

## Leetcode problems

### 1.Score of parantheses

Problem List < > 🔍

Run Submit ⌛ 📄

88 ⚙️ 🔥 0 👤 Premium

Description | Editor | Solutions | Submissions

< All Submissions 🔗

Accepted

komar\_sudarshan submitted at Feb 19, 2024 10:30

Editorial Solution

Runtime

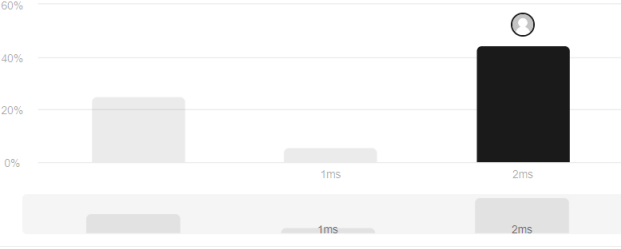
2 ms

Beats 69.44% of users with C

Memory

5.74 MB

Beats 27.78% of users with C



Code | C

```
int scoreOfParentheses(char* s) {
    int* st = (int*)malloc(strlen(s) * sizeof(int));
    int top = -1;
    int score = 0;
    for (int i = 0; i < strlen(s); i++) {
        if (s[i] == '(') {
            st[++top] = score;
            score = 0;
        }
    }
    free(st);
    return score;
}
```

</> Code

C v Auto

1 int scoreOfParentheses(char\* s) {
2 int\* st = (int\*)malloc(strlen(s) \* sizeof(int));
3 int top = -1;
4 int score = 0;
5 for (int i = 0; i < strlen(s); i++) {
6 if (s[i] == '(') {
7 st[++top] = score;
8 score = 0;
9 } else {
10 score = st[top--] + ((2 \* score) > 1 ? (2 \* score) : 1);
11 }
12 }
13 free(st);
14 return score;
15 }

Saved to local Ln 15, Col 2

Testcase > Test Result

Case 1 Case 2 Case 3 +

s =

"()"

</> Source

### 2.Deleting middle node in a single linked list

Problem List < > 🔍

Run Submit ⌛ 📄

88 ⚙️ 🔥 0 👤 Premium

Description | Editor | Solutions | Submissions

< All Submissions 🔗

Accepted

komar\_sudarshan submitted at Feb 19, 2024 10:22

Editorial Solution

Runtime

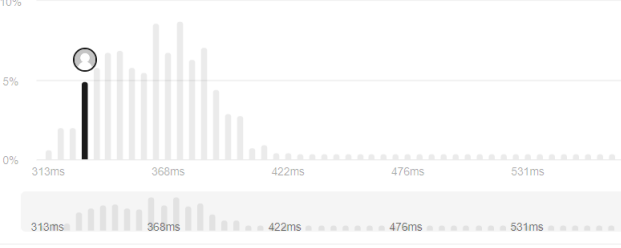
333 ms

Beats 91.71% of users with C

Memory

77.86 MB

Beats 73.12% of users with C



Code | C

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     struct ListNode *next;
 * };
 */
struct ListNode* deleteMiddle(struct ListNode* head) {
```

</> Code

C v Auto

6 \* };
7 \*/
8 struct ListNode\* deleteMiddle(struct ListNode\* head) {
9 int count=0;
10 struct ListNode \*ptr;
11 ptr=head;
12 while(ptr!=NULL){
13 count++;
14 ptr=ptr->next;
15 }
16 int mid=count/2;
17 struct ListNode \*prev=NULL;
18 ptr=head;
19 if(head->next==NULL){
20 head=NULL;
21 return head;
22 }
23 for(int i=0;i<mid;i++){
24 prev=ptr;
25 ptr=ptr->next;
26 }
27 prev->next=ptr->next;
28 free(ptr);
29 return head;
30 }

Saved to local Ln 16, Col 21

Testcase > Test Result

Case 1 Case 2 Case 3 +

</> Source

### 3. Odd even grouping in single linked list

Problem List

Run

Submit

0

Premium

Description

Editorial

Solutions

Submissions

All Submissions

Accepted

komar\_sudarshan submitted at Feb 19, 2024 10:34

Editorial

Solution

Runtime

6 ms

Beats 50.28% of users with C

Memory

6.67 MB

Beats 85.66% of users with C

Runtime (ms)	Percentage of Users
3	~8%
4	~4%
5	~22%
6	~10%
7	~25%
8	~10%
9	~2%
10	~1%
11	~1%
12	~1%
13	~1%

Code

C

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     struct ListNode *next;
 * };
 */
// struct ListNode* oddEvenList(struct ListNode* head) {
```

Code

C

Auto

```
20 // prev->next=temp;
21 // return head;
22 // }
23 struct ListNode* oddEvenList(struct ListNode* head) {
24     if (head == NULL || head->next == NULL) {
25         return head;
26     }
27     struct ListNode* oddHead = head;
28     struct ListNode* evenHead = head->next;
29     struct ListNode* odd = oddHead;
30     struct ListNode* even = evenHead;
31     while (even!=NULL && even->next!=NULL) {
32         odd->next = even->next;
33         odd = odd->next;
34         even->next = odd->next;
35         even = even->next;
36     }
37     odd->next = evenHead;
38     return head;
39 }
40
```

Saved to local

Ln 1, Col 1

Testcase

Test Result

Accepted

Runtime: 4 ms

Case 1

Case 2

Input

head =

1 2 3 4 5