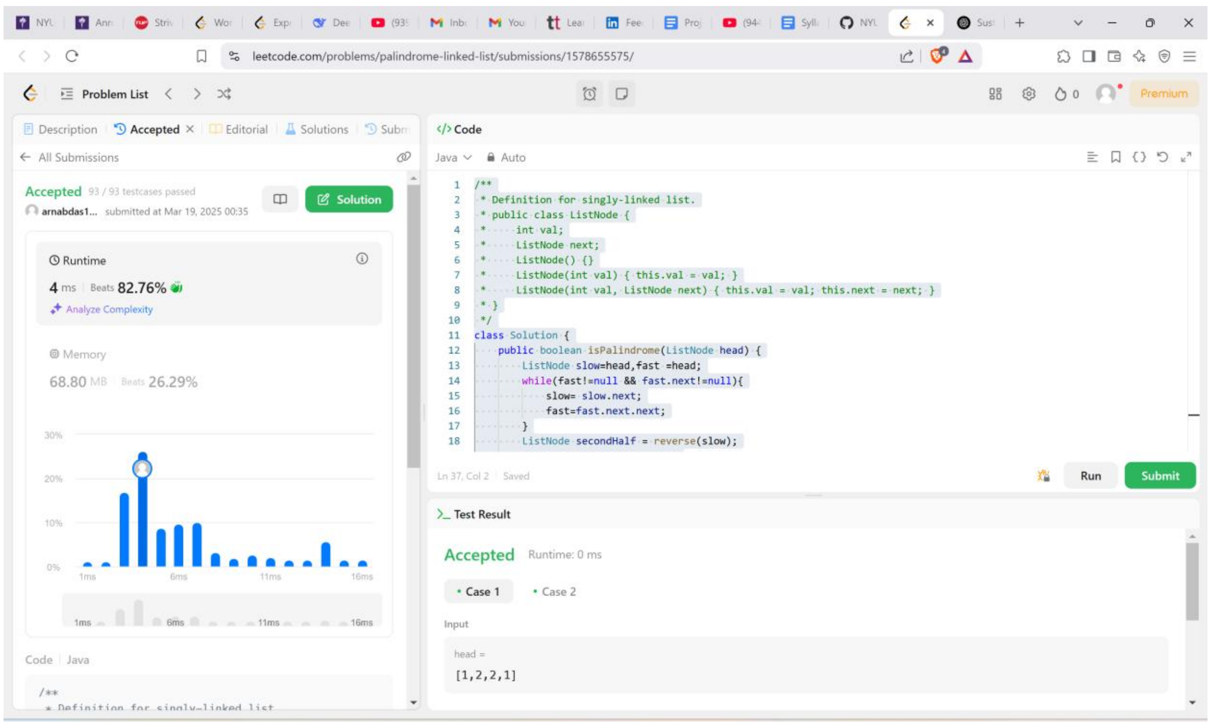
# Week 3 Assignment

```
Problem -1
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public boolean isPalindrome(ListNode head) {
        ListNode slow=head,fast =head;
        while(fast!=null && fast.next!=null){
            slow= slow.next;
            fast=fast.next.next;
        }
        ListNode secondHalf = reverse(slow);
        ListNode firstHalf = head;
        while(secondHalf!=null){
            if(firstHalf.val !=secondHalf.val) return false;
            firstHalf = firstHalf.next;
            secondHalf = secondHalf.next;
        }
        return true;
    }
    public ListNode reverse(ListNode head){
        ListNode curr = head,prev = null;
        while(curr != null){
            ListNode nextNode = curr.next;
            curr.next = prev;
            prev = curr;
            curr = nextNode;
        }
        return prev;
    }
}
```


leetcode.com/problems/palindrome-linked-list/submissions/1578655575/

Accepted 93 / 93 testcases passed
arnabdas1... submitted at Mar 19, 2025 00:35

Runtime
4 ms | Beats 82.76%

Memory
68.80 MB | Beats 26.29%

```java
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public boolean isPalindrome(ListNode head) {
        ListNode slow=head,fast =head;
        while(fast!=null && fast.next!=null){
            slow= slow.next;
            fast=fast.next.next;
        }
        ListNode secondHalf = reverse(slow);
```

Ln 37, Col 2 | Saved

Test Result

Accepted Runtime: 0 ms

Case 1 · Case 2

Input

head =
[1,2,2,1]

