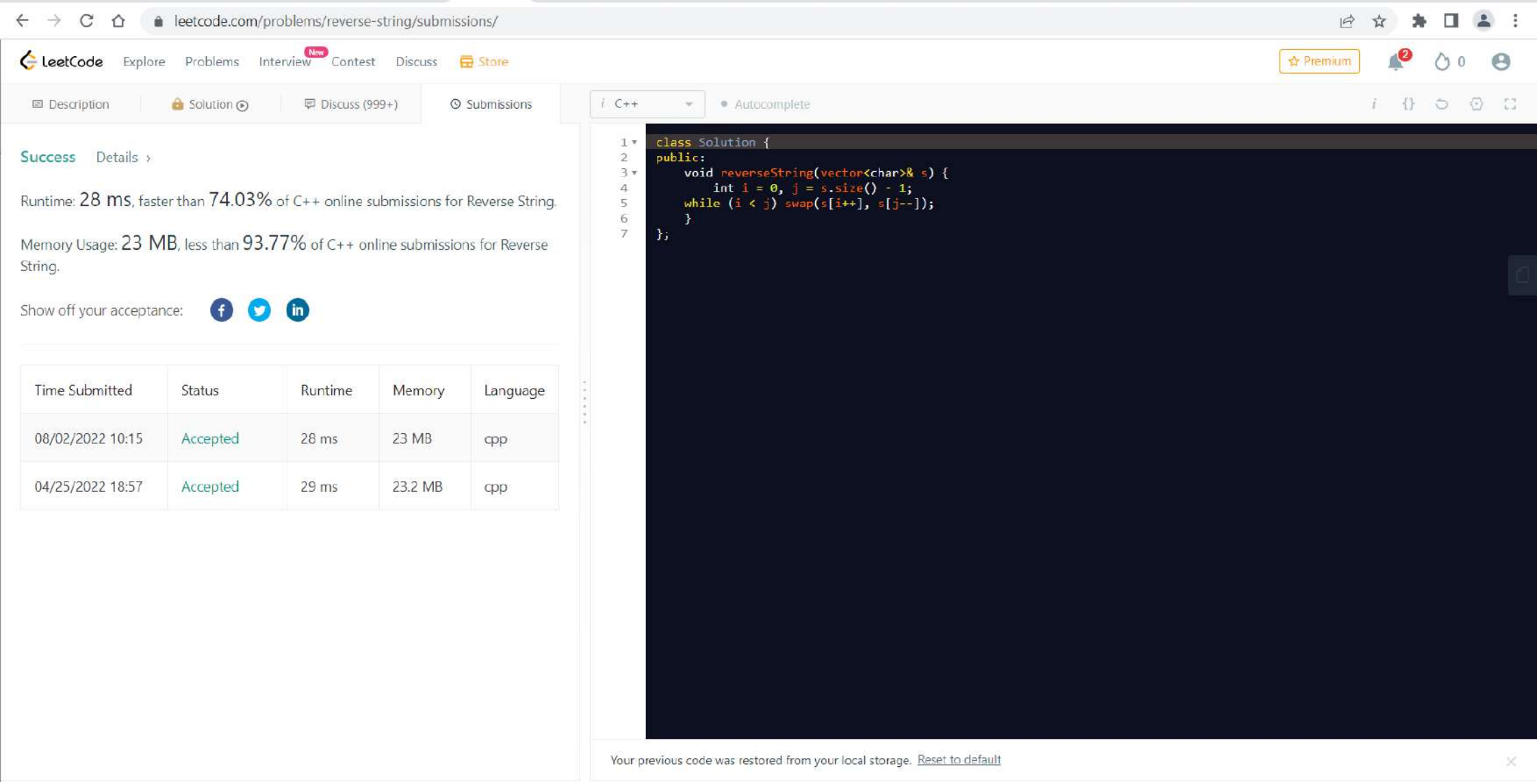
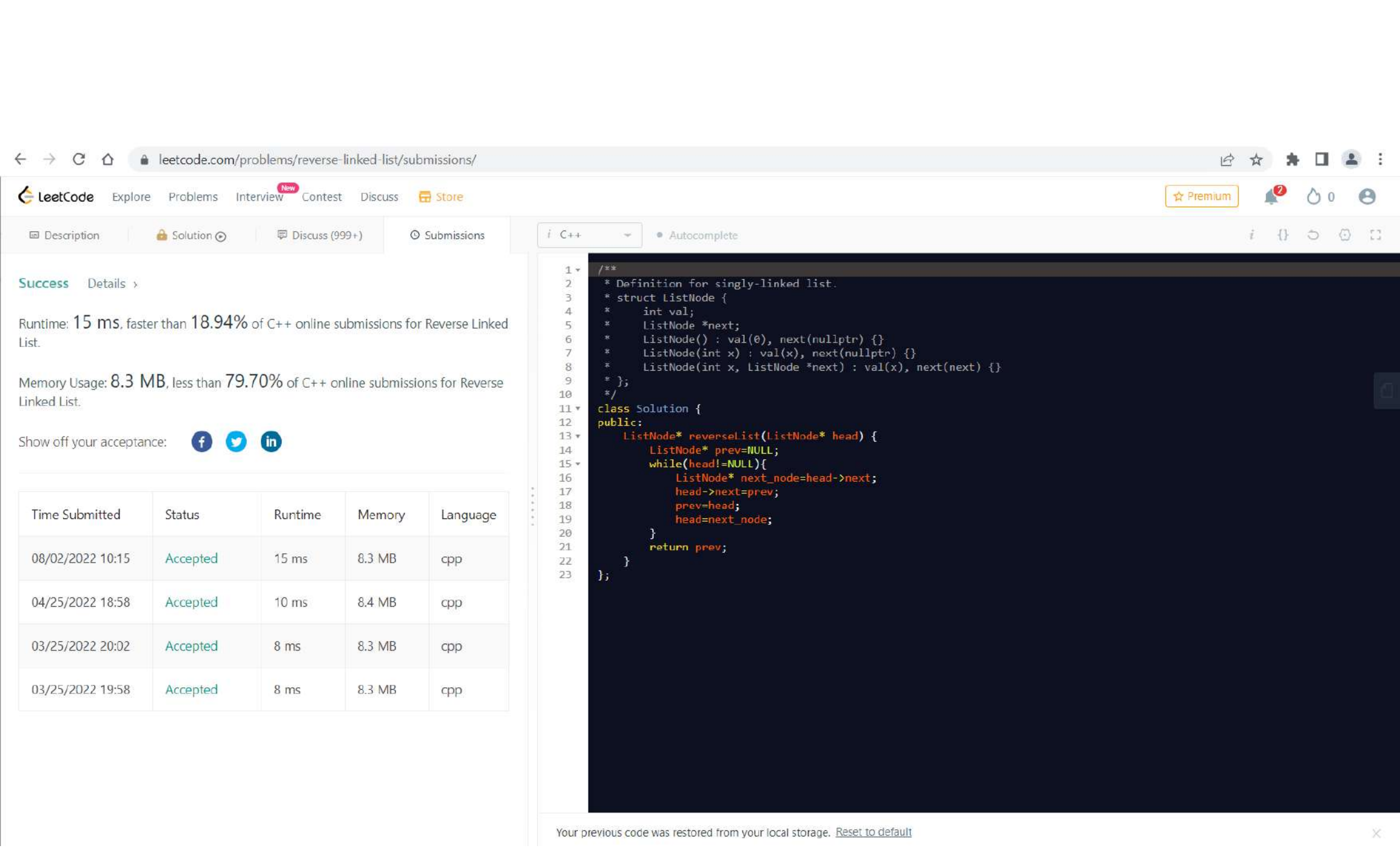
