Assignment Average Learner

Student Name: Riyan Reyaz **UID:22BCS11986**

Branch: CSE **Section/Group: NTPP 603/B**

Semester: 06 **Date of Performance: 03/04/2025**

Subject Name: AP Lab II **Subject Code: 22CSP-351**

1. **Aim**:
   a) Two Sum.
   b) Longest Substring without Repeating Characters.
   c) Palindrome Number.
   d) Detect a Cycle in a linked list.

2. **Source Code:**

a.

```
class Solution {
 public:
  vector<int> twoSum(vector<int>& nums, int target) {
    unordered_map<int, int> numToIndex;

    for (int i = 0; i < nums.size(); ++i) {
      if (const auto it = numToIndex.find(target - nums[i]);
          it != numToIndex.cend())
        return {it->second, i};
      numToIndex[nums[i]] = i;
    }

    throw;
  }
};
```

b.

```
class Solution {
 public:
  int lengthOfLongestSubstring(string s) {
    int ans = 0;
    vector<int> count(128);

    for (int l = 0, r = 0; r < s.length(); ++r) {
      ++count[s[r]];
      while (count[s[r]] > 1)
        --count[s[l++]];
      ans = max(ans, r - l + 1);
    }

    return ans;
  }
};
```

C.

```
class Solution {
 public:
  bool isPalindrome(int x) {
    if (x < 0)
      return false;
    long reversed = 0;
    int y = x;
    while (y > 0) {
      reversed = reversed * 10 + y % 10;
      y /= 10;
    return reversed == x;
  }
};
```

d.

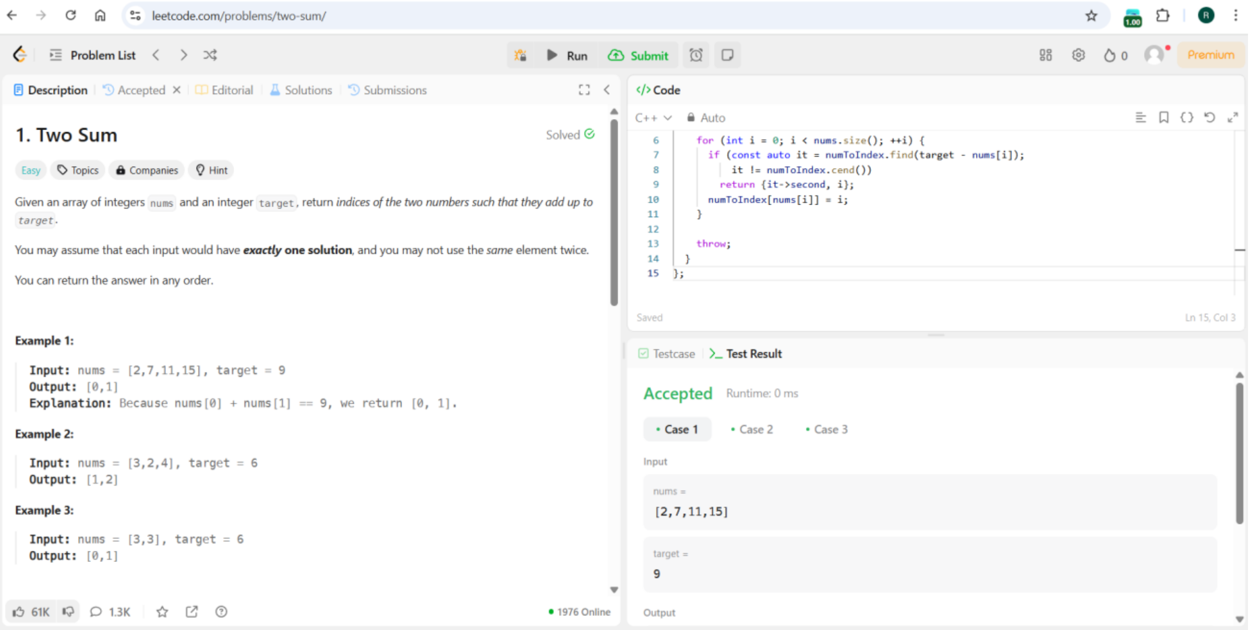
```
class Solution {
 public:
  bool hasCycle(ListNode* head) {
    ListNode* slow = head;
    ListNode* fast = head;

    while (fast != nullptr && fast->next != nullptr) {
      slow = slow->next;
      fast = fast->next->next;
      if (slow == fast)
        return true;
    }

    return false;
  }
};
```

