

DSA- Assignment-3

Questions List :-

1. Question was- <https://leetcode.com/problems/climbing-stairs/description/>

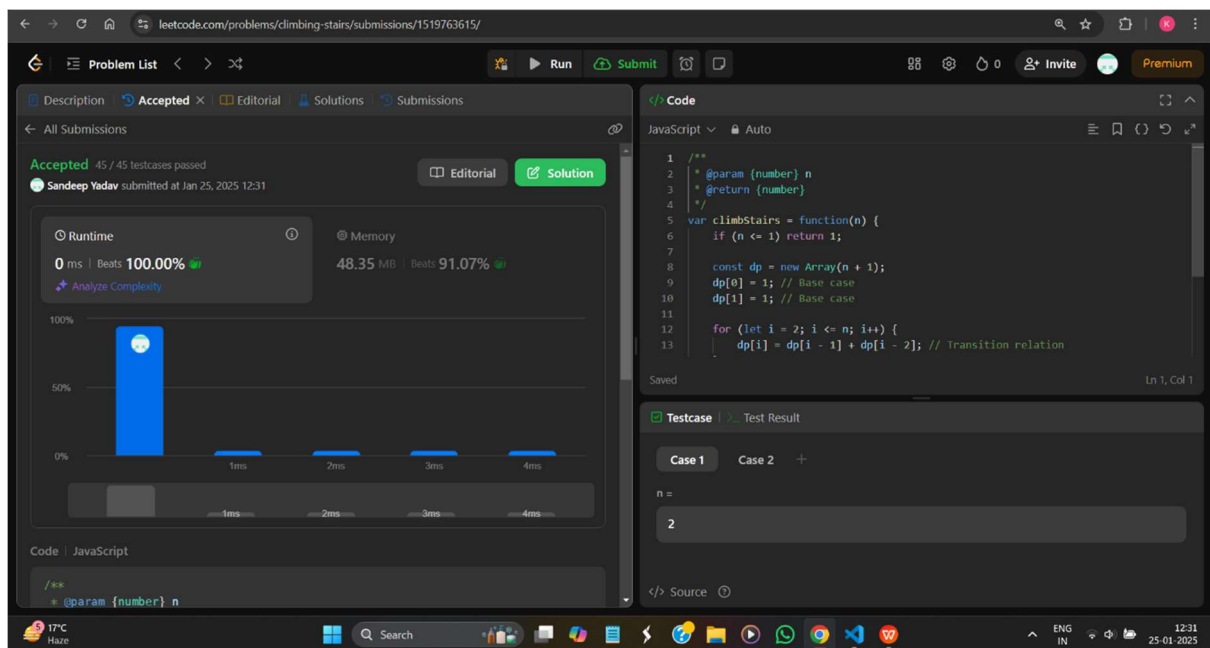
Solution Link – <https://leetcode.com/problems/climbing-stairs/submissions/1519763615/>

Description:

- (i) If $n \leq 1$ return 1 because only one way to climb.
- (ii) Create new Array length of $n + 1$.
- (iii) Define base cases $dp[0]$ & $dp[1] = 1$ because in both cases only one way to climb.
- (iv) Iterate 2 to n and find the way for $i - 1$ and $i - 2$.

