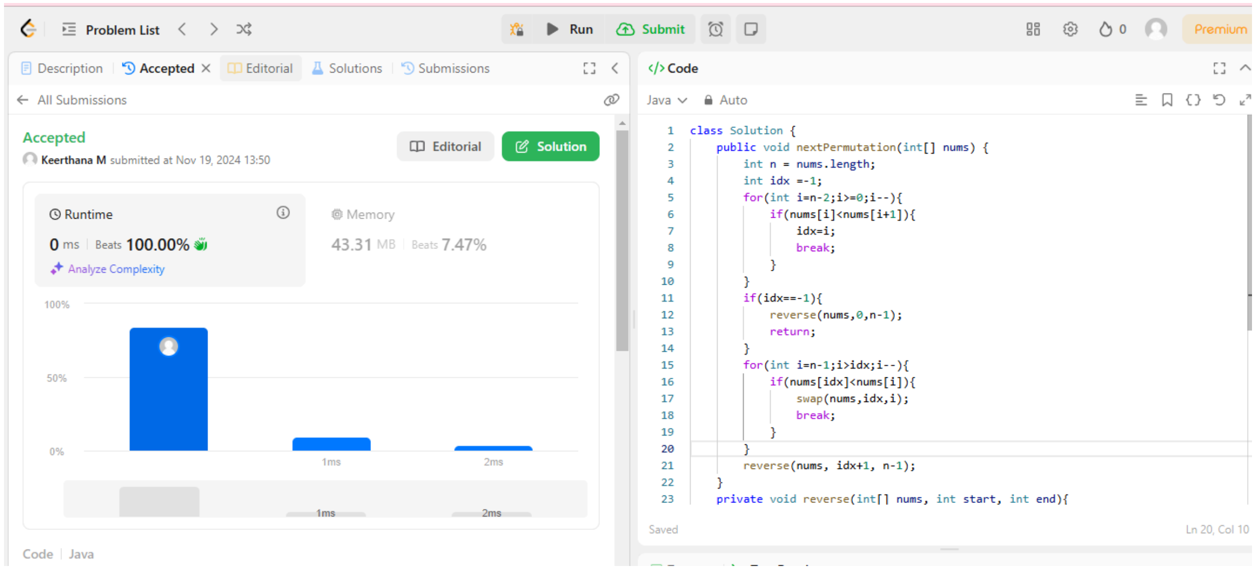
DSA Coding Practice-7

1.Next Permutation:

```
class Solution {
    public void nextPermutation(int[] nums) {
        int n = nums.length;
        int idx =-1;
        for(int i=n-2;i>=0;i--){
            if(nums[i]<nums[i+1]){
                idx=i;
                break;
            }
        }
        if(idx==-1){
            reverse(nums,0,n-1);
            return;
        }
        for(int i=n-1;i>idx;i--){
            if(nums[idx]<nums[i]){
                swap(nums,idx,i);
                break;
            }
        }
        reverse(nums, idx+1, n-1);
    }
    private void reverse(int[] nums, int start, int end){
        while(start<end){
            int temp = nums[start];
            nums[start] = nums[end];
            nums[end] = temp;
            start++;
            end--;
        }
    }
    private void swap(int[] nums, int idx, int i){
        int temp = nums[idx];
        nums[idx] = nums[i];
        nums[i] = temp;
    }
}
```



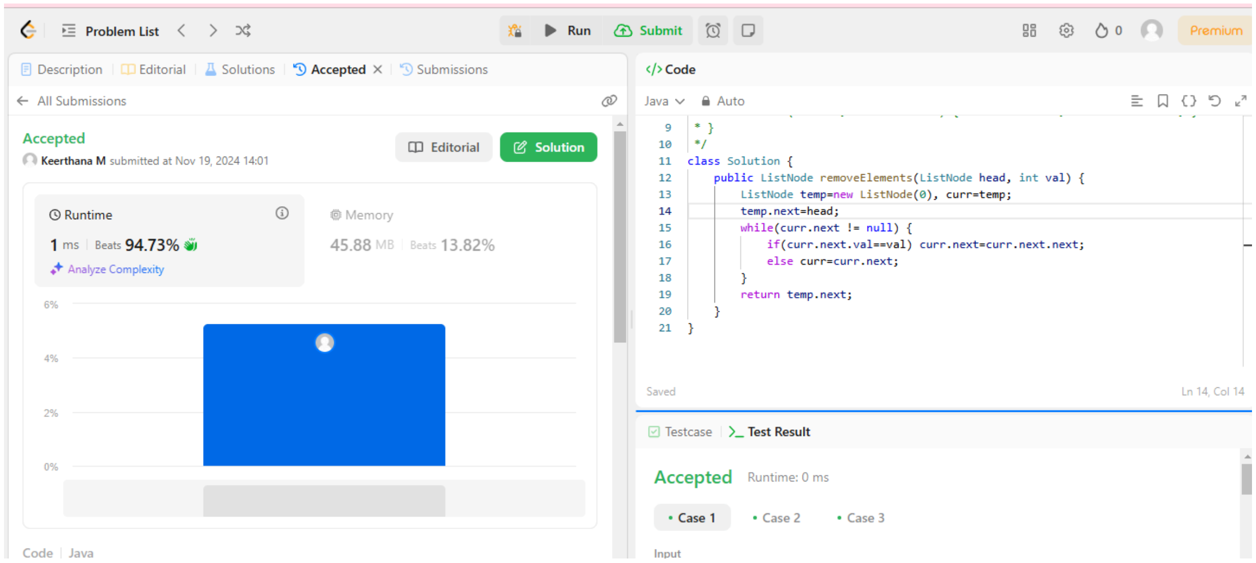
Accepted
Keerthana M submitted at Nov 19, 2024 13:50

