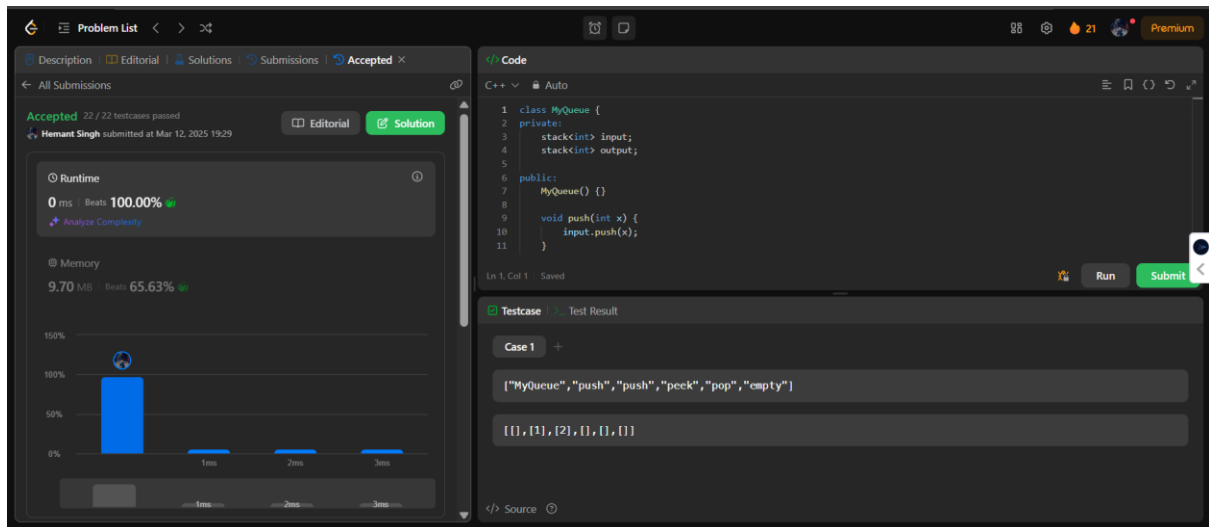


AP-EXPERIMENT-6

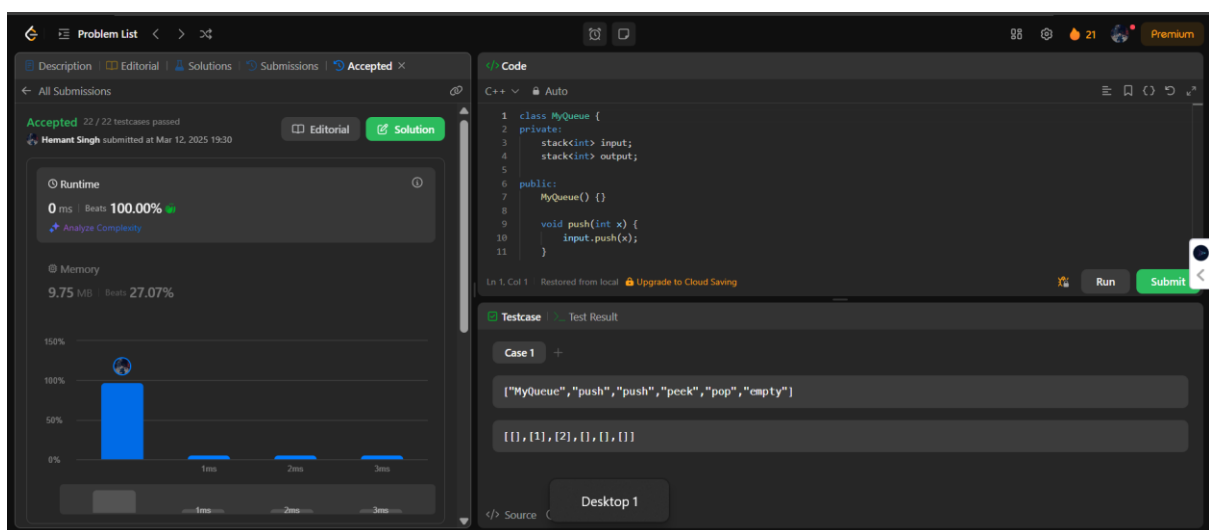
Name- Hemant Singh

UID-22BCS12820

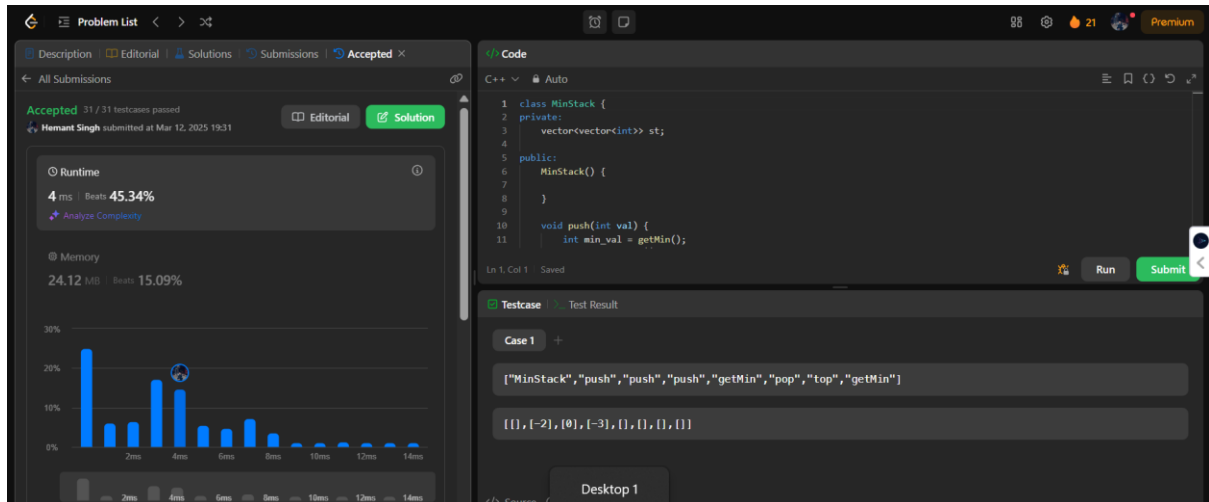
Q1. <https://leetcode.com/problems/implement-queue-using-stacks/submissions/1571436603/>



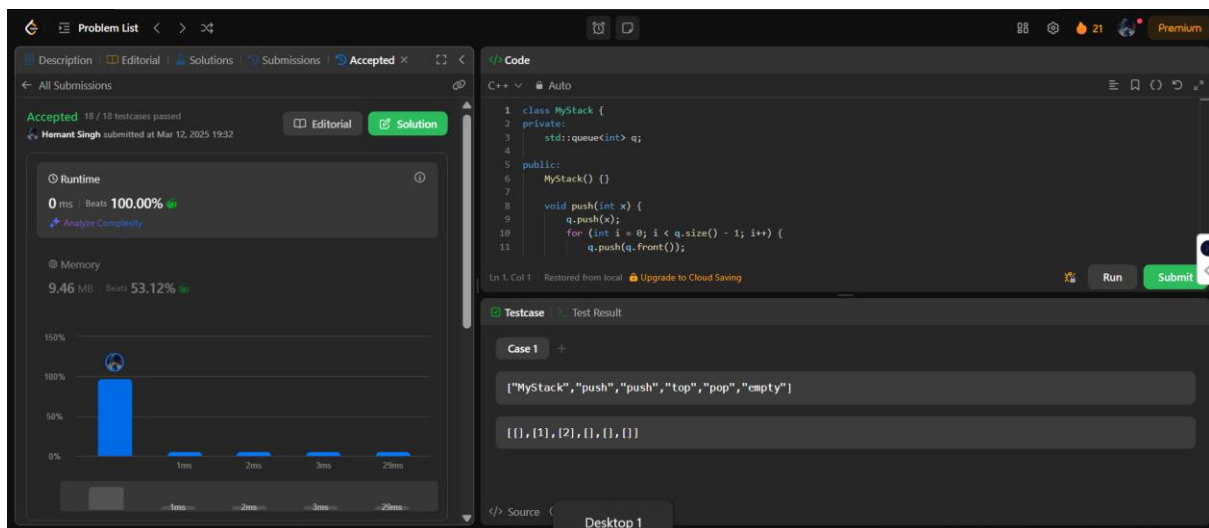
Q2. <https://leetcode.com/problems/implement-queue-using-stacks/submissions/1571437594/>



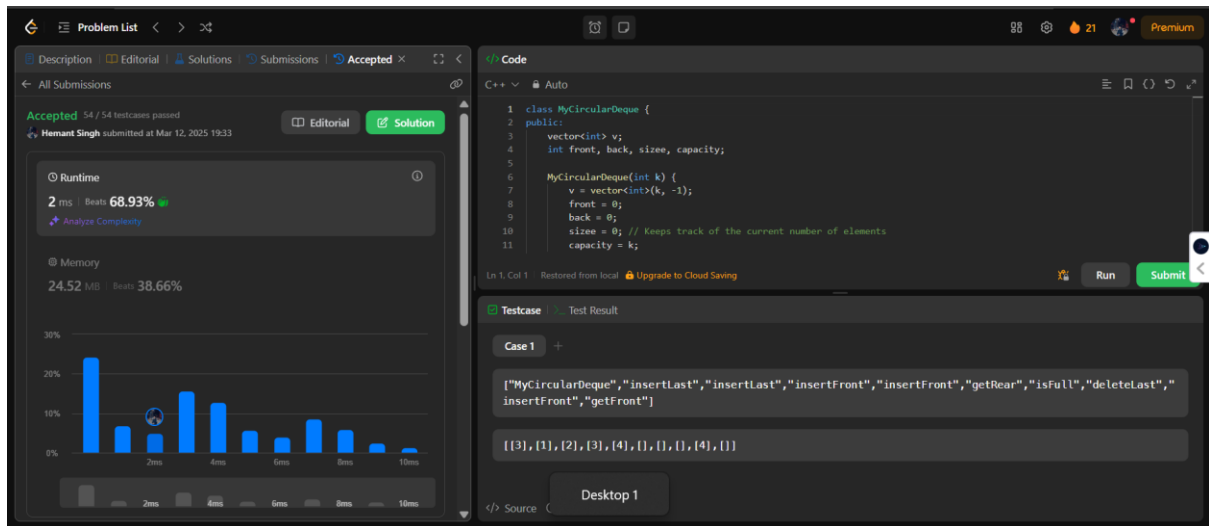
Q3. <https://leetcode.com/problems/min-stack/submissions/1571438671/>



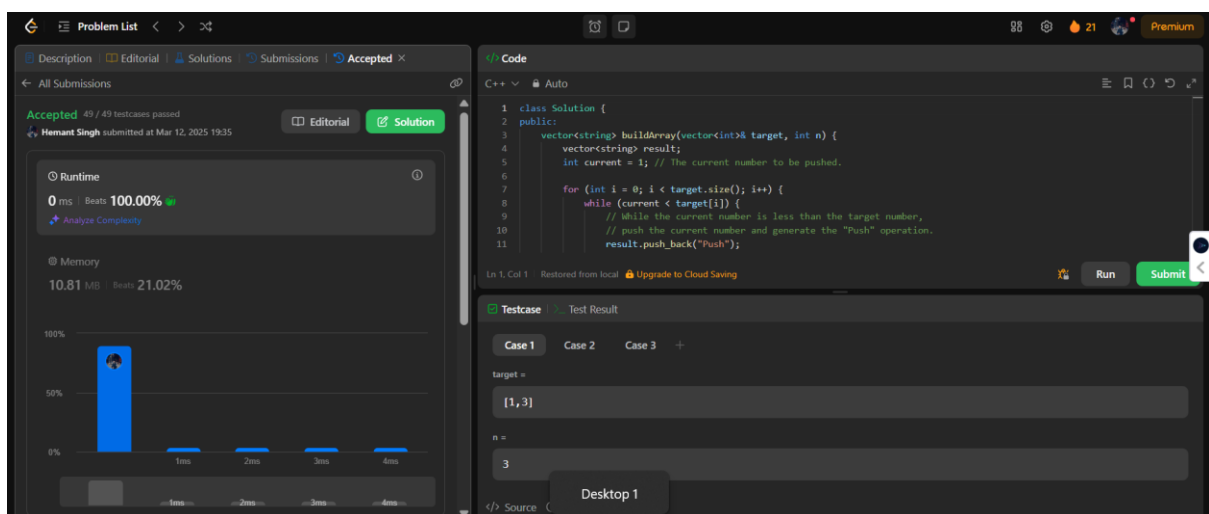
Q4. <https://leetcode.com/problems/implement-stack-using-queues/submissions/1571439512/>



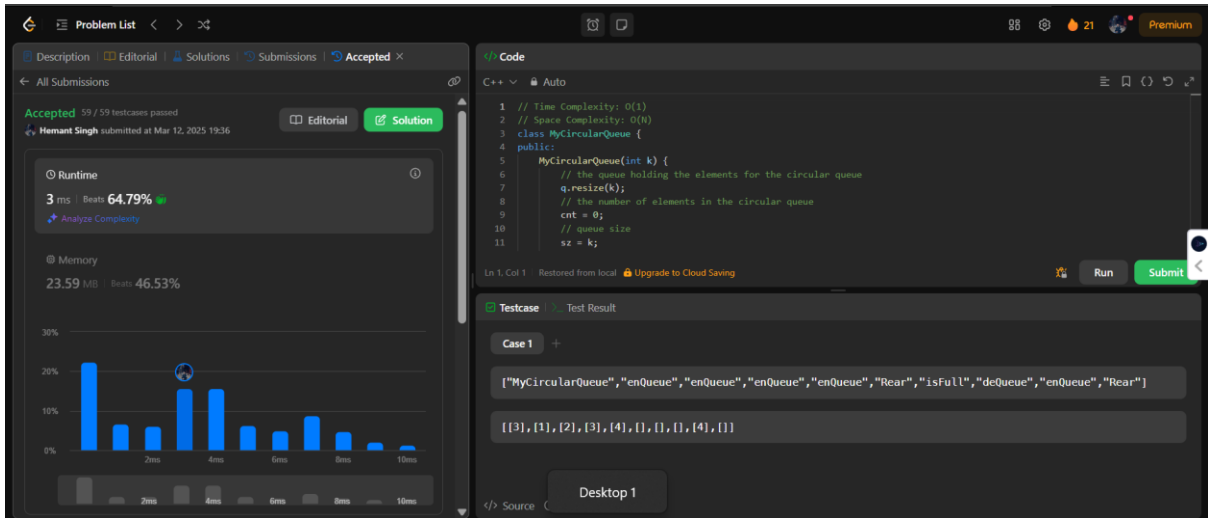
Q5. <https://leetcode.com/problems/design-circular-deque/submissions/1571440643/>



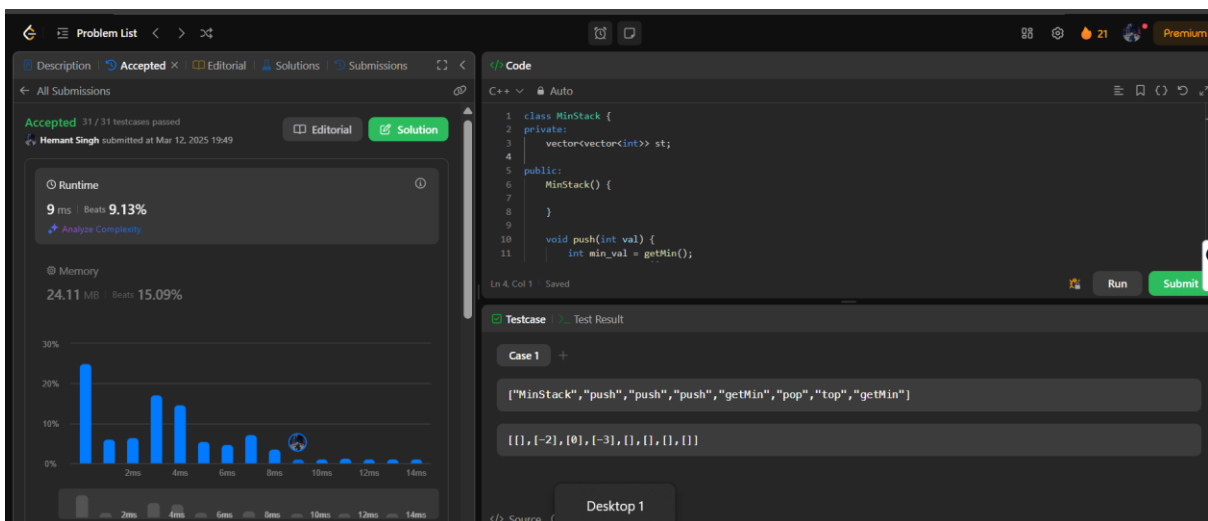
Q6. <https://leetcode.com/problems/build-an-array-with-stack-operations/submissions/1571442543/>



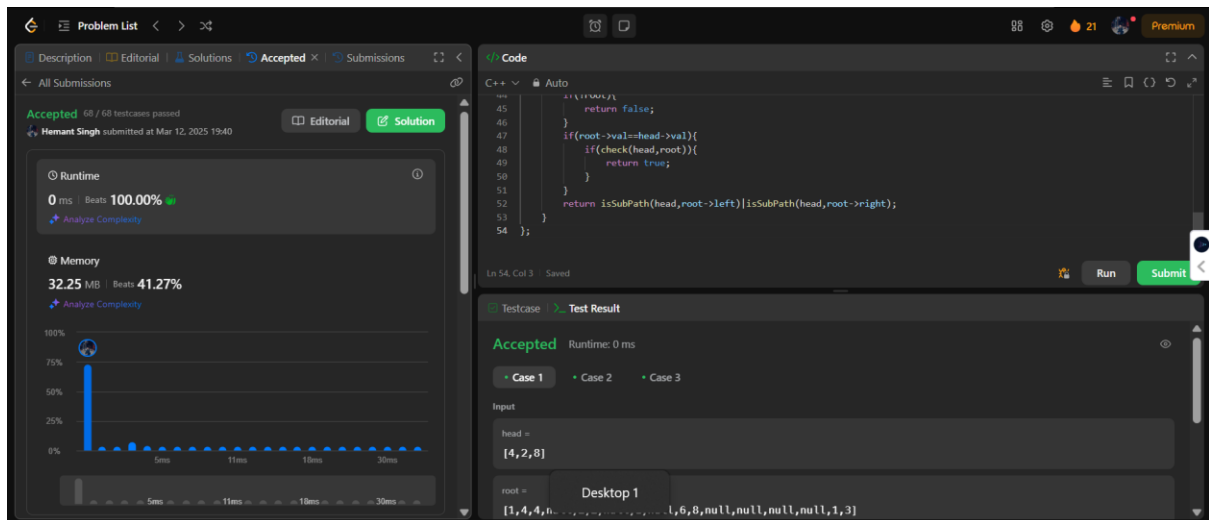
Q7. <https://leetcode.com/problems/design-circular-queue/submissions/1571443591/>



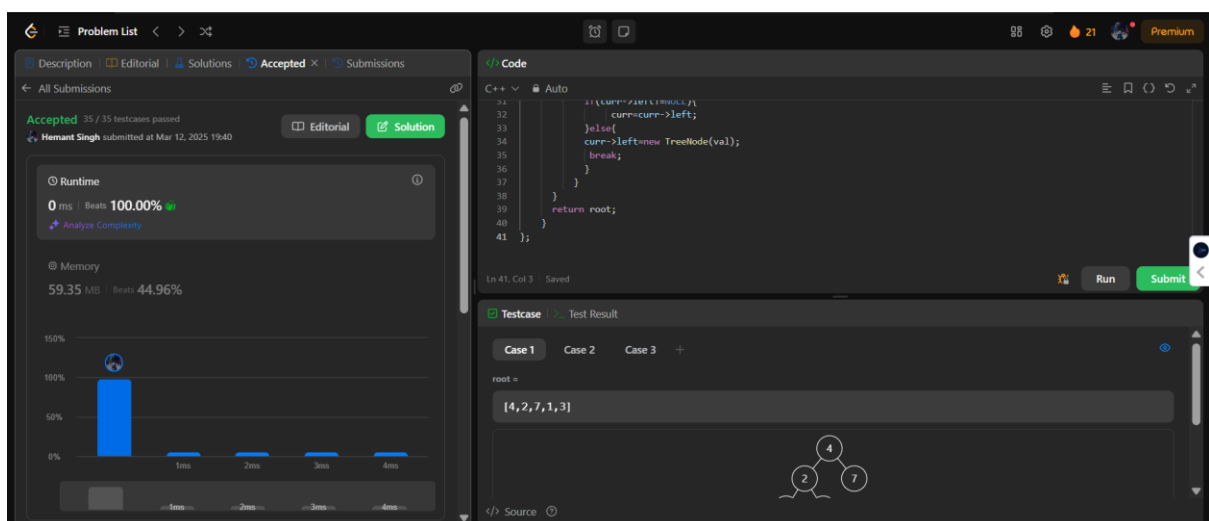
Q8. <https://leetcode.com/problems/min-stack/submissions/1571455259/>



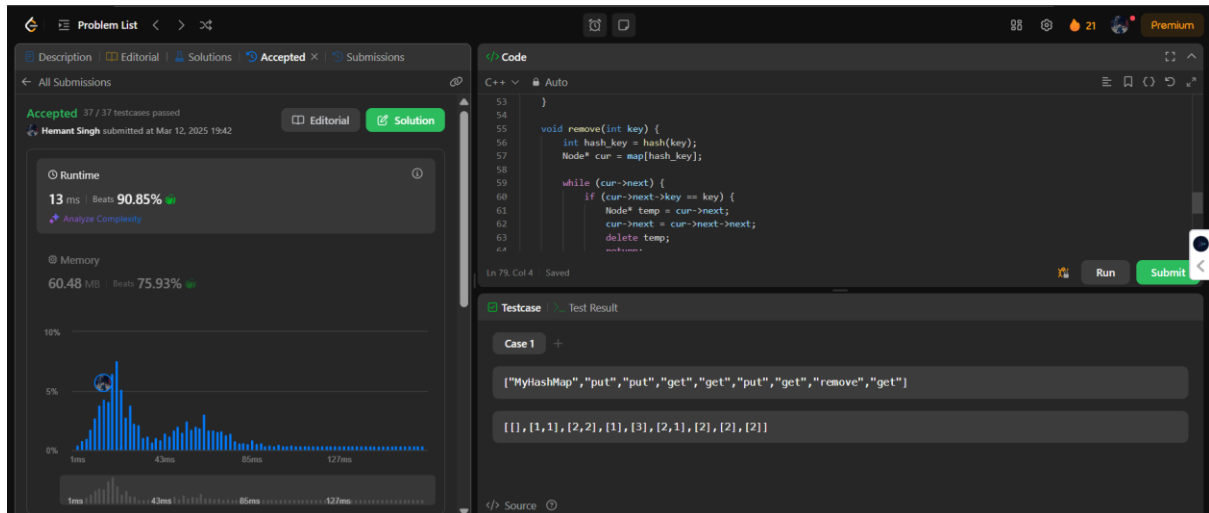
Q9. <https://leetcode.com/problems/linked-list-in-binary-tree/submissions/1571446726/>



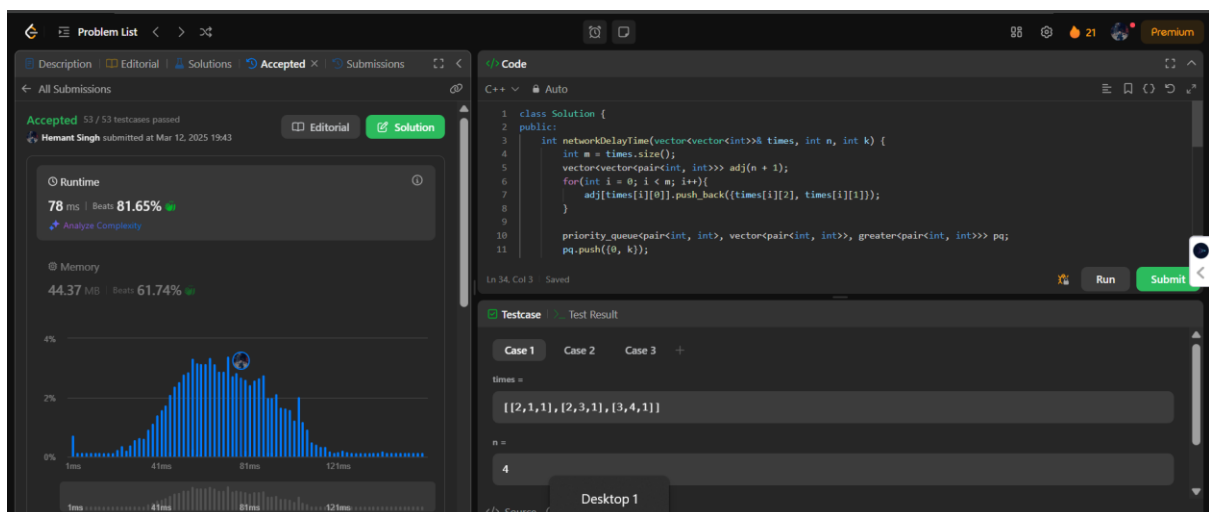
Q10. <https://leetcode.com/problems/insert-into-a-binary-search-tree/submissions/1571447402/>



Q11. <https://leetcode.com/problems/design-hashmap/submissions/1571448722/>



Q12. <https://leetcode.com/problems/network-delay-time/submissions/1571449591/>



Q13. <https://leetcode.com/problems/detect-cycles-in-2d-grid/submissions/1571459800/>

The screenshot shows the LeetCode submission interface for the problem "detect-cycles-in-2d-grid". The left sidebar displays the submission status as "Accepted" with 76/76 testcases passed. The runtime is 12 ms, beating 98.66% of submissions. The memory usage is 67.05 MB, beating 62.68%. A runtime graph shows the performance of the solution. The main area displays the C++ code, which uses a recursive DFS function to check for cycles in the grid. The code is as follows:

```
C++  
14  if (!vis[i][j]) {  
15      if (dfs(grid, i, j - 1, vis, i, j, c)) return true;  
16      } else if (i != pre_i || j != pre_j) {  
17          return true;  
18      }  
19  }  
20  return false;  
21  }  
22  public:  
23      bool containsCycle(vector<vector<char>> &grid) {  
24  }
```

The test case section shows a grid of characters:

```
grid =  
[[ "a","a","a","a"],["a","b","b","a"],["a","b","b","a"],["a","a","a","a"]]
```

Q14. <https://leetcode.com/problems/kth-largest-sum-in-a-binary-tree/submissions/1571461320/>

The screenshot shows the LeetCode submission interface for the problem "kth-largest-sum-in-a-binary-tree". The left sidebar displays the submission status as "Accepted" with 51/51 testcases passed. The runtime is 38 ms, beating 48.14% of submissions. The memory usage is 266.95 MB, beating 33.60%. A runtime graph shows the performance of the solution. The main area displays the C++ code, which uses a BFS approach to calculate the sum of nodes at each level and find the kth largest sum. The code is as follows:

```
C++  
13  public:  
14      long kthLargestLevelSum(TreeNode* root, int k) {  
15          vector<long long> res; // To store sum of each level  
16          queue<TreeNode*> q; // Queue for level-order traversal (BFS)  
17          q.push(root); // Start BFS from the root node  
18          while (!q.empty()) {  
19              int n = q.size(); // Number of nodes at the current level  
20              long long sum = 0; // Sum of node values at the current level  
21              // Process all nodes at the current level  
22              while (n--) {  
23  }
```

The test case section shows a binary tree structure with root 5 and children 8 and 9:

```
root =  
[5, 8, 9, 2, 1, 3, 7, 4, 6]
```

A diagram of the binary tree is also shown:

```
graph TD  
    5((5)) --- 8((8))  
    5 --- 9((9))  
    8 --- 2((2))  
    8 --- 1((1))  
    9 --- 3((3))  
    9 --- 7((7))  
    2 --- 4((4))  
    2 --- 6((6))
```