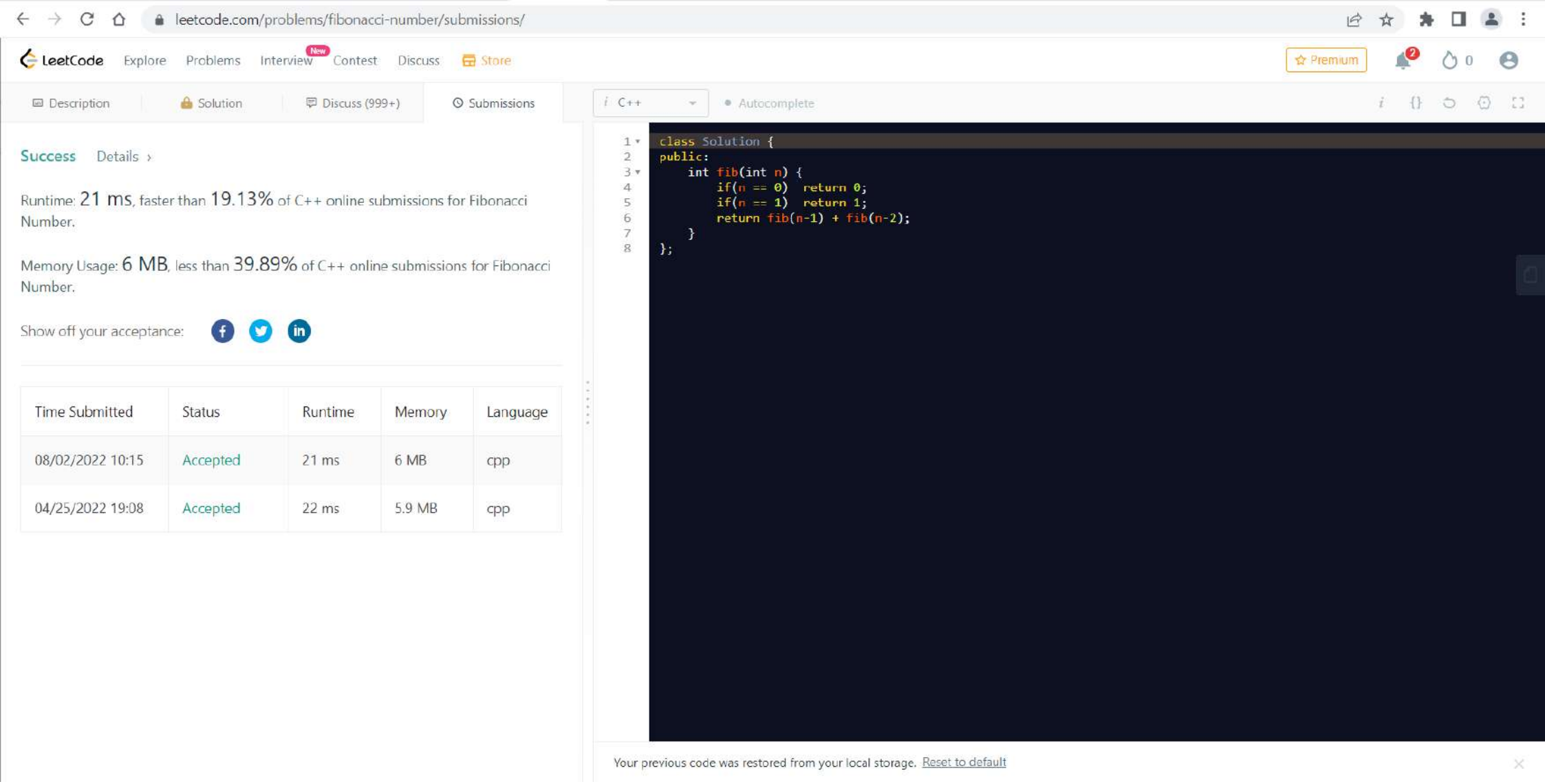