Runtime 0 ms | Beats 100.00%
Memory 43.31 MB | Beats 7.47%

```java
class Solution {
    public void nextPermutation(int[] nums) {
        int n = nums.length;
        int idx =-1;
        for(int i=n-2;i>=0;i--){
            if(nums[i]<nums[i+1]){
                idx=i;
                break;
            }
        }
        if(idx==-1){
            reverse(nums,0,n-1);
            return;
        }
        for(int i=n-1;i>idx;i--){
            if(nums[idx]<nums[i]){
                swap(nums,idx,i);
                break;
            }
        }
        reverse(nums, idx+1, n-1);
    }
    private void reverse(int[] nums, int start, int end){
```

Saved   Ln 20, Col 10

2.Remove Linked List:



Accepted
Keerthana M submitted at Nov 19, 2024 14:01

Runtime 1 ms | Beats 94.73%
Memory 45.88 MB | Beats 13.82%

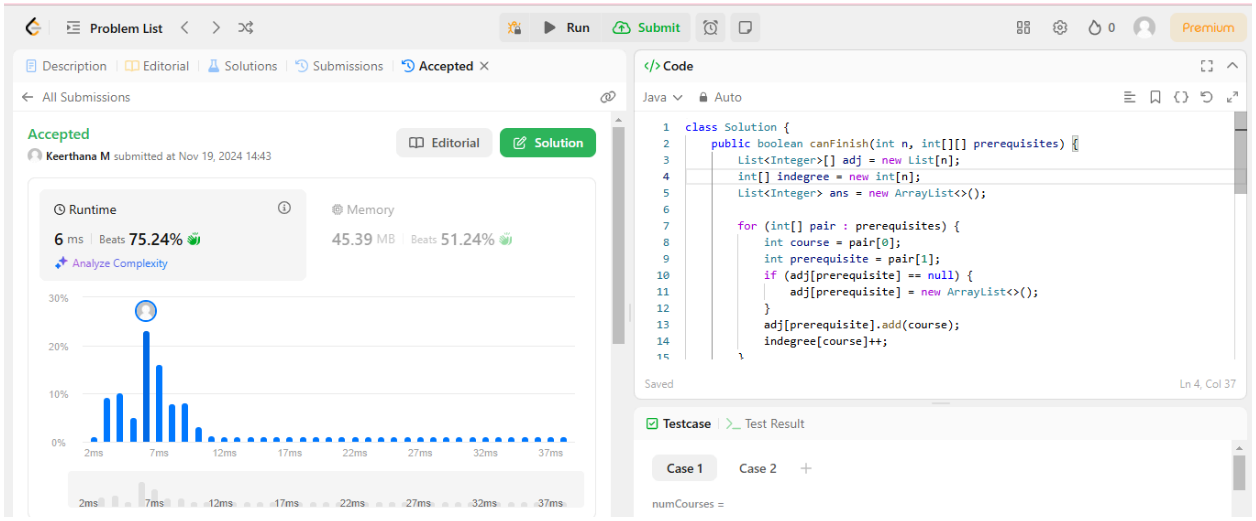
```java
 * }
 */
class Solution {
    public ListNode removeElements(ListNode head, int val) {
        ListNode temp=new ListNode(0), curr=temp;
        temp.next=head;
        while(curr.next != null) {
            if(curr.next.val==val) curr.next=curr.next.next;
            else curr=curr.next;
        }
        return temp.next;
    }
}
```