Screenshot of Outputs:

a.



1. Two Sum

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to* `target`.

You may assume that each input would have ***exactly* one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

```
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

Example 2:

```
Input: nums = [3,2,4], target = 6
Output: [1,2]
```

Example 3:

```
Input: nums = [3,3], target = 6
Output: [0,1]
```

C++ Auto

```
    for (int i = 0; i < nums.size(); ++i) {
      if (const auto it = numToIndex.find(target - nums[i]);
          it != numToIndex.cend())
        return {it->second, i};
      numToIndex[nums[i]] = i;
    }

    throw;
  }
};
```

Accepted Runtime: 0 ms

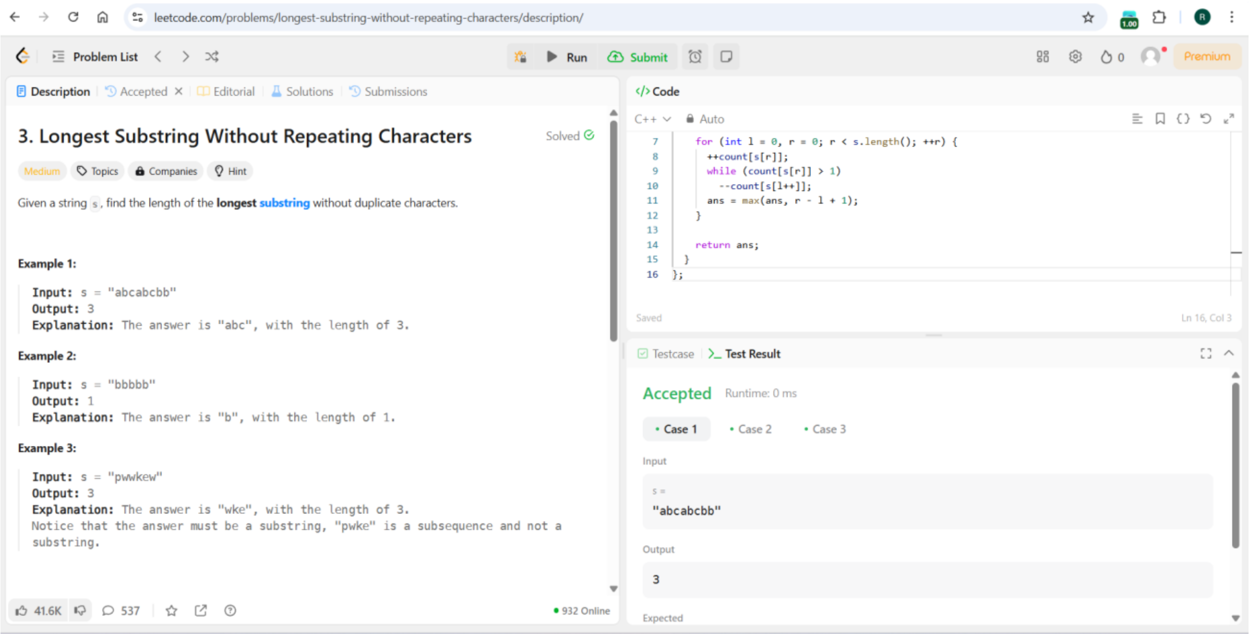
Input

nums =
[2,7,11,15]

target =
9

Output

b.



3. Longest Substring Without Repeating Characters

Given a string `s`, find the length of the **longest substring** without duplicate characters.

Example 1:

```
Input: s = "abcabcbb"
Output: 3
Explanation: The answer is "abc", with the length of 3.
```

Example 2:

```
Input: s = "bbbbb"
Output: 1
Explanation: The answer is "b", with the length of 1.
```

Example 3:

```
Input: s = "pwwkew"
Output: 3
Explanation: The answer is "wke", with the length of 3.
Notice that the answer must be a substring, "pwke" is a subsequence and not a
substring.
```

```
    for (int l = 0, r = 0; r < s.length(); ++r) {
      ++count[s[r]];
      while (count[s[r]] > 1)
        --count[s[l++]];
      ans = max(ans, r - l + 1);
    }

    return ans;
  }
};
```

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

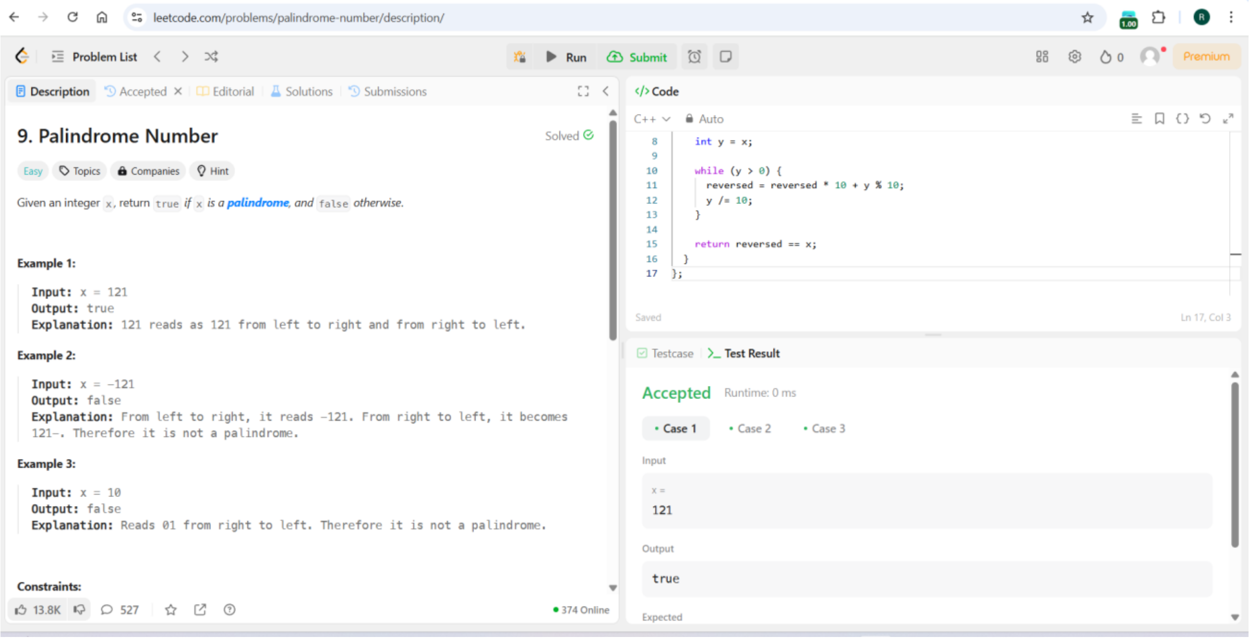
```
s =
"abcabcbb"
```

Output

```
3
```

Expected

c.



9. Palindrome Number

Given an integer `x`, return `true` *if* `x` *is a* ***palindrome****, and* `false` *otherwise.*

Example 1:

```
Input: x = 121
Output: true
Explanation: 121 reads as 121 from left to right and from right to left.
```

Example 2:

```
Input: x = -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it becomes
121-. Therefore it is not a palindrome.
```

Example 3:

```
Input: x = 10
Output: false
Explanation: Reads 01 from right to left. Therefore it is not a palindrome.
```

Constraints:

```
    int y = x;

    while (y > 0) {
      reversed = reversed * 10 + y % 10;
      y /= 10;
    }

    return reversed == x;
  }
};
```

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

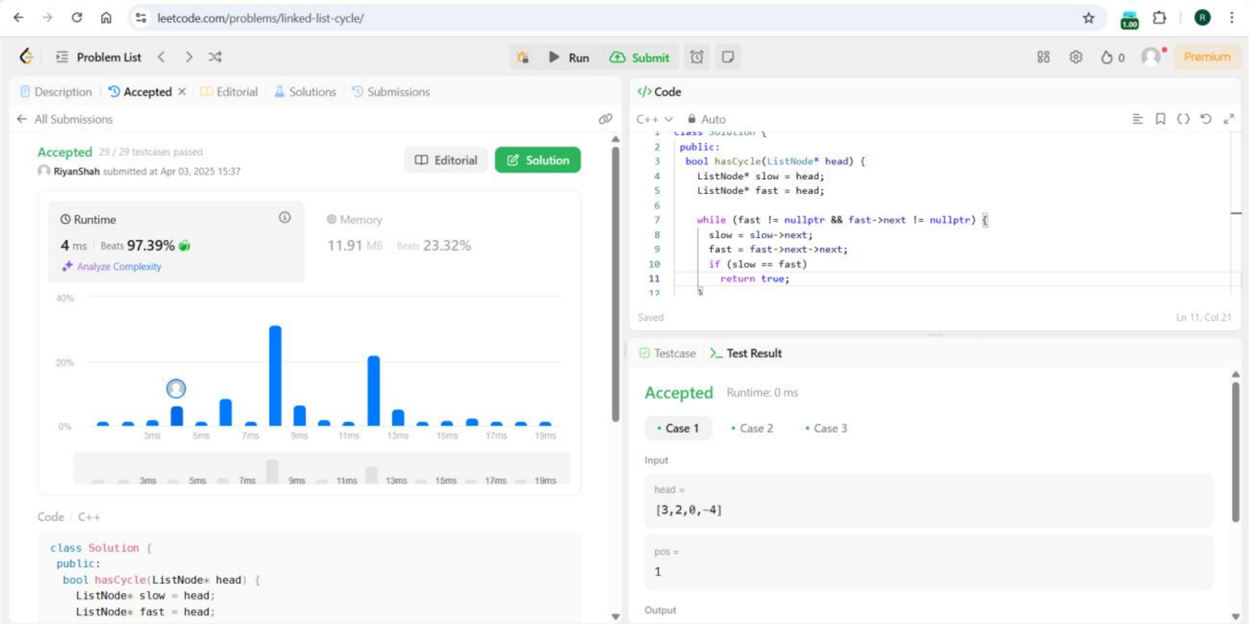
```
x =
121
```

Output

```
true
```

Expected

d.



Accepted 29 / 29 testcases passed

RiyanShah submitted at Apr 03, 2025 15:37

Runtime

4 ms | Beats 97.39%

Analyze Complexity

Memory

11.91 MB | Beats 23.32%

Code | C++

```
class Solution {
 public:
  bool hasCycle(ListNode* head) {
    ListNode* slow = head;
    ListNode* fast = head;
```

```
class Solution {
 public:
  bool hasCycle(ListNode* head) {
    ListNode* slow = head;
    ListNode* fast = head;

    while (fast != nullptr && fast->next != nullptr) {
      slow = slow->next;
      fast = fast->next->next;
      if (slow == fast)
        return true;
    }
```

Saved

Ln 11, Col 21

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

head =

```
[3,2,0,-4]
```

pos =

```
1
```

Output