leetcode.com/problems/reverse-string/submissions/
LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
Premium
Description
Solution
Discuss (999+)
Submissions
C++
Autocomplete
Success
Details ›
Runtime: 28 ms, faster than 74.03% of C++ online submissions for Reverse String.
Memory Usage: 23 MB, less than 93.77% of C++ online submissions for Reverse String.
Show off your acceptance:
Time Submitted
Status
Runtime
Memory
Language
08/02/2022 10:15
Accepted
28 ms
23 MB
cpp
04/25/2022 18:57
Accepted
29 ms
23.2 MB
cpp
class Solution {
public:
void reverseString(vector<char>& s) {
int i = 0, j = s.size() - 1;
while (i < j) swap(s[i++], s[j--]);
}
};
Your previous code was restored from your local storage.
Reset to default


leetcode.com/problems/reverse-linked-list/submissions/
LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
Premium
Description
Solution
Discuss (999+)
Submissions
C++
Autocomplete
Success
Details ›
Runtime: 15 ms, faster than 18.94% of C++ online submissions for Reverse Linked List.
Memory Usage: 8.3 MB, less than 79.70% of C++ online submissions for Reverse Linked List.
Show off your acceptance:
Time Submitted
Status
Runtime
Memory
Language
08/02/2022 10:15
Accepted
15 ms
8.3 MB
cpp
04/25/2022 18:58
Accepted
10 ms
8.4 MB
cpp
03/25/2022 20:02
Accepted
8 ms
8.3 MB
cpp
03/25/2022 19:58
Accepted
8 ms
8.3 MB
cpp
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode() : val(0), next(nullptr) {}
* ListNode(int x) : val(x), next(nullptr) {}
* ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* reverseList(ListNode* head) {
ListNode* prev=NULL;
while(head!=NULL){
ListNode* next_node=head->next;
head->next=prev;
prev=head;
head=next_node;
}
return prev;
}
};
Your previous code was restored from your local storage. Reset to default


