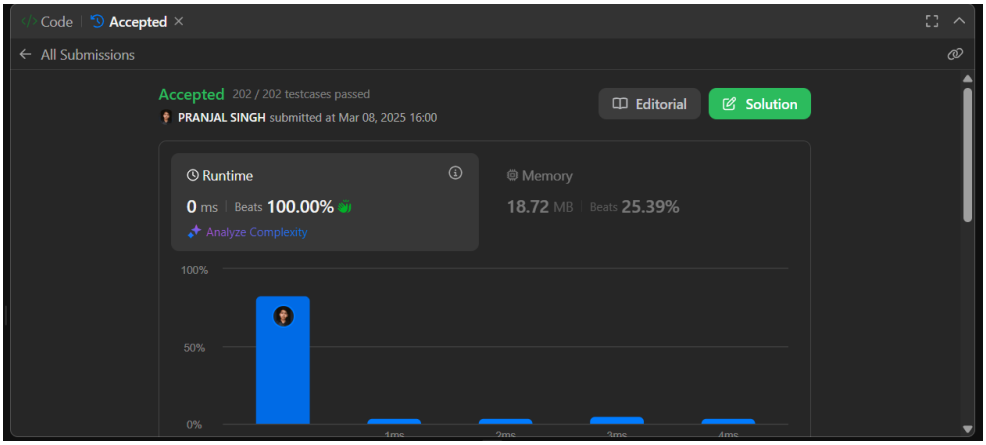
Name: Pranjal Singh **UID:** 22BCS13041
Sec: FL_IOT-601/ A **Sub:** AP Lab -II

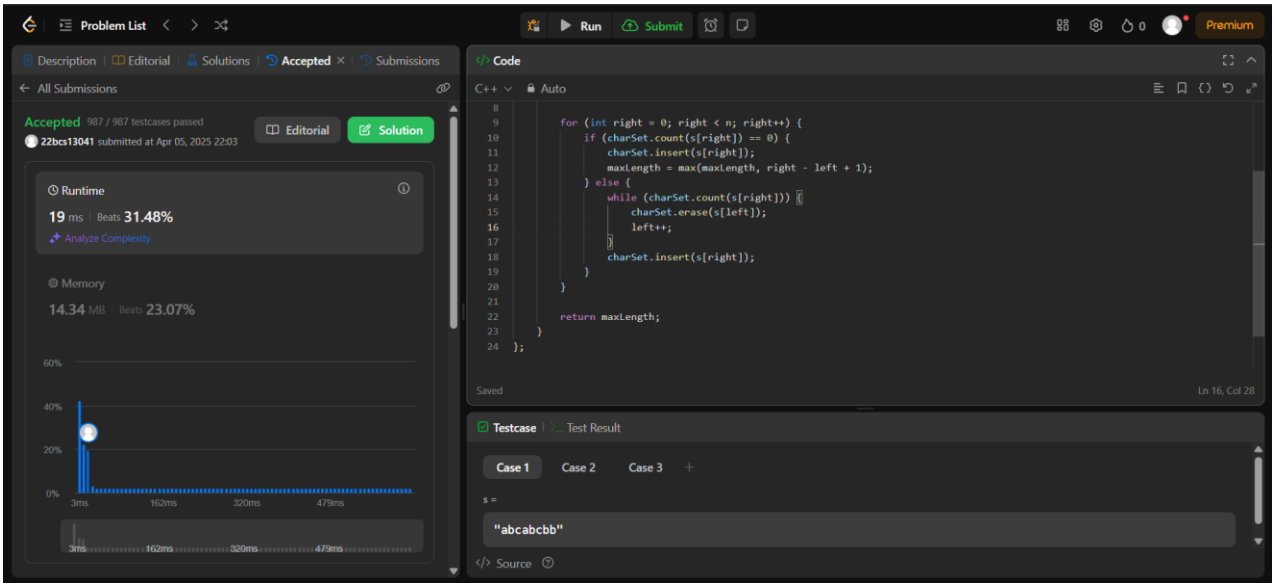
Set Zeros

```
class Solution {
public:
    void setZeroes(vector<vector<int>>& matrix) {
        int n=matrix.size();
        int m=matrix[0].size();
        unordered_set<int> setRows;
        unordered_set<int> setCols;
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(matrix[i][j]==0){
                    setRows.insert(i);
                    setCols.insert(j);
                }
            }
        }
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(setRows.count(i)>0||setCols.count(j)>0){
                    matrix[i][j]=0;
                }
            }
        }
    }
};
```



