Experiment 4

Student Name: Ansh **UID: 22BCS13469**
Branch: CSE **Section/Group: 608-B**
Semester: 6th **Date of Performance:13/3/25**
Subject Name: Advanced Programming - 2 **Subject Code: 22CSH-351**

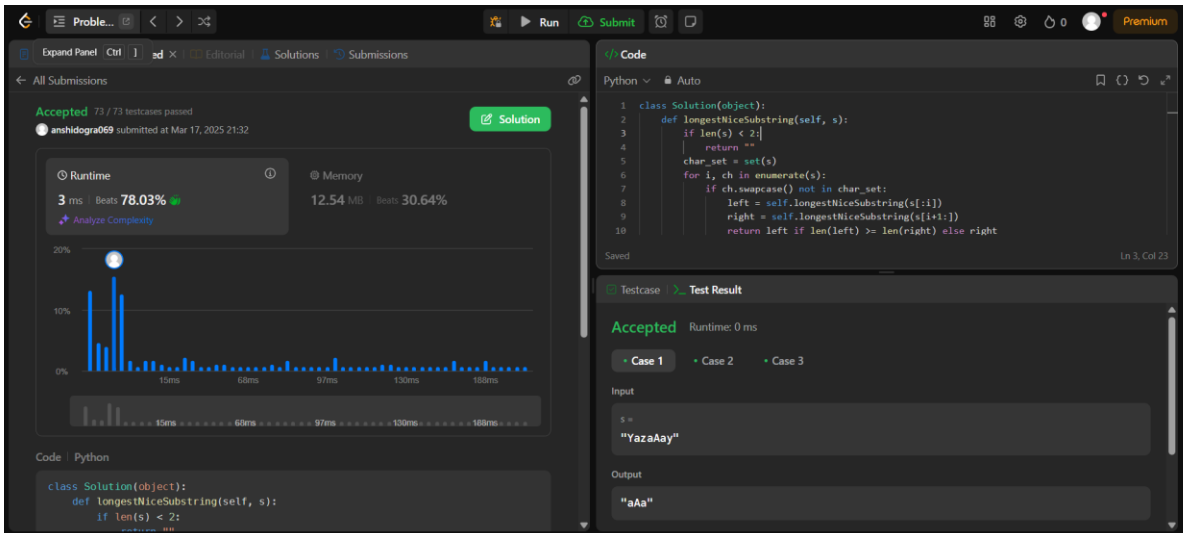
Ques 1:

Aim: Longest Nice Substring:

Code:

```
class Solution(object):
    def longestNiceSubstring(self, s):
        if len(s) < 2:
            return ""
        char_set = set(s)
        for i, ch in enumerate(s):
            if ch.swapcase() not in char_set:
                left = self.longestNiceSubstring(s[:i])
                right = self.longestNiceSubstring(s[i+1:])
                return left if len(left) >= len(right) else right
        return s
```

Submission Screenshot:



Accepted 73 / 73 testcases passed
anshidogra069 submitted at Mar 17, 2025 21:32

Runtime 3 ms | Beats 78.03%
Memory 12.54 MB | Beats 30.64%

```
class Solution(object):
    def longestNiceSubstring(self, s):
        if len(s) < 2:
            return ""
        char_set = set(s)
        for i, ch in enumerate(s):
            if ch.swapcase() not in char_set:
                left = self.longestNiceSubstring(s[:i])
                right = self.longestNiceSubstring(s[i+1:])
                return left if len(left) >= len(right) else right
```