leetcode.com/problems/fibonacci-number/submissions/
LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
Premium
Description
Solution
Discuss (999+)
Submissions
C++
Autocomplete
Success
Details ›
Runtime: 21 ms, faster than 19.13% of C++ online submissions for Fibonacci Number.
Memory Usage: 6 MB, less than 39.89% of C++ online submissions for Fibonacci Number.
Show off your acceptance:
Time Submitted
Status
Runtime
Memory
Language
08/02/2022 10:15
Accepted
21 ms
6 MB
cpp
04/25/2022 19:08
Accepted
22 ms
5.9 MB
cpp
class Solution {
public:
int fib(int n) {
if(n == 0) return 0;
if(n == 1) return 1;
return fib(n-1) + fib(n-2);
}
};
Your previous code was restored from your local storage.
Reset to default

leetcode.com/problems/power-of-four/submissions/

LeetCode Explore Problems Interview (New) Contest Discuss Store ☆ Premium 2 0

Description Solution Discuss (999+) Submissions

Success Details ›

Runtime: 4 ms, faster than 40.30% of C++ online submissions for Power of Four.

Memory Usage: 5.9 MB, less than 25.30% of C++ online submissions for Power of Four.

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
08/02/2022 10:15	Accepted	4 ms	5.9 MB	cpp
04/25/2022 19:40	Accepted	0 ms	5.9 MB	cpp

C++ Autocomplete

```cpp
class Solution {
public:
    bool isPowerOfFour(int n) {
        return n > 0 && (n & (n - 1)) == 0 && (n - 1) % 3 == 0;
    }
};
```

Your previous code was restored from your local storage. Reset to default

You have attempted to submit too soon. Please try again in a few seconds, or subscribe to reduce wait time.

Time Submitted	Status	Runtime	Memory	Language
04/25/2022 19:14	Accepted	13 ms	14.9 MB	cpp
04/25/2022 19:09	Accepted	7 ms	14.6 MB	cpp
03/25/2022 00:03	Accepted	15 ms	14.7 MB	cpp

C++ Autocomplete

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        if(list1==NULL)
            return list2;
        if(list2==NULL)
            return list1;
        if(list1 -> val <= list2 -> val)
        {
            list1 -> next = mergeTwoLists(list1 -> next, list2);
            return list1;
        }
        else
        {
            list2 -> next = mergeTwoLists(list1, list2 -> next);
            return list2;
        }
    }
};
```

Your previous code was restored from your local storage. Reset to default

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

LeetCode Explore Problems Interview (New) Contest Discuss Store ☆ Premium 2 0

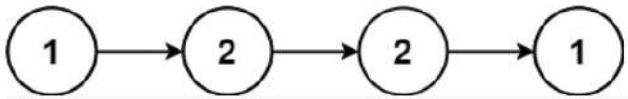
Description Solution Discuss (999+) Submissions C++ Autocomplete

234. Palindrome Linked List

Easy 10119 612 Add to List Share

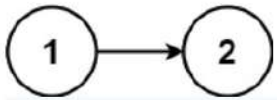
Given the `head` of a singly linked list, return `true` if it is a palindrome.

Example 1:




```
Input: head = [1,2,2,1]
Output: true
```

Example 2:




```
Input: head = [1,2]
Output: false
```

Constraints:

- The number of nodes in the list is in the range `[1, 10^5]`.
- `0 <= Node.val <= 9`

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* temp;
    bool isPalindrome(ListNode* head) {
        temp = head;
        return check(head);
    }

    bool check(ListNode* p) {
        if (NULL == p) return true;
        bool isPal = check(p->next) & (temp->val == p->val);
        temp = temp->next;
        return isPal;
    }
};
```

Your previous code was restored from your local storage. Reset to default

leetcode.com/problems/remove-linked-list-elements/

LeetCode Explore Problems Interview (New) Contest Discuss Store ☆ Premium

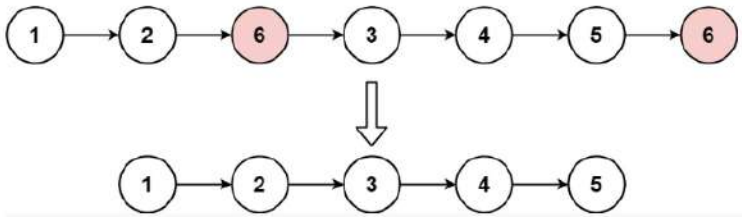
Description Solution Discuss (999+) Submissions C++ Autocomplete

203. Remove Linked List Elements

Easy 5524 175 Add to List Share

Given the `head` of a linked list and an integer `val`, remove all the nodes of the linked list that has `Node.val == val`, and return *the new head*.

Example 1:




```
Input: head = [1,2,6,3,4,5,6], val = 6
Output: [1,2,3,4,5]
```

Example 2:

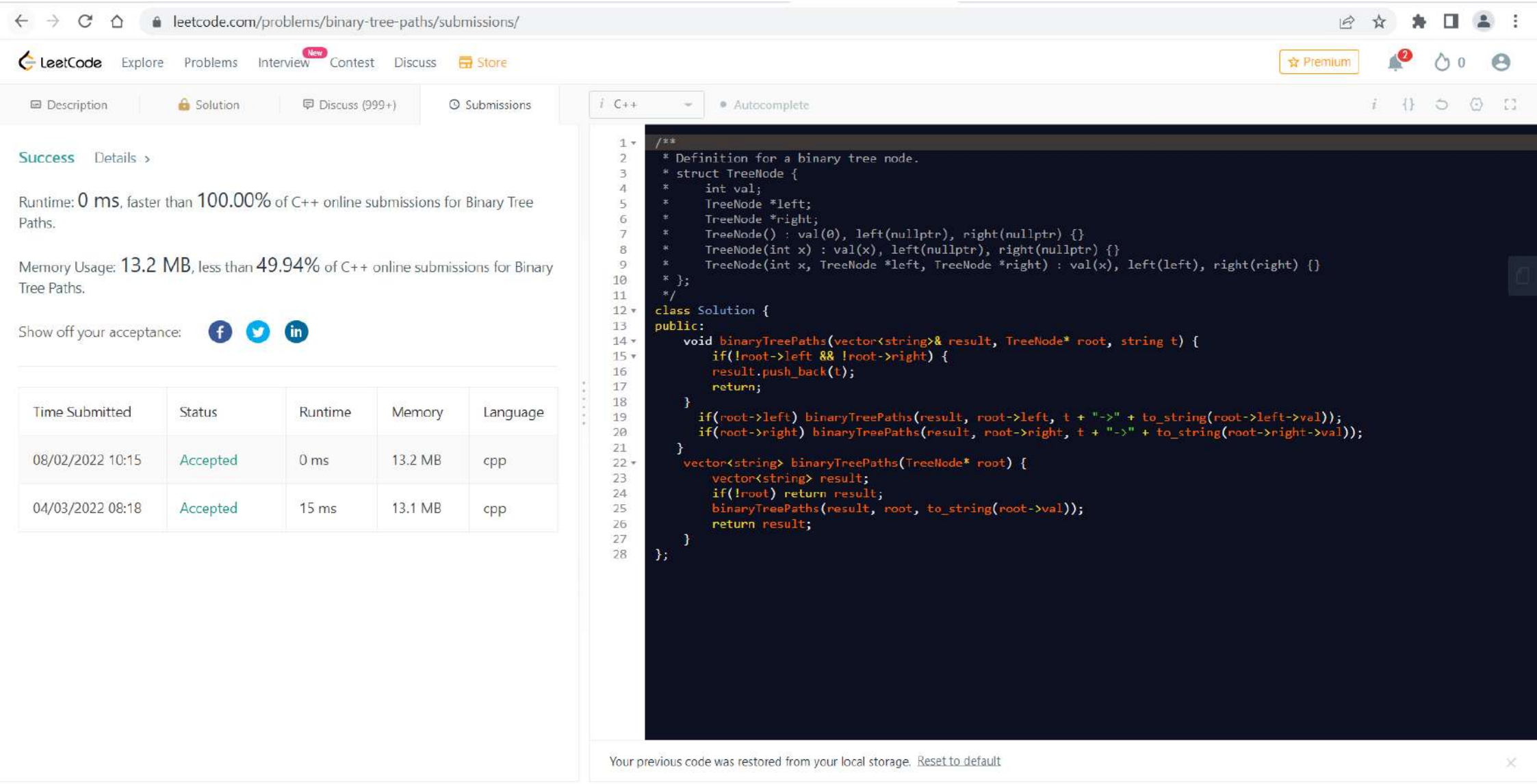
```
Input: head = [], val = 1
Output: []
```