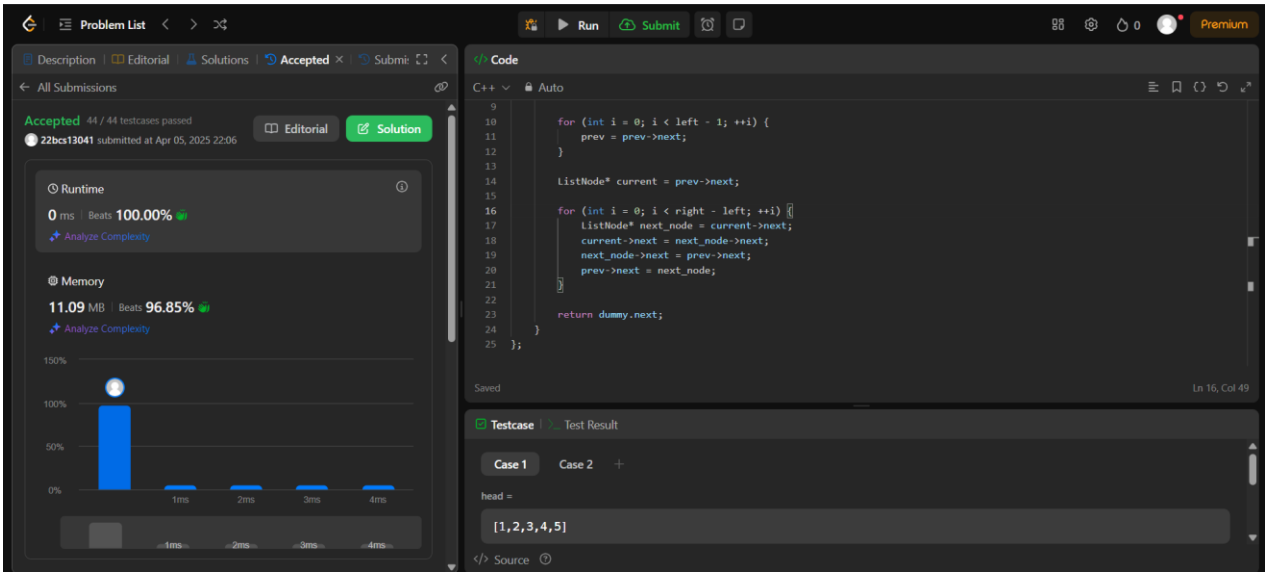
Length of Longest Substring

```
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        int n = s.length();
        int maxLength = 0;
        unordered_set<char> charSet;
        int left = 0;

        for (int right = 0; right < n; right++) {
            if (charSet.count(s[right]) == 0) {
                charSet.insert(s[right]);
                maxLength = max(maxLength, right - left + 1);
            } else {
                while (charSet.count(s[right])) {
                    charSet.erase(s[left]);
                    left++;
                }
                charSet.insert(s[right]);
            }
        }

        return maxLength;
    }
};
```