Time Complexity: $O(n)$

Space Complexity: $O(n)$



2. Question was- <https://leetcode.com/problems/merge-two-sorted-lists/description/>

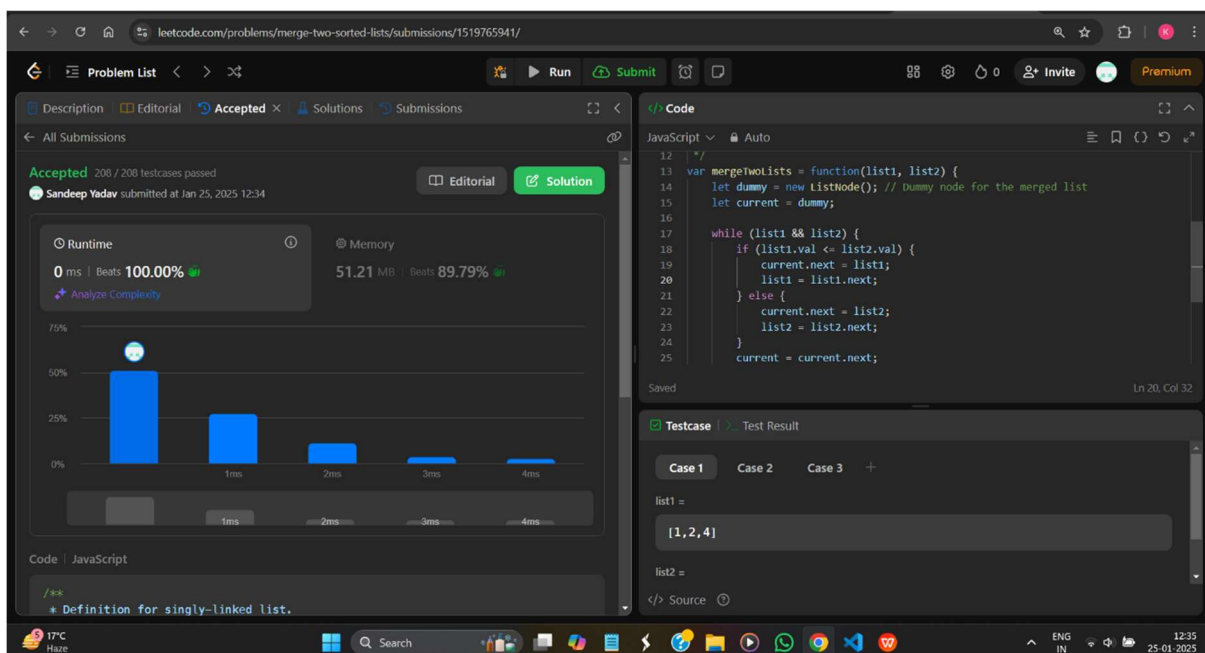
Solution Link- <https://leetcode.com/problems/merge-two-sorted-lists/submissions/1519765941/>

Description:

- (i) Create new dummy node for merge List.
- (ii) Create new current node and assign dummy in current.
- (iii) Compare values between list1 and list2 and take the small value in current and move current.next.
- (iv) And also move the pointer on current.next.
- (v) Attach Attach the remaining nodes, if any.
- (vi) Return dummy.next.

Time Complexity: $O(n+m)$ // (for sorting array)

Space Complexity: $O(1)$ (Not use any extra space,in-place merge)



3. Question was- <https://leetcode.com/problems/palindrome-linked-list/description/>

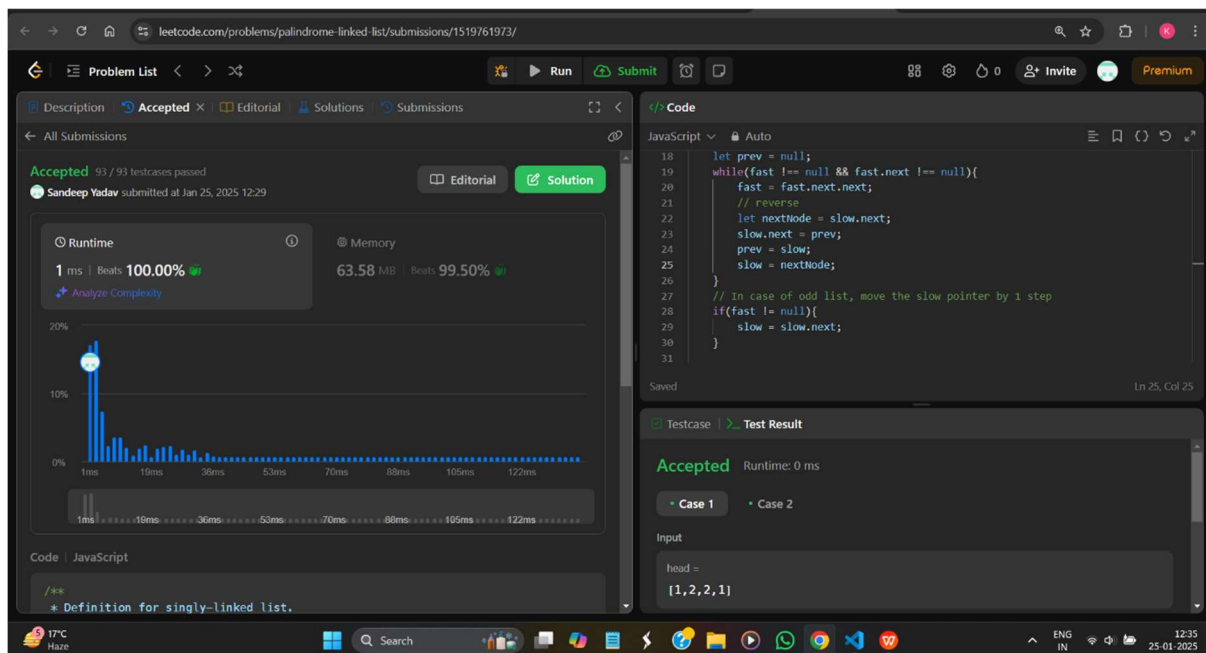
Solution Link- <https://leetcode.com/problems/palindrome-linked-list/submissions/1519761973/>