Problem 2

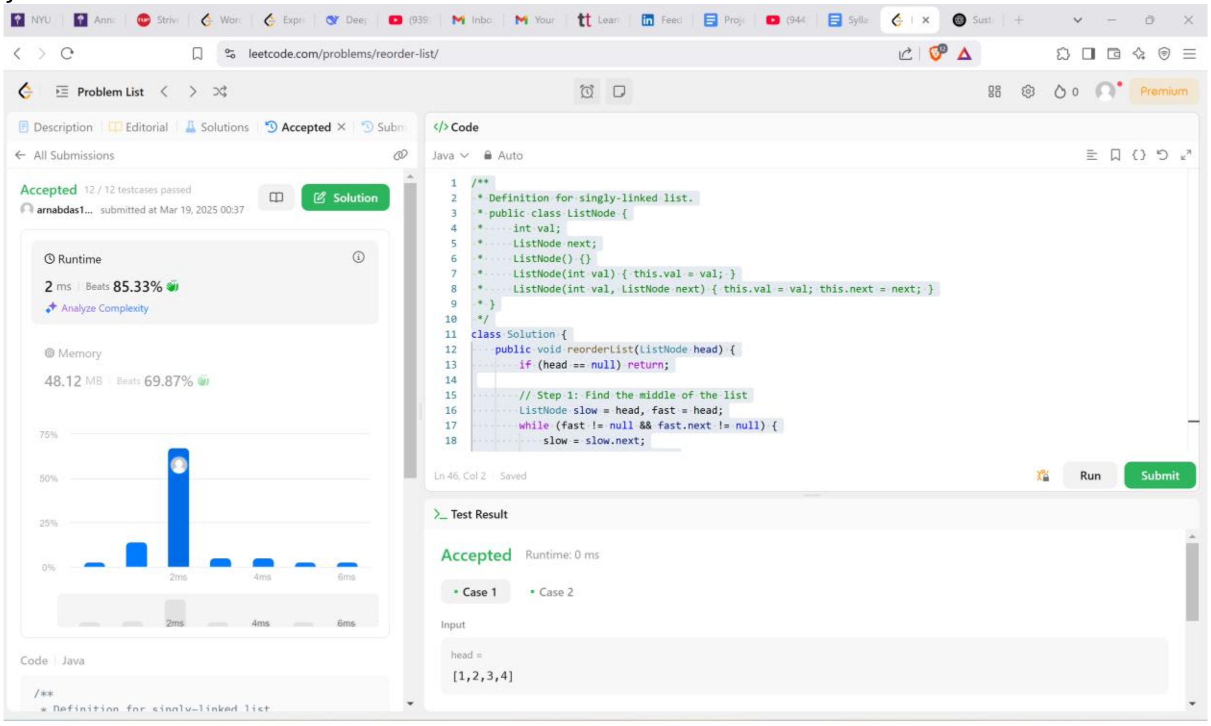
```java
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public void reorderList(ListNode head) {
        if (head == null) return;

        // Step 1: Find the middle of the list
        ListNode slow = head, fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
            fast = fast.next.next;
        }

        // Step 2: Reverse the second half of the list
        ListNode second = slow.next;
        slow.next = null;
        ListNode node = null;
        while (second != null) {
            ListNode temp = second.next;
            second.next = node;
            node = second;
            second = temp;
        }

        // Step 3: Merge the two halves
        ListNode first = head;
        second = node;

        while (second != null) {
            ListNode temp1 = first.next, temp2 = second.next;
            first.next = second;
            second.next = temp1;
            first = temp1;
            second = temp2;
        }
    }
}
```



leetcode.com/problems/reorder-list/

Accepted 12 / 12 testcases passed

arnabdas1... submitted at Mar 19, 2025 00:37

Runtime: 2 ms | Beats 85.33%

Memory: 48.12 MB | Beats 69.87%

Code | Java

```
/**
 * Definition for singly-linked list.
 * public class ListNode {
 *     int val;
 *     ListNode next;
 *     ListNode() {}
 *     ListNode(int val) { this.val = val; }
 *     ListNode(int val, ListNode next) { this.val = val; this.next = next; }
 * }
 */
class Solution {
    public void reorderList(ListNode head) {
        if (head == null) return;

        // Step 1: Find the middle of the list
        ListNode slow = head, fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
```