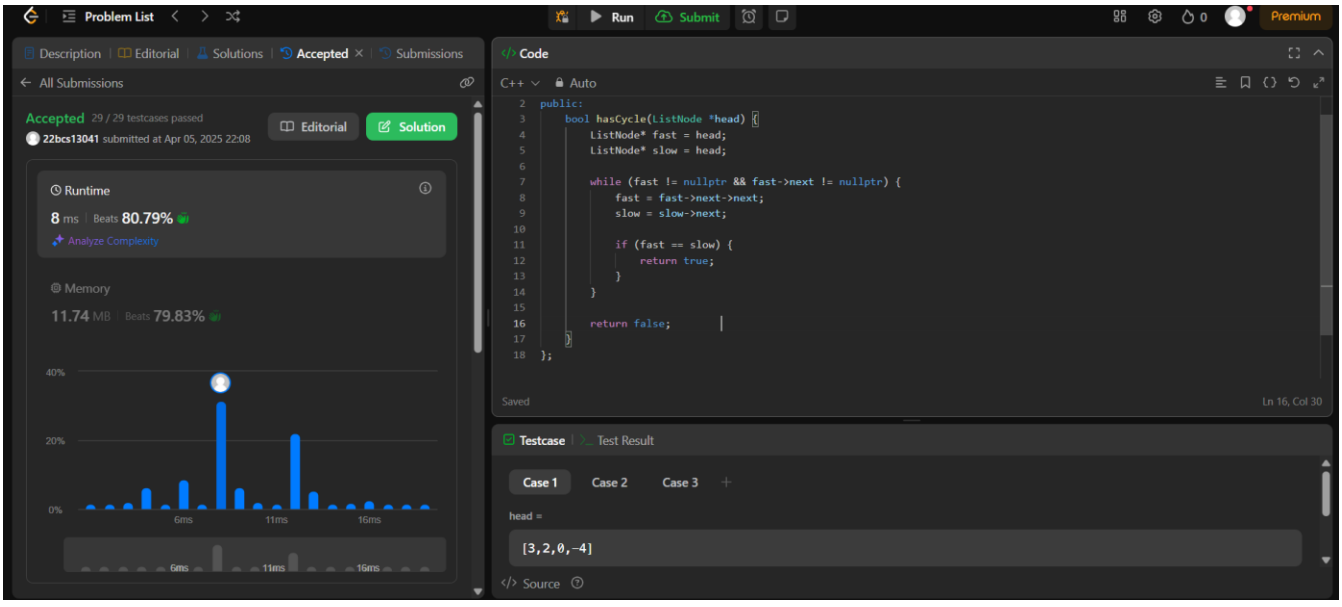
Reverse Linked list

```
class Solution {
public:
    ListNode* reverseBetween(ListNode* head, int left, int right) {
        if (!head || left == right) return head;

        ListNode dummy(0);
        dummy.next = head;
        ListNode* prev = &dummy;

        for (int i = 0; i < left - 1; ++i) {
            prev = prev->next;
        }
        ListNode* current = prev->next;

        for (int i = 0; i < right - left; ++i) {
            ListNode* next_node = current->next;
            current->next = next_node->next;
            next_node->next = prev->next;
            prev->next = next_node;
        }

        return dummy.next;
    }
}
```