Example 3:

```
Input: head = [7,7,7,7], val = 7
Output: []
```

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* removeElements(ListNode* head, int val) {
        if(head==NULL) return NULL;
        ListNode* start=new ListNode(-1);
        start->next=head;
        ListNode* cur=start;
        while(cur!=NULL  && cur->next!=NULL){
            if(cur->next->val==val){
                ListNode* tem=cur->next;
                cur->next=tem->next;
                delete tem;
            }
            else{
                cur=cur->next;
            }
        }
        return start->next;
    }
};
```

Your previous code was restored from your local storage. Reset to default


leetcode.com/problems/binary-tree-paths/submissions/
LeetCode
Explore
Problems
Interview
New
Contest
Discuss
Store
Premium
Description
Solution
Discuss (999+)
Submissions
C++
Autocomplete
Success
Details ›
Runtime: 0 ms, faster than 100.00% of C++ online submissions for Binary Tree Paths.
Memory Usage: 13.2 MB, less than 49.94% of C++ online submissions for Binary Tree Paths.
Show off your acceptance:
Time Submitted
Status
Runtime
Memory
Language
08/02/2022 10:15
Accepted
0 ms
13.2 MB
cpp
04/03/2022 08:18
Accepted
15 ms
13.1 MB
cpp
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode() : val(0), left(nullptr), right(nullptr) {}
* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
* TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
* };
*/
class Solution {
public:
void binaryTreePaths(vector<string>& result, TreeNode* root, string t) {
if(!root->left && !root->right) {
result.push_back(t);
return;
}
if(root->left) binaryTreePaths(result, root->left, t + "->" + to_string(root->left->val));
if(root->right) binaryTreePaths(result, root->right, t + "->" + to_string(root->right->val));
}
vector<string> binaryTreePaths(TreeNode* root) {
vector<string> result;
if(!root) return result;
binaryTreePaths(result, root, to_string(root->val));
return result;
}
};
Your previous code was restored from your local storage.
Reset to default

22. Generate Parentheses

Medium 14285 538 Add to List Share

Given `n` pairs of parentheses, write a function to *generate all combinations of well-formed parentheses.*

Example 1:

```
Input: n = 3
Output: ["((()))","(()())","(())()","()(())","()()()"]
```

Example 2:

```
Input: n = 1
Output: ["()"]
```

Constraints:

- `1 <= n <= 8`

Accepted 1,147,717 Submissions 1,615,324

Seen this question in a real interview before? Yes No

Companies

```cpp
class Solution {
public:
    vector<string> generateParenthesis(int n) {
        vector<string> res;
        addingpar(res, "", n, 0);
        return res;
    }
    void addingpar(vector<string> &v, string str, int n, int m){
        if(n==0 && m==0) {
            v.push_back(str);
            return;
        }
        if(m > 0){ addingpar(v, str+")", n, m-1); }
        if(n > 0){ addingpar(v, str+"(", n-1, m+1); }
    }
};
```