Saved Ln 3, Col 23

Accepted Runtime: 0 ms

Input
```
s =
"YazaAay"
```
Output
```
"aAa"
```

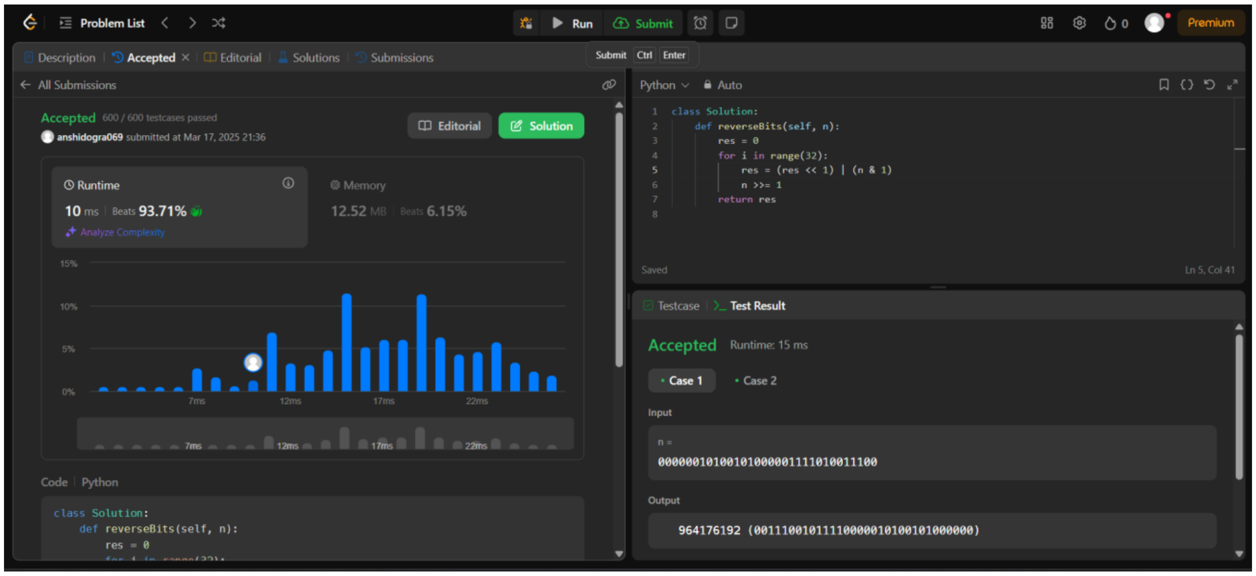
Ques 2:

Aim: Reverse Bits:

Code:

```
class Solution:
    def reverseBits(self, n):
        res = 0
        for i in range(32):
            res = (res << 1) | (n & 1)
            n >>= 1
        return res
```

Submission Screenshot:



Accepted 600 / 600 testcases passed

anshidogra069 submitted at Mar 17, 2025 21:36

10 ms | Beats 93.71%

12.52 MB | Beats 6.15%

```
class Solution:
    def reverseBits(self, n):
        res = 0
        for i in range(32):
            res = (res << 1) | (n & 1)
            n >>= 1
        return res
```

Saved

Ln 5, Col 41

Accepted Runtime: 15 ms

Input

```
n =
00000010100101000001111010011100
```

Output

```
964176192 (00111001011110000010100101000000)
```

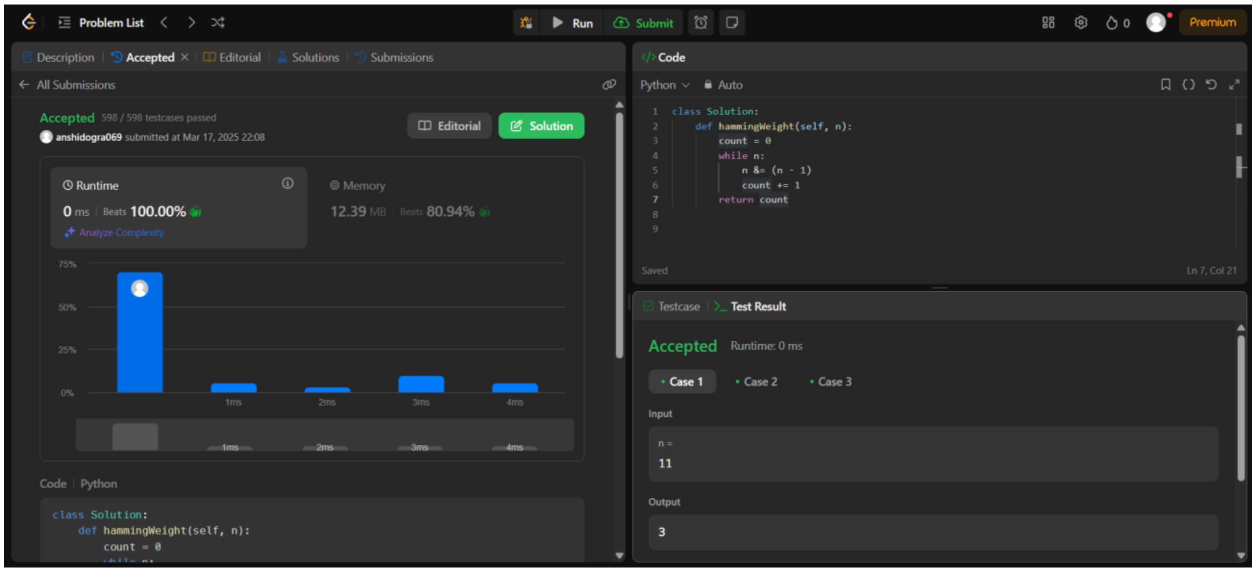
Ques 3:

Aim: Number of 1 Bits:

Code:

```
class Solution:
    def hammingWeight(self, n):
        count = 0
        while n:
            n &= (n - 1)
            count += 1
        return count
```

Submission Screenshot:



Accepted 598 / 598 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:08

Runtime: 0 ms | Beats 100.00%
Memory: 12.39 MB | Beats 80.94%

Code | Python

```
class Solution:
    def hammingWeight(self, n):
        count = 0
        while n:
            n &= (n - 1)
            count += 1
        return count
```

Saved
Ln 7, Col 21

Accepted Runtime: 0 ms

Input
n =
11

Output
3

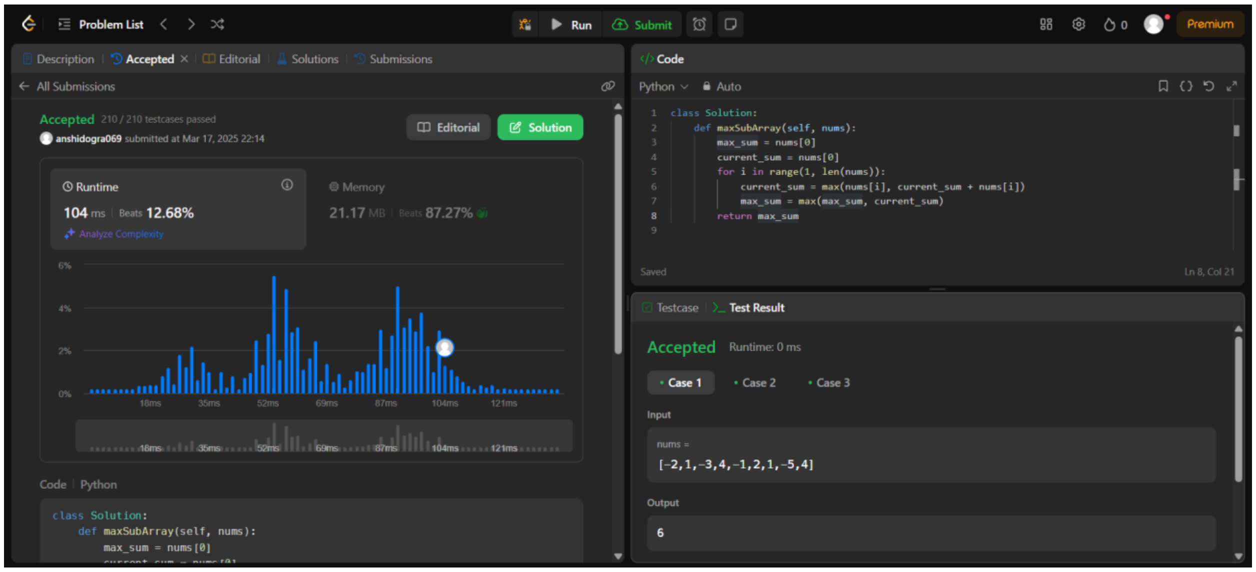
Ques 4:

Aim: Maximum Subarray:

Code:

```
class Solution:
    def maxSubArray(self, nums):
        max_sum = nums[0]
        current_sum = nums[0]
        for i in range(1, len(nums)):
            current_sum = max(nums[i], current_sum + nums[i])
            max_sum = max(max_sum, current_sum)
        return max_sum
```

Submission Screenshot:



Accepted 210 / 210 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:14

Runtime
104 ms | Beats 12.68%
Analyze Complexity

Memory
21.17 MB | Beats 87.27%

```
class Solution:
    def maxSubArray(self, nums):
        max_sum = nums[0]
        current_sum = nums[0]
        for i in range(1, len(nums)):
            current_sum = max(nums[i], current_sum + nums[i])
            max_sum = max(max_sum, current_sum)
        return max_sum
```

Saved
Ln 8, Col 21

Accepted Runtime: 0 ms
Case 1 Case 2 Case 3

Input
```
nums =
[-2,1,-3,4,-1,2,1,-5,4]
```

Output
```
6
```

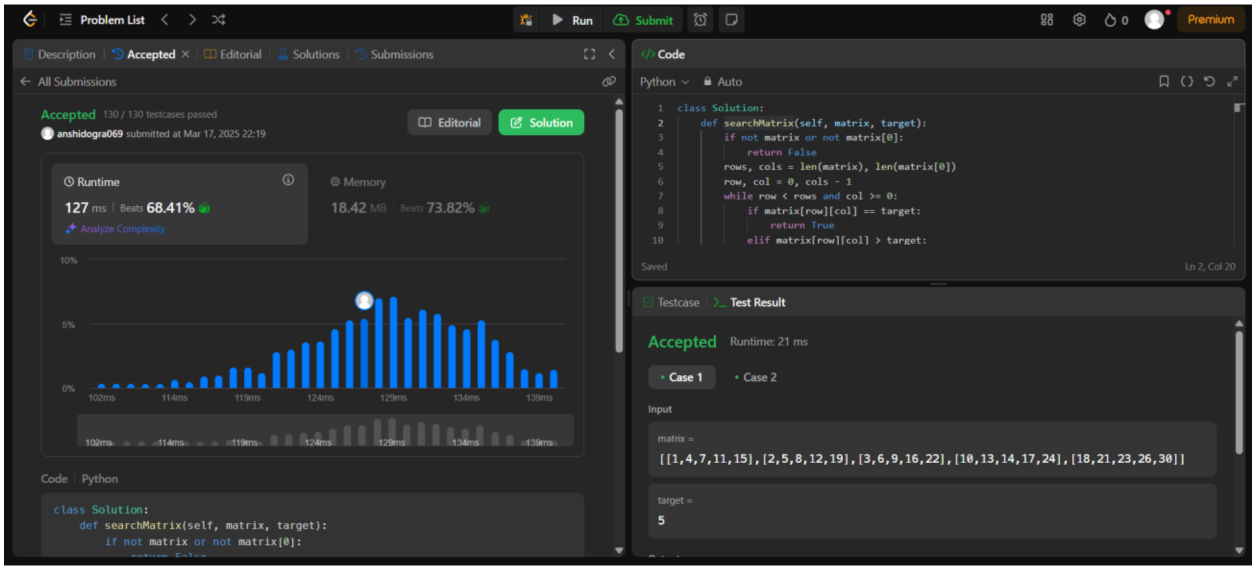
Ques 5:

Aim: Search a 2D Matrix II:

Code:

```
class Solution:
    def searchMatrix(self, matrix, target):
        if not matrix or not matrix[0]:
            return False
        rows, cols = len(matrix), len(matrix[0])
        row, col = 0, cols - 1
        while row < rows and col >= 0:
            if matrix[row][col] == target:
                return True
            elif matrix[row][col] > target:
                col -= 1
            else:
                row += 1
        return False
```

Submission Screenshot:



Accepted 130 / 130 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:19

Runtime 127 ms | Beats 68.41%
Memory 18.42 MB | Beats 73.82%

Code | Python

```
class Solution:
    def searchMatrix(self, matrix, target):
        if not matrix or not matrix[0]:
```

```
class Solution:
    def searchMatrix(self, matrix, target):
        if not matrix or not matrix[0]:
            return False
        rows, cols = len(matrix), len(matrix[0])
        row, col = 0, cols - 1
        while row < rows and col >= 0:
            if matrix[row][col] == target:
                return True
            elif matrix[row][col] > target:
```

Accepted Runtime: 21 ms

Input

matrix =
[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]

target =
5

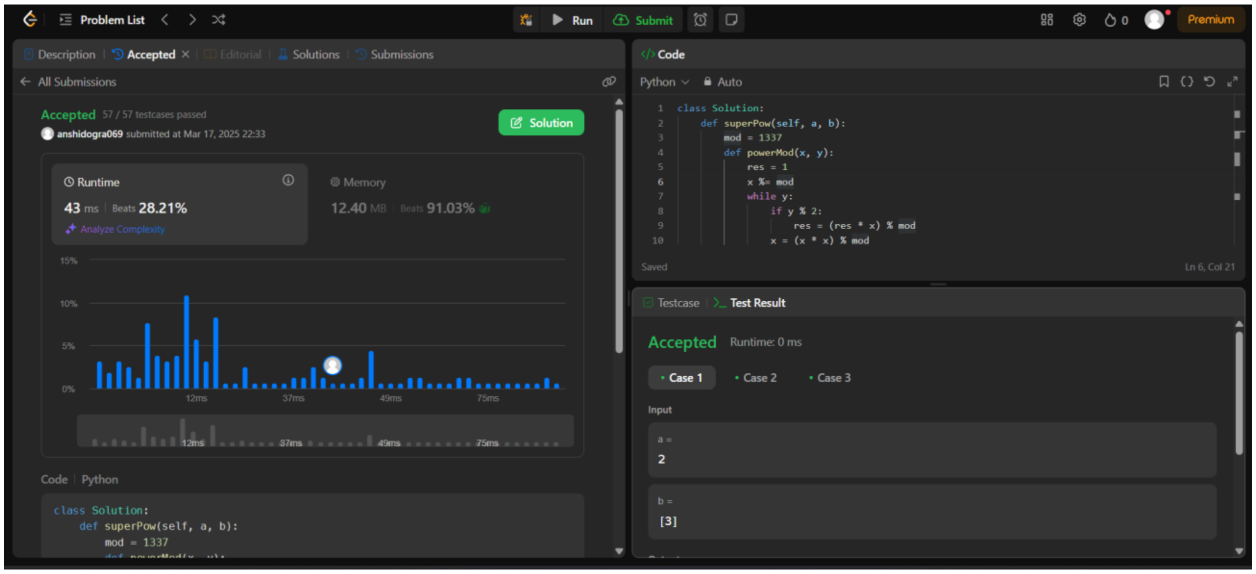
Ques 6:

Aim: Super Pow:

Code:

```
class Solution:
    def superPow(self, a, b):
        mod = 1337
        def powerMod(x, y):
            res = 1
            x %= mod
            while y:
                if y % 2:
                    res = (res * x) % mod
                x = (x * x) % mod
                y //= 2
            return res
        result = 1
        for digit in b:
            result = powerMod(result, 10) * powerMod(a, digit) % mod
        return result
```

Submission Screenshot:



Accepted 57 / 57 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:33

Runtime
43 ms | Beats 28.21%

Memory
12.40 MB | Beats 91.03%

Code | Python

```
class Solution:
    def superPow(self, a, b):
        mod = 1337
```

```
class Solution:
    def superPow(self, a, b):
        mod = 1337
        def powerMod(x, y):
            res = 1
            x %= mod
            while y:
                if y % 2:
                    res = (res * x) % mod
                x = (x * x) % mod
```

Saved   Ln 6, Col 21

Accepted Runtime: 0 ms

Input

a =
2

b =
[3]

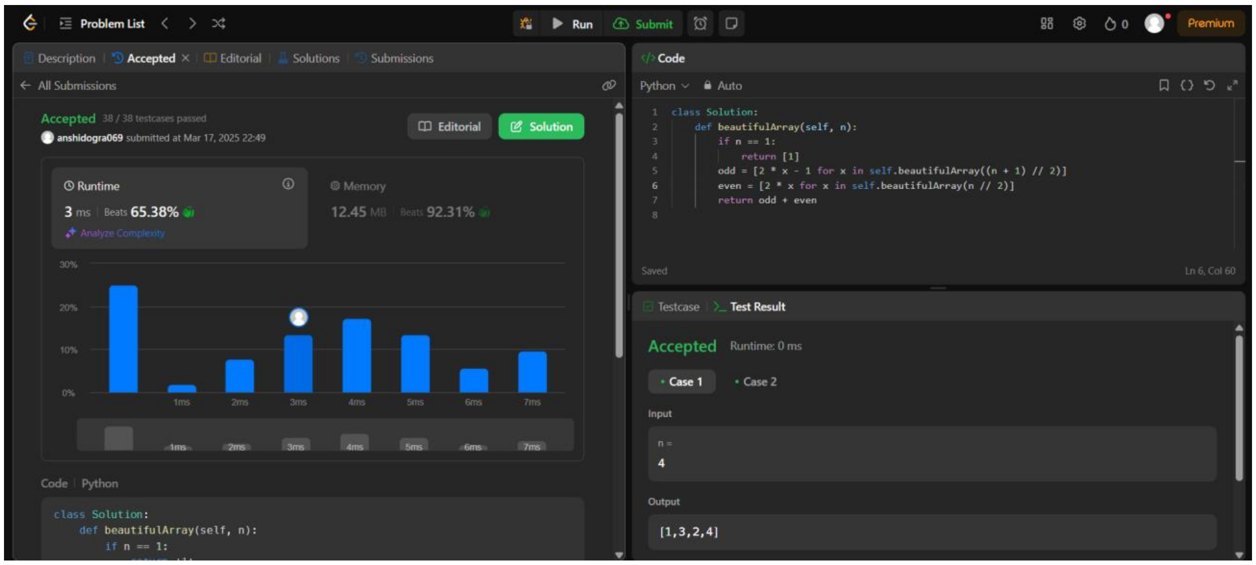
Ques 7:

Aim: Beautiful Array:

Code:

```
class Solution:
    def beautifulArray(self, n):
        if n == 1:
            return [1]
        odd = [2 * x - 1 for x in self.beautifulArray((n + 1) // 2)]
        even = [2 * x for x in self.beautifulArray(n // 2)]
        return odd + even
```

Submission Screenshot:



Accepted 38 / 38 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:49

Runtime
3 ms | Beats 65.38%

Memory
12.45 MB | Beats 92.31%

```
class Solution:
    def beautifulArray(self, n):
        if n == 1:
            return [1]
        odd = [2 * x - 1 for x in self.beautifulArray((n + 1) // 2)]
        even = [2 * x for x in self.beautifulArray(n // 2)]
        return odd + even
```

Saved
Ln 6, Col 60

Accepted Runtime: 0 ms

Input
n =
4

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

Code | Python

```
class Solution:
    def beautifulArray(self, n):
        if n == 1:
```

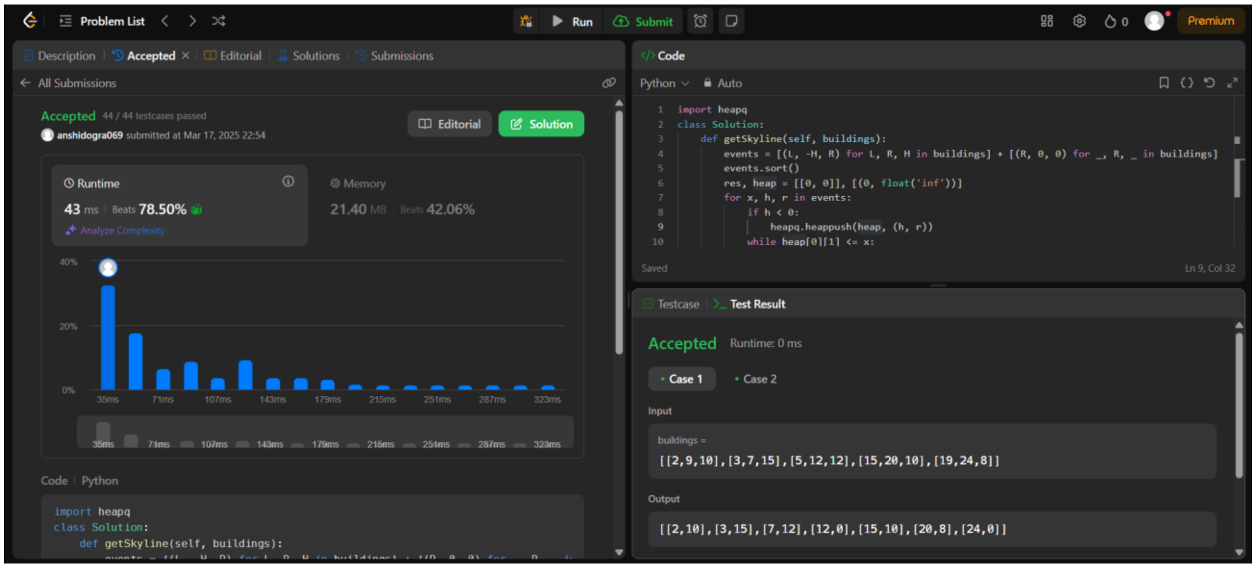
Ques 8:

Aim: The Skyline Problem:

Code:

```
import heapq
class Solution:
    def getSkyline(self, buildings):
        events = [(L, -H, R) for L, R, H in buildings] + [(R, 0, 0) for _, R, _ in buildings]
        events.sort()
        res, heap = [[0, 0]], [(0, float('inf'))]
        for x, h, r in events:
            if h < 0:
                heapq.heappush(heap, (h, r))
            while heap[0][1] <= x:
                heapq.heappop(heap)
            if res[-1][1] != -heap[0][0]:
                res.append([x, -heap[0][0]])
        return res[1:]
```

Submission Screenshot:



Accepted 44 / 44 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:54

Runtime
43 ms | Beats 78.50%

Memory
21.40 MB | Beats 42.06%

Code | Python

```
import heapq
class Solution:
    def getSkyline(self, buildings):
```

```
import heapq
class Solution:
    def getSkyline(self, buildings):
        events = [(L, -H, R) for L, R, H in buildings] + [(R, 0, 0) for _, R, _ in buildings]
        events.sort()
        res, heap = [[0, 0]], [(0, float('inf'))]
        for x, h, r in events:
            if h < 0:
                heapq.heappush(heap, (h, r))
            while heap[0][1] <= x:
```