Description:

- (i) Reverse the half List.
- (ii) Compare 1st Half from Second Half.
- (iii) If all node values are matched return true .
- (iv) If all node values are not matched then return false.

Time Complexity: $O(n)$ n is length of List.

Space Complexity: $O(1)$ no use extra space



4. Question was- <https://leetcode.com/problems/linked-list-cycle/>

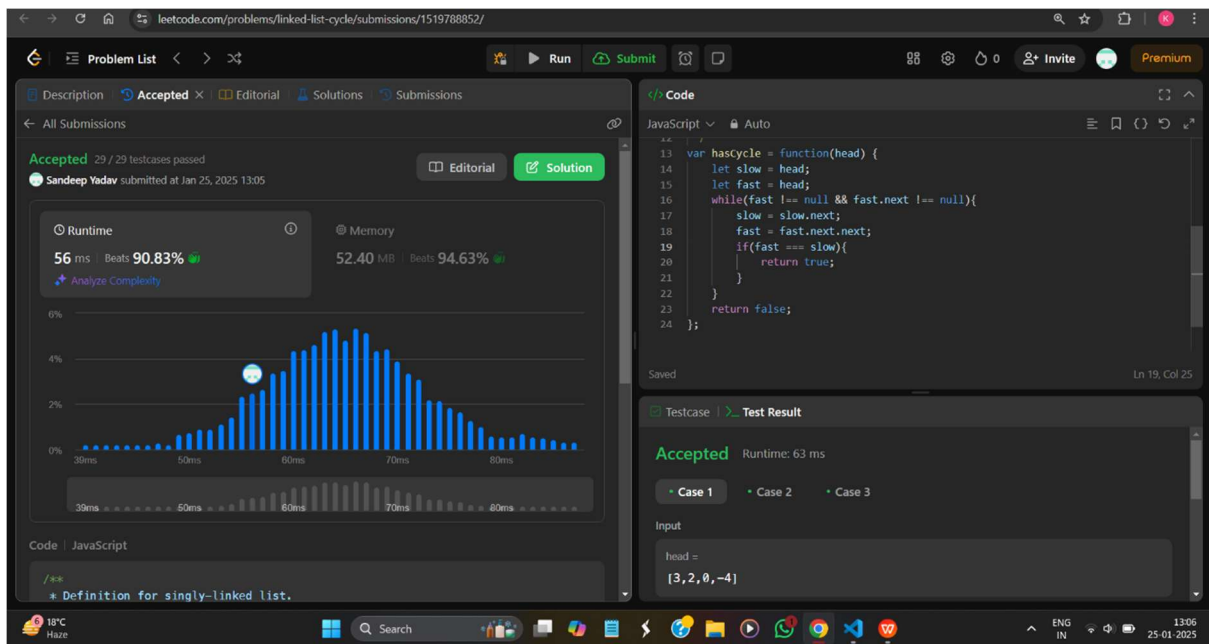
Solution Link– <https://leetcode.com/problems/linked-list-cycle/submissions/1519788852/>

Description:

- (i) Create slow and fast pointer.
- (ii) Traverse all the nodes till the fast & fast.next not equal to null.
- (iii) Slow move step and fast move 2 step at a time.
- (iv) If slow === fast then return true.
- (v) If slow !== fast means, cycle not present return false.

Time Complexity: $O(n + k)$ n is the numbers of nodes and k is the cycle length

Space Complexity: $O(1)$ no extra space



5. Question was- <https://leetcode.com/problems/remove-nth-node-from-end-of-list/description/>

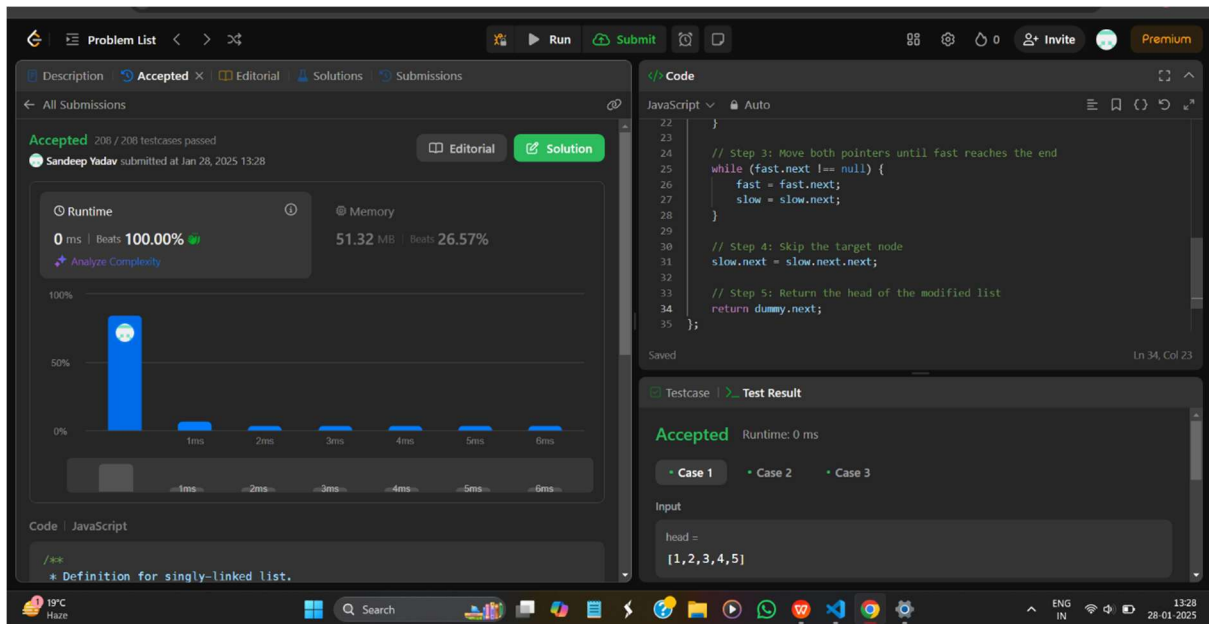
Solution Link- <https://leetcode.com/problems/remove-nth-node-from-end-of-list/submissions/1523019444/>

Description:

- (i) Create a dummy node pointing to head (for handle edge cases easily).
- (ii) Move the fast pointer n steps ahead.
- (iii) Move both pointers until fast reaches the end.
- (iv) Skip the target node.
- (v) Return head of the modified List.

Time Complexity: $O(L)$, single traversal using two pointers. // L is length of List

Space Complexity: $O(1)$ no use extra space



6. Question was- <https://leetcode.com/problems/powx-n/description/>

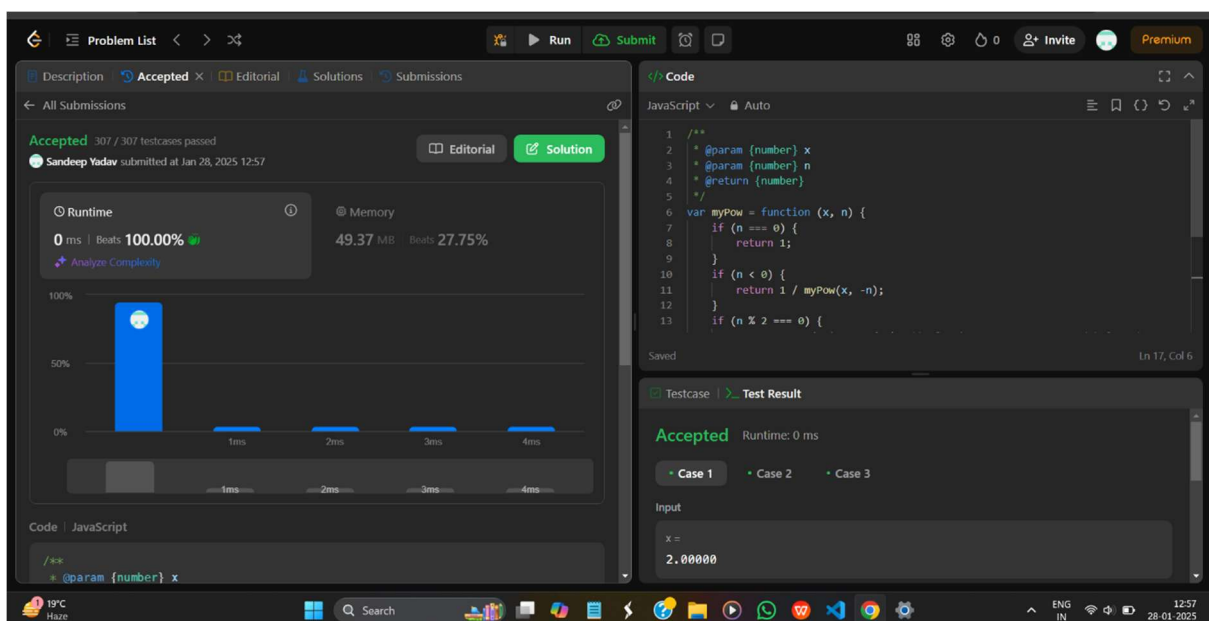
Solution Link- <https://leetcode.com/problems/powx-n/submissions/1522997596/>

Description:

- (i) If $n == 0$ return 1 because any number who have 0 pow = 1 .
- (ii) If $n < 0$ means pow is negative then return $1/\text{result}$.
- (iii) If n is odd number call $x * \text{myPow}(x * x, (n - 1) / 2)$.
- (iv) If n is even call $\text{myPow}(x * x, (n - 1) / 2)$.

Time Complexity: $O(n)$

Space Complexity: $O(1)$ no use extra space



7. Question was- <https://leetcode.com/problems/delete-node-in-a-linked-list/description/>

Solution Link- <https://leetcode.com/problems/delete-node-in-a-linked-list/submissions/1522973045/>

Description:

- (i) Copy the value of next node to the current node.
- (ii) Skip the next node by pointing to next.next node.

Time Complexity: $O(1)$ // Not Traversing entire List

Space Complexity: $O(1)$ no use extra space

The screenshot displays the LeetCode submission page for the problem "Delete Node in a Linked List". The interface is divided into several sections:

- Problem List:** Shows the problem name and a search bar.
- Description:** Contains the problem statement and a link to the solution.
- Accepted:** Indicates that the solution is accepted, with 41/41 test cases passed.
- Runtime:** Shows a runtime of 0 ms, which is 100.00% faster than other solutions.
- Memory:** Shows a memory usage of 43.65 MB, which is 96.03% better than other solutions.
- Code:** Displays the Java code for the solution, which implements the deletion of a node by copying the next node's value and skipping the next node.
- Test Result:** Shows the test results for the solution, indicating that it is accepted.

```
class Solution {
    public void deleteNode(ListNode node) {
        // Copy the value of the next node to the current node
        node.val = node.next.val;
        // Skip the next node by pointing to the next-next node
        node.next = node.next.next;
    }
}
```

Key Features :-

- Solutions are less time taking.

Difficulties :-

- Facing Difficulties in Code Optimization.
- Try to make code efficient and use better approach to solve them.

GitHub Repository Link : -

<https://github.com/SandyBhai03/Internshala-Assignments/blob/main/Assignment-Course5/DSA-2/Assignment-3/app.js>