Saved   Ln 14, Col 14

Accepted Runtime: 0 ms
Case 1 · Case 2 · Case 3

3.Course Schedule:

```java
class Solution {
    public boolean canFinish(int n, int[][] prerequisites) {
        List<Integer>[] adj = new List[n];
        int[] indegree = new int[n];
        List<Integer> ans = new ArrayList<>();

        for (int[] pair : prerequisites) {
            int course = pair[0];
            int prerequisite = pair[1];
            if (adj[prerequisite] == null) {
                adj[prerequisite] = new ArrayList<>();
            }
            adj[prerequisite].add(course);
            indegree[course]++;
        }

        Queue<Integer> queue = new LinkedList<>();
        for (int i = 0; i < n; i++) {
            if (indegree[i] == 0) {
                queue.offer(i);
            }
        }

        while (!queue.isEmpty()) {
            int current = queue.poll();
            ans.add(current);

            if (adj[current] != null) {
                for (int next : adj[current]) {
                    indegree[next]--;
                    if (indegree[next] == 0) {
                        queue.offer(next);
                    }
                }
            }
        }

        return ans.size() == n;
    }
}
```



Problem List | Run | Submit | Premium

Description | Editorial | Solutions | Submissions | Accepted

All Submissions

Accepted

Keerthana M submitted at Nov 19, 2024 14:43

Editorial | Solution

Runtime
6 ms | Beats 75.24%
Analyze Complexity

Memory
45.39 MB | Beats 51.24%

Code
Java | Auto

```java
class Solution {
    public boolean canFinish(int n, int[][] prerequisites) {
        List<Integer>[] adj = new List[n];
        int[] indegree = new int[n];
        List<Integer> ans = new ArrayList<>();

        for (int[] pair : prerequisites) {
            int course = pair[0];
            int prerequisite = pair[1];
            if (adj[prerequisite] == null) {
                adj[prerequisite] = new ArrayList<>();
            }
            adj[prerequisite].add(course);
            indegree[course]++;
```

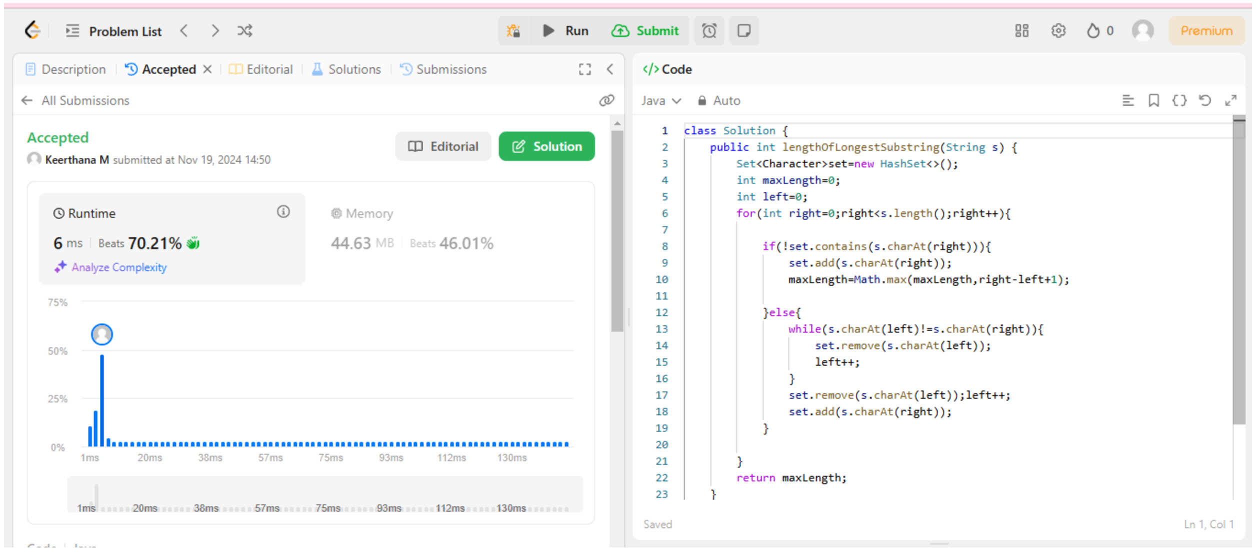
Saved | Ln 4, Col 37

Testcase | Test Result

Case 1 | Case 2

numCourses =

4.Longest Substring Without Repeating Character:



Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted

Keerthana M submitted at Nov 19, 2024 14:50

Editorial | Solution

Runtime
6 ms | Beats 70.21%
Analyze Complexity

Memory
44.63 MB | Beats 46.01%

Code
Java | Auto

```java
class Solution {
    public int lengthOfLongestSubstring(String s) {
        Set<Character>set=new HashSet<>();
        int maxLength=0;
        int left=0;
        for(int right=0;right<s.length();right++){

            if(!set.contains(s.charAt(right))){
                set.add(s.charAt(right));
                maxLength=Math.max(maxLength,right-left+1);

            }else{
                while(s.charAt(left)!=s.charAt(right)){
                    set.remove(s.charAt(left));
                    left++;
                }
                set.remove(s.charAt(left));left++;
                set.add(s.charAt(right));
            }

        }
        return maxLength;
    }
```

Saved | Ln 1, Col 1