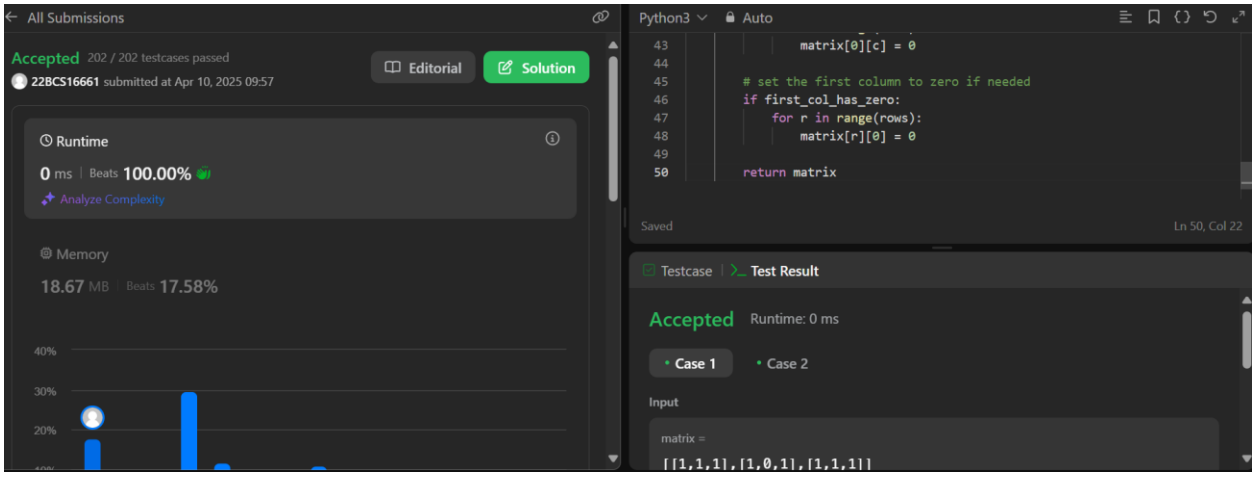
Student Name: Parth Arora
UID: 22BCS16661
Branch: BE-CSE
Section/Group: 22BCS_NTPP-602-A
Semester: 6th
Date of Performance: 11/04/2025
Subject Name: AP LAB - II
Subject Code: 22CSP-351

1. **Aim:** To modify the given matrix such that if an element is 0, its entire row and column are set to 0.
2. **Source Code:**

```
class Solution:
    def setZeroes(self, matrix: List[List[int]]) -> None:
        rows = len(matrix)
        cols = len(matrix[0])

        first_row_has_zero = False
        first_col_has_zero = False

        # check if the first row contains zero
        for c in range(cols):
            if matrix[0][c] == 0:
                first_row_has_zero = True
                break

        # check if the first column contains zero
        for r in range(rows):
            if matrix[r][0] == 0:
                first_col_has_zero = True
                break

        # use the first row and column as a note
        for r in range(1, rows):
            for c in range(1, cols):
                if matrix[r][c] == 0:
                    matrix[r][0] = 0
                    matrix[0][c] = 0
```

```
# set the marked rows to zero
for r in range(1, rows):
    if matrix[r][0] == 0:
        for c in range(1, cols):
            matrix[r][c] = 0

# set the marked columns to zero
for c in range(1, cols):
    if matrix[0][c] == 0:
        for r in range(1, rows):
            matrix[r][c] = 0

# set the first row to zero if needed
if first_row_has_zero:
    for c in range(cols):
        matrix[0][c] = 0

# set the first column to zero if needed
if first_col_has_zero:
    for r in range(rows):
        matrix[r][0] = 0

return matrix
```

3. Screenshots of outputs:



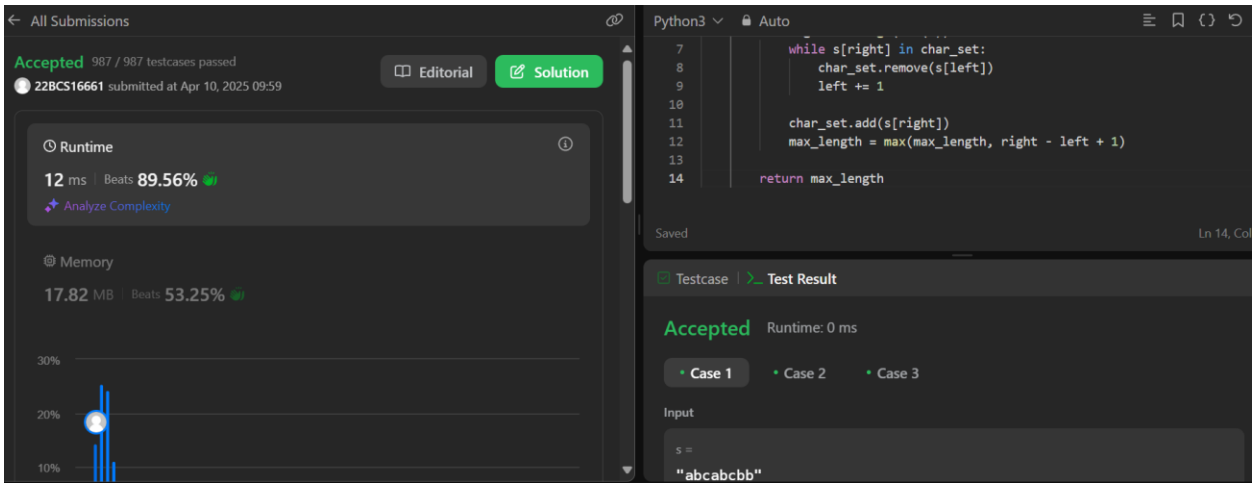
2.

Aim: To find the length of the longest substring in a given string without any repeating characters.

Source Code:

```
class Solution:
    def lengthOfLongestSubstring(self, s: str) -> int:
        left = max_length = 0
        char_set = set()

        for right in range(len(s)):
            while s[right] in char_set:
                char_set.remove(s[left])
                left += 1

            char_set.add(s[right])
            max_length = max(max_length, right - left + 1)

        return max_length
```

Screenshots of outputs:



3.

Aim: To reverse a part of a linked list from position left to right.

Source Code:

```
class Solution:
    def reverseBetween(self, head: Optional[ListNode], left: int, right: int) -> Optional[ListNode]:

        if not head or left == right:
            return head

        dummy = ListNode(0, head)
        prev = dummy

        for _ in range(left - 1):
            prev = prev.next

        cur = prev.next
        for _ in range(right - left):
            temp = cur.next
            cur.next = temp.next
            temp.next = prev.next
            prev.next = temp

        return dummy.next
```

4. Screenshots of outputs: