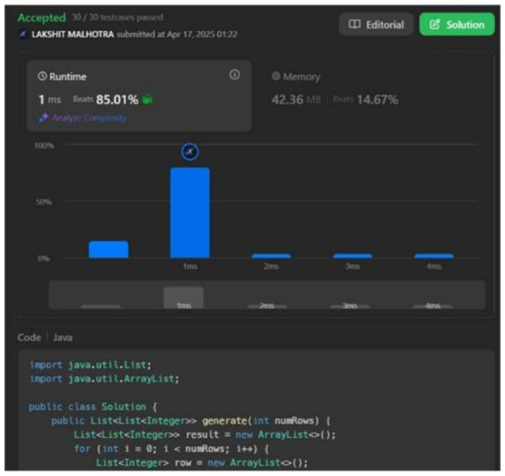
EXPERIMENT 10

STUDENT NAME: Akash Pande **UID:** 22BCS11135
BRANCH: CSE **SECTION:** 22BCS_FL_IOT_601A
SEMESTER: 6 **DATE OF PERFORMANCE:** 17/4/25
SUBJECT NAME: AP LAB -2 **SUBJECT CODE:** 22CSP-351

118. Pascal's Triangle
https://leetcode.com/problems/pascals-triangle/

```
import java.util.List;
import java.util.ArrayList;
public class Solution {
    public List<List<Integer>> generate(int numRows) {
        List<List<Integer>> result = new ArrayList<>();
        for (int i = 0; i < numRows; i++) {
            List<Integer> row = new ArrayList<>();
            row.add(1);
            for (int j = 1; j < i; j++) {
                row.add(result.get(i - 1).get(j - 1) + result.get(i -
1).get(j));
            }
            if (i > 0) {
                row.add(1);
            }
            result.add(row);
        }
        return result;
    }
}
```



Accepted 30 / 30 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:22
Runtime 1 ms Beats 85.01%
Memory 42.36 MB Beats 14.67%
Code | Java

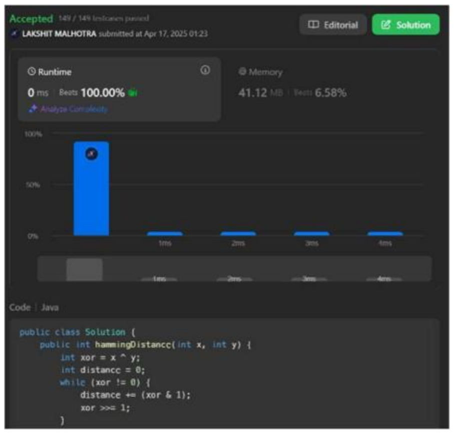
```
import java.util.List;
import java.util.ArrayList;

public class Solution {
    public List<List<Integer>> generate(int numRows) {
        List<List<Integer>> result = new ArrayList<>();
        for (int i = 0; i < numRows; i++) {
            List<Integer> row = new ArrayList<>();
```

461. Hamming Distance
https://leetcode.com/problems/hamming-distance/

```
public class Solution {
    public int hammingDistance(int x, int y) {
        int xor = x ^ y;
        int distance = 0;
```

```
 while (xor != 0) {
            distance += (xor & 1);
            xor >>= 1;
        }
        return distance;
    }
}
```



Accepted 149 / 149 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:23

Runtime 0 ms | Beats 100.00%
Memory 41.12 MB | Beats 6.58%

Code | Java

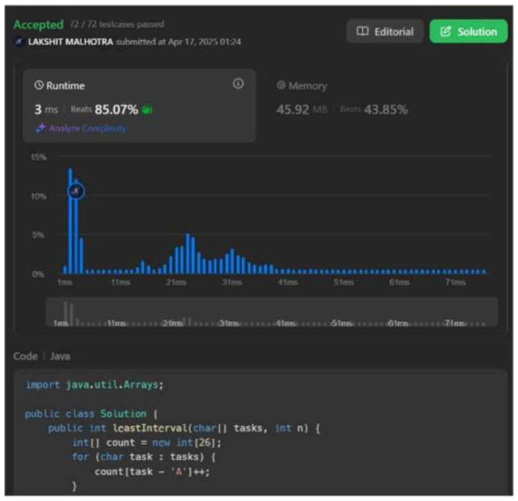
```
public class Solution {
    public int hammingDistance(int x, int y) {
        int xor = x ^ y;
        int distance = 0;
        while (xor != 0) {
            distance += (xor & 1);
            xor >>= 1;
        }
```

621. Task Scheduler

https://leetcode.com/problems/task-scheduler/

```
import java.util.Arrays;
public class Solution {
    public int leastInterval(char[] tasks, int n) {
        int[] count = new int[26];
        for (char task : tasks) {
            count[task - 'A']++;
        }
        Arrays.sort(count);
        int maxCount = count[25] - 1;
        int idleSlots = maxCount * n;

        for (int i = 24; i >= 0 && count[i] > 0; i--) {
            idleSlots -= Math.min(count[i], maxCount);
        }
        return idleSlots > 0 ? tasks.length + idleSlots : tasks.length;
    }
}
```



Accepted 72 / 72 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:24

Runtime 3 ms | Beats 85.07%
Memory 45.92 MB | Beats 43.85%

Code | Java

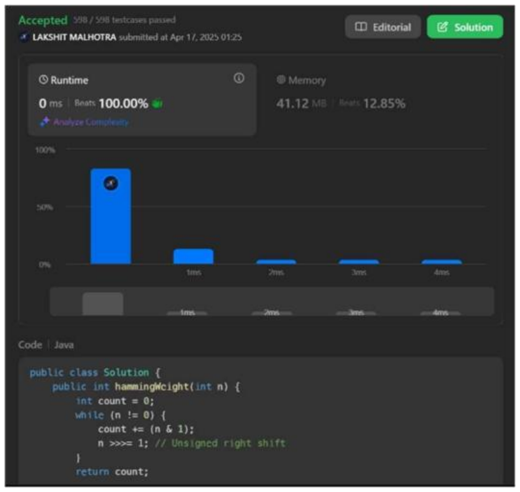
```
import java.util.Arrays;

public class Solution {
    public int leastInterval(char[] tasks, int n) {
        int[] count = new int[26];
        for (char task : tasks) {
            count[task - 'A']++;
        }
```

191. Number of 1 Bits

https://leetcode.com/problems/number-of-1-bits/

```
public class Solution {
    public int hammingWeight(int n) {
        int count = 0;
        while (n != 0) {
            count += (n & 1);
            n >>>= 1;
```



Accepted 598 / 598 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:25

Runtime 0 ms | Beats 100.00%
Memory 41.12 MB | Beats 12.85%

Code | Java

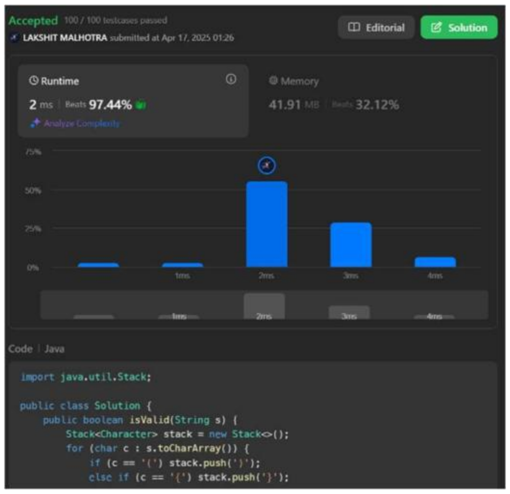
```
public class Solution {
    public int hammingWeight(int n) {
        int count = 0;
        while (n != 0) {
            count += (n & 1);
            n >>>= 1; // Unsigned right shift
        }
        return count;
```

```
  }
        return count;
    }
}
```

20. Valid Parentheses

https://leetcode.com/problems/valid-parentheses/

```
import java.util.Stack;
public class Solution {
    public boolean isValid(String s) {
        Stack<Character> stack = new Stack<>();
        for (char c : s.toCharArray()) {
            if (c == '(') stack.push(')');
            else if (c == '{') stack.push('}');
            else if (c == '[') stack.push(']');
            else if (stack.isEmpty() || stack.pop() != c) return
false;
        }
        return stack.isEmpty();
    }
}
```



Accepted 100 / 100 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:26

Runtime
2 ms | Beats 97.44%

Memory
41.91 MB | Beats 32.12%

Code | Java

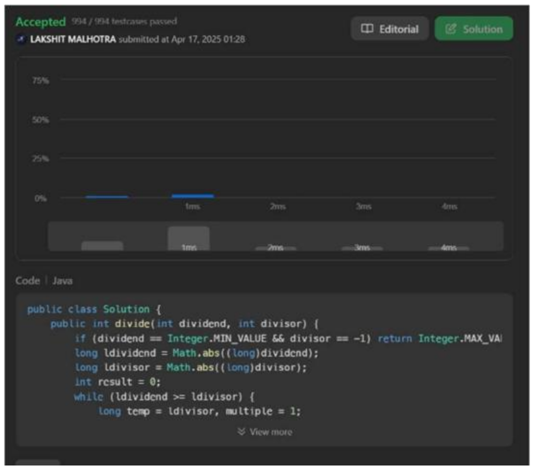
```
import java.util.Stack;

public class Solution {
    public boolean isValid(String s) {
        Stack<Character> stack = new Stack<>();
        for (char c : s.toCharArray()) {
            if (c == '(') stack.push(')');
            else if (c == '{') stack.push('}');
```

29. Divide Two Integers

https://leetcode.com/problems/divide-two-integers/

```
public class Solution {
    public int divide(int dividend, int divisor) {
        if (dividend == Integer.MIN_VALUE && divisor
== -1) return Integer.MAX_VALUE;
        long ldividend = Math.abs((long) dividend);
        long ldivisor = Math.abs((long) divisor);
        int result = 0;
        while (ldividend >= ldivisor) {
            long temp = ldivisor, multiple = 1;
            while (ldividend >= (temp << 1)) {
                temp <<= 1;
                multiple <<= 1;
```



Accepted 994 / 994 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:28

Code | Java

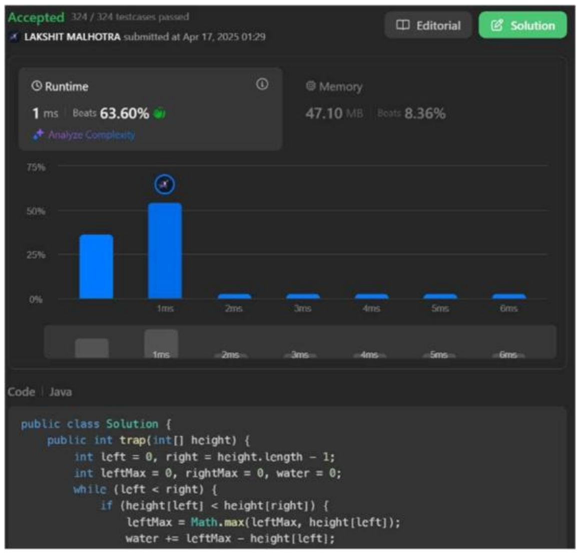
```
public class Solution {
    public int divide(int dividend, int divisor) {
        if (dividend == Integer.MIN_VALUE && divisor == -1) return Integer.MAX_VA
        long ldividend = Math.abs((long)dividend);
        long ldivisor = Math.abs((long)divisor);
        int result = 0;
        while (ldividend >= ldivisor) {
            long temp = ldivisor, multiple = 1;
```

```
  }
            ldividend -= temp;
            result += multiple;
  }
        return ((dividend > 0) == (divisor > 0)) ? result : -result;
    }
}
```

42. Trapping Rain Water

https://leetcode.com/problems/trapping-rain-water/

```
public class Solution {
    public int trap(int[] height) {
        int left = 0, right = height.length - 1;
        int leftMax = 0, rightMax = 0, water = 0;
        while (left < right) {
            if (height[left] < height[right]) {
                leftMax = Math.max(leftMax,
height[left]);
                water += leftMax - height[left];
                left++;
            } else {
                rightMax = Math.max(rightMax,
height[right]);
                water += rightMax - height[right];
                right--;
            }
        }
        return water;
    }
}
```



Accepted 324 / 324 testcases passed

LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:29

Runtime 1 ms | Beats 63.60%

Memory 47.10 MB | Beats 8.36%

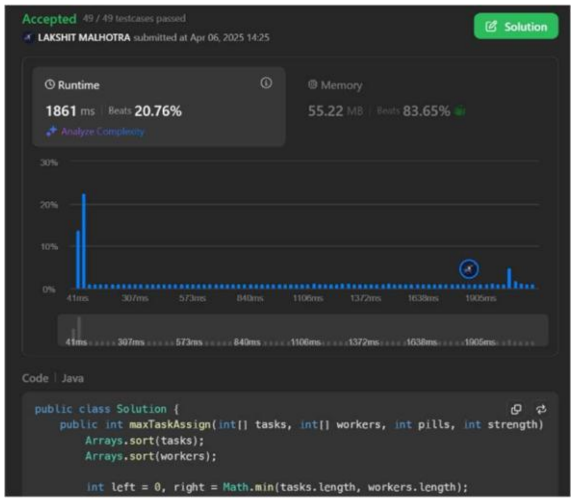
Code | Java

```
public class Solution {
    public int trap(int[] height) {
        int left = 0, right = height.length - 1;
        int leftMax = 0, rightMax = 0, water = 0;
        while (left < right) {
            if (height[left] < height[right]) {
                leftMax = Math.max(leftMax, height[left]);
                water += leftMax - height[left];
```

2071. Maximum Number of Tasks You Can Assign

https://leetcode.com/problems/maximum-number-of-tasks-you-can-assign/

```
import java.util.Arrays;
import java.util.PriorityQueue;
public class Solution {
    public int maxTaskAssign(int[] tasks, int[] workers, int pills, int strength) {
        Arrays.sort(tasks);
    Arrays.sort(workers);
        int left = 0, right = Math.min(tasks.length, workers.length), res = 0;
        while (left <= right) {
            int mid = (left + right) / 2;
            if (canAssign(mid, tasks, workers, pills, strength)) {
res = mid;
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return res;
    }
    private boolean canAssign(int k, int[] tasks, int[] workers, int pills, int strength) {
        PriorityQueue<Integer> pq = new PriorityQueue<>();
        int j = workers.length - k;
        for (int i = 0; i < k; i++) pq.add(tasks[i]);
        int remainingPills = pills;
        for (int i = j; i < workers.length; i++) {
            if (!pq.isEmpty() && workers[i] >= pq.peek()) {
                pq.poll();
            } else if (remainingPills > 0) {
                Integer task = null;
                for (int t : pq) {
                    if (workers[i] + strength >= t) {
                        task = t;
                        break;
                    }
                }
                if (task != null) {
                    pq.remove(task);
                    remainingPills--;
                } else {
                    return false;
                }
            } else return false;
        }
        return pq.isEmpty();
    }
}
```


Accepted 49 / 49 testcases passed
LAKSHIT MALHOTRA submitted at Apr 06, 2025 14:25

Runtime
1861 ms | Beats 20.76%

Memory
55.22 MB | Beats 83.65%

Code | Java

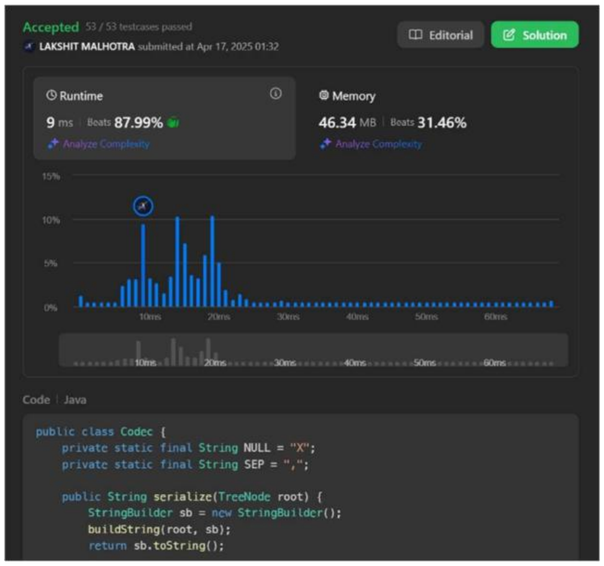
```
public class Solution {
    public int maxTaskAssign(int[] tasks, int[] workers, int pills, int strength)
        Arrays.sort(tasks);
        Arrays.sort(workers);

        int left = 0, right = Math.min(tasks.length, workers.length);
```

297. Serialize and Deserialize Binary Tree

https://leetcode.com/problems/serialize-and-deserialize-binary-tree/

```
public class Codec {
   private static final String NULL = "X";
   private static final String SEP = ",";
   public String serialize(TreeNode root) {
      StringBuilder sb = new StringBuilder();
      buildString(root, sb);
      return sb.toString();
   }
   private void buildString(TreeNode node, StringBuilder sb) {
      if (node == null) {
         sb.append(NULL).append(SEP);
         return;
      }
      sb.append(node.val).append(SEP);
      buildString(node.left, sb);
      buildString(node.right, sb);
   }
   public TreeNode deserialize(String data) {
      LinkedList<String> nodes = new LinkedList<>(Arrays.asList(data.split(SEP)));
      return buildTree(nodes);
   }
   private TreeNode
buildTree(LinkedList<String> nodes) {
      String val = nodes.removeFirst();
      if (val.equals(NULL)) return null;
      TreeNode node = new
TreeNode(Integer.parseInt(val));
      node.left = buildTree(nodes);
      node.right = buildTree(nodes);
      return node;
   }
}
```



Accepted 53 / 53 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:32

Runtime
9 ms | Beats 87.99%

Memory
46.34 MB | Beats 31.46%

Code | Java

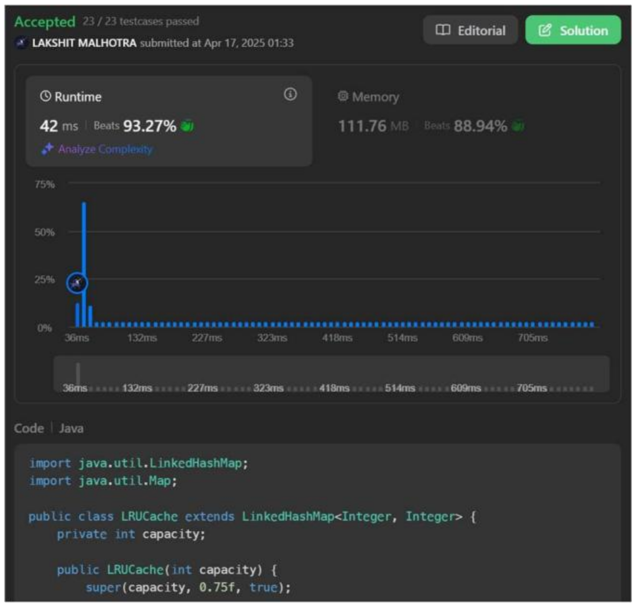
```
public class Codec {
    private static final String NULL = "X";
    private static final String SEP = ",";

    public String serialize(TreeNode root) {
        StringBuilder sb = new StringBuilder();
        buildString(root, sb);
        return sb.toString();
```

146. LRU Cache

https://leetcode.com/problems/lru-cache/

```
import java.util.LinkedHashMap;
import java.util.Map;
public class LRUCache extends LinkedHashMap<Integer, Integer> {
    private int capacity;
    public LRUCache(int capacity) {
        super(capacity, 0.75f, true);
        this.capacity = capacity;
    }
    public int get(int key) {
        return super.getOrDefault(key, -1);
    }
    public void put(int key, int value) {
        super.put(key, value);
    }
    @Override
    protected boolean
removeEldestEntry(Map.Entry<Integer,
Integer> eldest) {
        return size() > capacity;
    }
}
```


Accepted 23 / 23 testcases passed
LAKSHIT MALHOTRA submitted at Apr 17, 2025 01:33

Runtime
42 ms | Beats 93.27%

Memory
111.76 MB | Beats 88.94%

Code | Java

```
import java.util.LinkedHashMap;
import java.util.Map;

public class LRUCache extends LinkedHashMap<Integer, Integer> {
    private int capacity;

    public LRUCache(int capacity) {
        super(capacity, 0.75f, true);
```