Linked list has no cycle

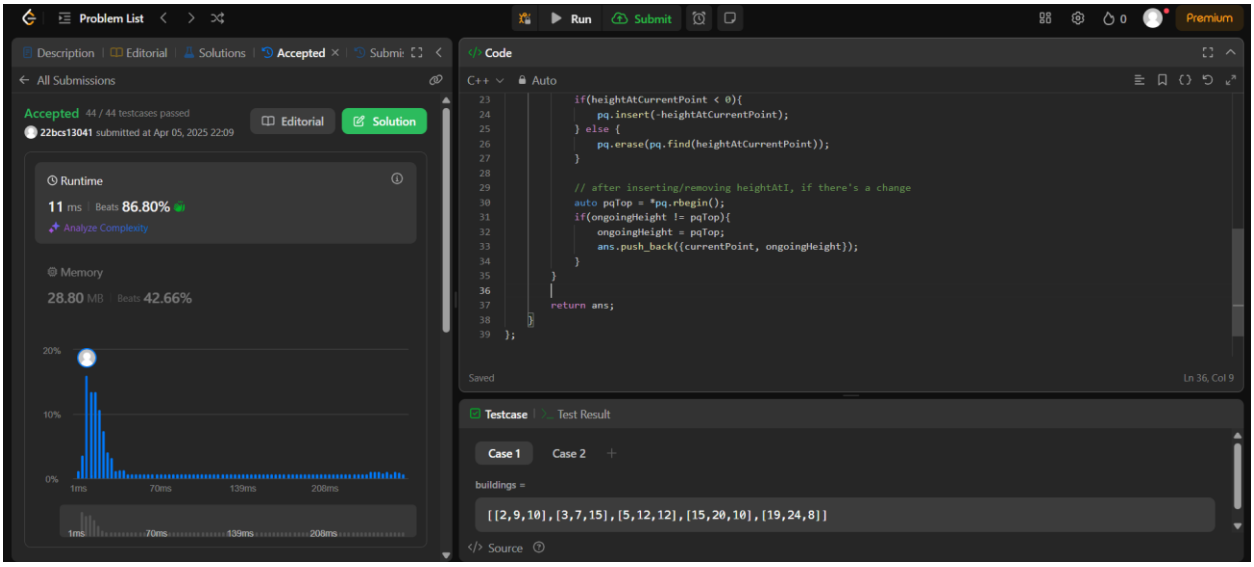
```
class Solution {
public:
    bool hasCycle(ListNode *head) {
        ListNode* fast = head;
        ListNode* slow = head;
        while (fast != nullptr && fast->next != nullptr) {
            fast = fast->next->next;
            slow = slow->next;
            if (fast == slow) {
                return true;
            }
        }
        return false;
    }
};
```



Skyline problem

```
class Solution {
public:
   vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
      vector<vector<int>> ans;
      multiset<int> pq{0};

      vector<pair<int, int>> points;

      for(auto b: buildings){
         points.push_back({b[0], -b[2]});
         points.push_back({b[1], b[2]});
      }

      sort(points.begin(), points.end());

      int ongoingHeight = 0;

      // points.first = x coordinate, points.second = height
      for(int i = 0; i < points.size(); i++){
         int currentPoint = points[i].first;
         int heightAtCurrentPoint = points[i].second;

         if(heightAtCurrentPoint < 0){
            pq.insert(-heightAtCurrentPoint);
         } else {
            pq.erase(pq.find(heightAtCurrentPoint));
         }

         // after inserting/removing heightAtI, if there's a change
         auto pqTop = *pq.rbegin();
         if(ongoingHeight != pqTop){
            ongoingHeight = pqTop;
            ans.push_back({currentPoint, ongoingHeight});
         }
      }
```

```
        return ans;
    }
};
```


Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Accepted
All Submissions
Accepted 44 / 44 testcases passed
22bcs13041 submitted at Apr 05, 2025 22:09
Editorial
Solution
Runtime
11 ms Beats 86.80%
Analyze Complexity
Memory
28.80 MB Beats 42.66%
20%
10%
0%
1ms
70ms
139ms
208ms
Code
C++
Auto
if(heightAtCurrentPoint < 0){
pq.insert(-heightAtCurrentPoint);
} else {
pq.erase(pq.find(heightAtCurrentPoint));
}
// after inserting/removing heightAtI, if there's a change
auto pqTop = *pq.rbegin();
if(ongoingHeight != pqTop){
ongoingHeight = pqTop;
ans.push_back({currentPoint, ongoingHeight});
}
}
return ans;
};
Saved
Ln 36, Col 9
Testcase
Test Result
Case 1
Case 2
buildings =
[[2,9,10],[3,7,15],[5,12,12],[15,20,10],[19,24,8]]
Source