Accepted Runtime: 0 ms

Input

buildings =
```
[[2,9,10],[3,7,15],[5,12,12],[15,20,10],[19,24,8]]
```

Output
```
[[2,10],[3,15],[7,12],[12,0],[15,10],[20,8],[24,0]]
```

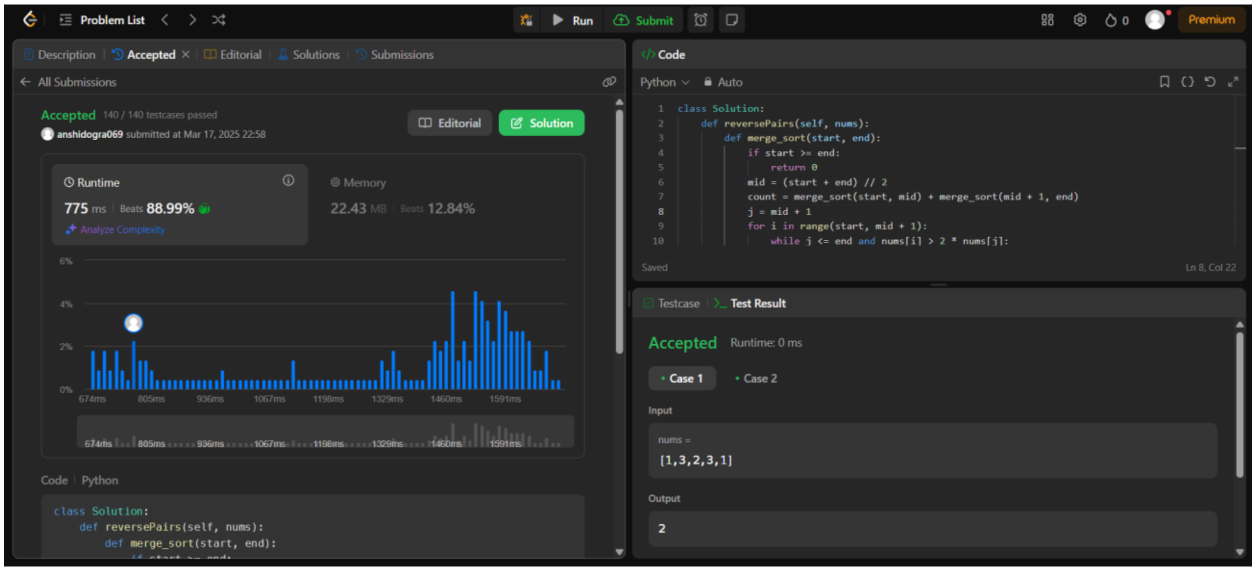
Ques 9:

Aim: Reverse Pairs:

Code:

```
class Solution:
    def reversePairs(self, nums):
        def merge_sort(start, end):
            if start >= end:
                return 0
            mid = (start + end) // 2
            count = merge_sort(start, mid) + merge_sort(mid + 1, end)
            j = mid + 1
            for i in range(start, mid + 1):
                while j <= end and nums[i] > 2 * nums[j]:
                    j += 1
                count += (j - (mid + 1))
            nums[start:end + 1] = sorted(nums[start:end + 1])
            return count
        return merge_sort(0, len(nums) - 1)
```

Submission Screenshot:



Accepted 140 / 140 testcases passed
anshidogra069 submitted at Mar 17, 2025 22:58

Runtime
775 ms | Beats 88.99%

Memory
22.43 MB | Beats 12.84%

Code | Python

```
class Solution:
    def reversePairs(self, nums):
        def merge_sort(start, end):
```

```
class Solution:
    def reversePairs(self, nums):
        def merge_sort(start, end):
            if start >= end:
                return 0
            mid = (start + end) // 2
            count = merge_sort(start, mid) + merge_sort(mid + 1, end)
            j = mid + 1
            for i in range(start, mid + 1):
                while j <= end and nums[i] > 2 * nums[j]:
```

Accepted Runtime: 0 ms

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

Output
2

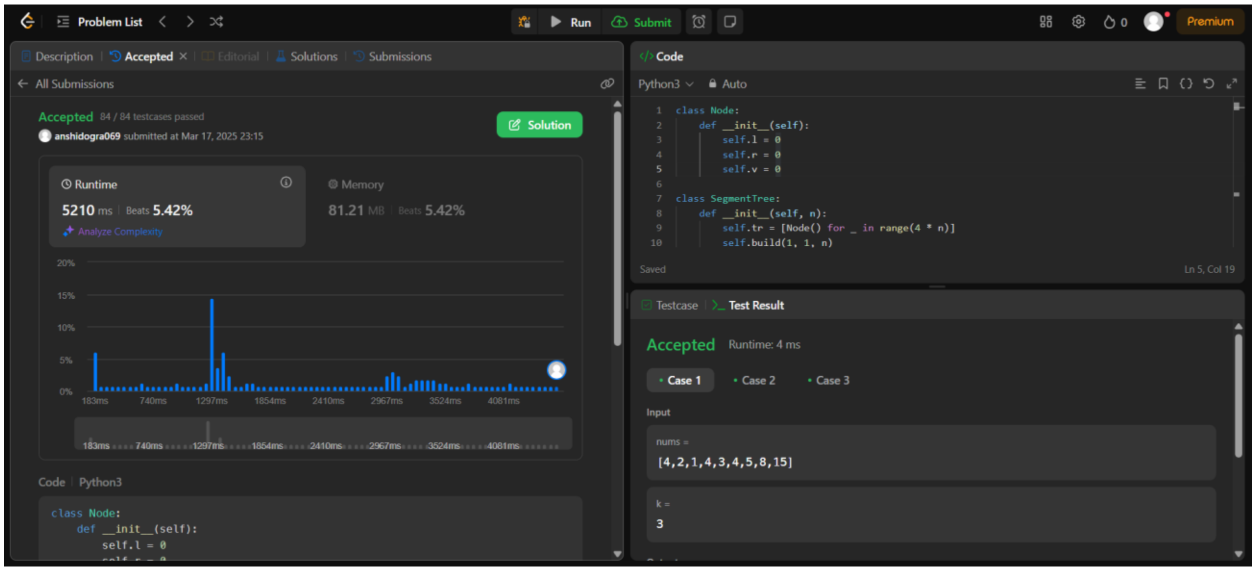
Ques 10:

Aim: Longest Increasing Subsequence II:

Code:

```
class Node:
    def __init__(self):
        self.l = 0
        self.r = 0
        self.v = 0

class SegmentTree:
    def __init__(self, n):
        self.tr = [Node() for _ in range(4 * n)]
        self.build(1, 1, n)

    def build(self, u, l, r):
        self.tr[u].l = l
        self.tr[u].r = r
        if l == r:
            return
        mid = (l + r) >> 1
        self.build(u << 1, l, mid)
        self.build(u << 1 | 1, mid + 1, r)

    def modify(self, u, x, v):
        if self.tr[u].l == x and self.tr[u].r == x:
            self.tr[u].v = v
            return
        mid = (self.tr[u].l + self.tr[u].r) >> 1
        if x <= mid:
            self.modify(u << 1, x, v)
        else:
            self.modify(u << 1 | 1, x, v)
        self.pushup(u)

    def pushup(self, u):
        self.tr[u].v = max(self.tr[u << 1].v, self.tr[u << 1 | 1].v)
    def query(self, u, l, r):
        if self.tr[u].l >= l and self.tr[u].r <= r:
            return self.tr[u].v
        mid = (self.tr[u].l + self.tr[u].r) >> 1
        v = 0
        if l <= mid:
            v = self.query(u << 1, l, r)
        if r > mid:
            v = max(v, self.query(u << 1 | 1, l, r))
        return v
class Solution:
    def lengthOfLIS(self, nums: List[int], k: int) -> int:
        tree = SegmentTree(max(nums))
        ans = 1
        for v in nums:
            t = tree.query(1, v - k, v - 1) + 1
            ans = max(ans, t)
            tree.modify(1, v, t)
        return ans
```

Submission Screenshot:



Accepted 84 / 84 testcases passed
anshidogra069 submitted at Mar 17, 2025 23:15

Runtime
5210 ms | Beats 5.42%

Memory
81.21 MB | Beats 5.42%

Code | Python3

```
class Node:
    def __init__(self):
        self.l = 0
```

```
class Node:
    def __init__(self):
        self.l = 0
        self.r = 0
        self.v = 0

class SegmentTree:
    def __init__(self, n):
        self.tr = [Node() for _ in range(4 * n)]
        self.build(1, 1, n)
```

Saved
Ln 5, Col 19

Accepted Runtime: 4 ms

Input

nums =
[4,2,1,4,3,4,5,8,15]

k =
3