Ln 46, Col 2 Saved

Test Result

Accepted Runtime: 0 ms

Case 1 · Case 2

Input

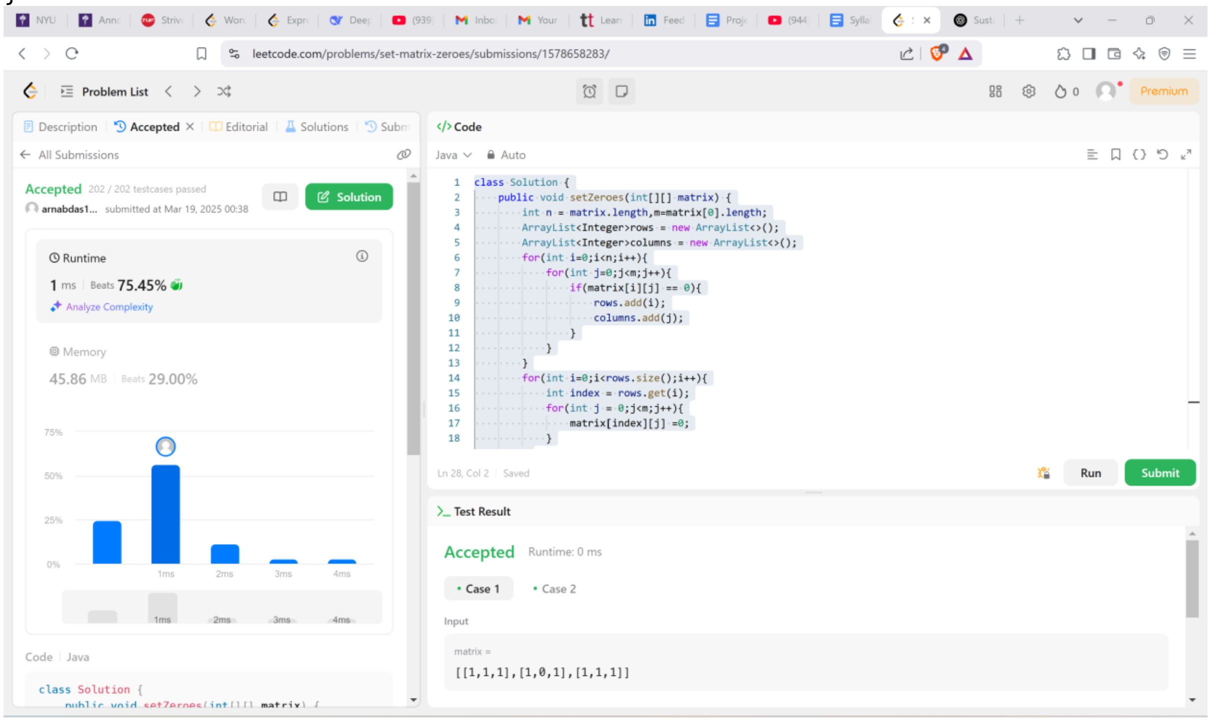
head =
[1,2,3,4]

Problem 3

```
class Solution {

    public void setZeroes(int[][] matrix) {
        int n = matrix.length,m=matrix[0].length;
        ArrayList<Integer>rows = new ArrayList<>();
        ArrayList<Integer>columns = new ArrayList<>();
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(matrix[i][j] == 0){
                    rows.add(i);
                    columns.add(j);
                }
            }
        }
        for(int i=0;i<rows.size();i++){
            int index = rows.get(i);
            for(int j = 0;j<m;j++){
                matrix[index][j] =0;
            }
        }
        for(int i=0;i<columns.size();i++){
            int index = columns.get(i);
            for(int j = 0;j<n;j++){
                matrix[j][index] =0;
            }
        }

    }
}
```



leetcode.com/problems/set-matrix-zeroes/submissions/1578658283/

Accepted 202 / 202 testcases passed
arnabdas1... submitted at Mar 19, 2025 00:38

Runtime
1 ms | Beats 75.45%

Memory
45.86 MB | Beats 29.00%

Code | Java

```
class Solution {
    public void setZeroes(int[][] matrix) {
```

```
class Solution {
    public void setZeroes(int[][] matrix) {
        int n = matrix.length,m=matrix[0].length;
        ArrayList<Integer>rows = new ArrayList<>();
        ArrayList<Integer>columns = new ArrayList<>();
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(matrix[i][j] == 0){
                    rows.add(i);
                    columns.add(j);
                }
            }
        }
        for(int i=0;i<rows.size();i++){
            int index = rows.get(i);
            for(int j = 0;j<m;j++){
                matrix[index][j] =0;
            }
```

Ln 28, Col 2 | Saved

Accepted Runtime: 0 ms

matrix =
[[1,1,1],[1,0,1],[1,1,1]]