

7. leetcode: Delete middle node of a linked list

```
struct ListNode *deleteMiddle(struct ListNode *head)
{
```

```
    struct ListNode *temp;
    struct ListNode *midnode = NULL;
```

```
    int n = 0, i = 1;
```

```
    temp = head;
```

```
    while (temp != NULL)
    {
```

```
        n++;
```

```
        temp = temp->next;
```

```
    }
```

```
    temp = head;
```

```
    int mid = n/2;
```

```
    if (head->next == NULL)
```

```
    { head = NULL;
```

```
      return head;
```

```
    }
```

```
    else {
```

```
        while (i < mid)
```

```
        {
```

```
            temp = temp->next;
```

```
            i++;
```

```
        }
```

```
    }
```

```
    midnode = temp->next;
```

```
    temp->next = midnode->next;
```

```
    free(midnode);
```

```
    return head;
```

```
}
```

Accepted  
19/05/2024



Problem List



Run



Submit



Premium

Description | Editorial | Solutions | Submissions

All Submissions

Accepted

swapnil\_sahil submitted at Feb 20, 2024 00:14

Editorial

Solution

Runtime

328 ms

Beats 95.55% of users with C

Memory

77.95 MB

Beats 59.45% of users with C

10%

5%

0%

313ms 368ms 422ms 476ms 531ms

313ms 368ms 422ms 476ms 531ms

Code | C

Code

C Auto

```
8 struct ListNode* deleteMiddle(struct ListNode* head) {
9     struct ListNode *temp;
10    struct ListNode *midnode=NULL;
11    int n=0,i=1;
12    temp=head;
13    while(temp!=NULL)
14    {
15        n++;
16        temp=temp->next;
17    }
18    temp=head;
19    int mid=n/2;
20
21    if(head->next==NULL)
22    {
23        head=NULL;
24        return head;
25    }
26    else{
27        while(i<mid)
28        {
29            temp=temp->next;
30        }
```

Saved to local

Activate Windows

Ln 1, Col 1

Go to Settings to activate Windows.

Testcase | Test Result

### 3. Leetcode : Odd even linked list.

```

struct ListNode * oddEvenList(struct ListNode * head) {
    struct ListNode * odd = head;
    struct ListNode * even, * evenTail;

    if (head == NULL || head->next == NULL)
    {
        return head;
    }

    even = head->next;
    evenTail = even;

    while (even != NULL && even->next != NULL)
    {
        odd->next = even->next;
        odd = odd->next;
        even->next = odd->next;
        even = even->next;
    }

    odd->next = evenTail;
    return head;
}
    
```

Accepted

swapnil\_sahil submitted at Feb 19, 2024 23:47

Editorial

Solution

Runtime

7 ms

Beats 46.55% of users with C

Memory

6.95 MB

Beats 56.98% of users with C



Code

C Auto

```
8 struct ListNode* oddEvenList(struct ListNode* head) {
9     struct ListNode *odd=head;
10    struct ListNode *even;
11    struct ListNode *evenTail;
12
13    if(head==NULL || head->next==NULL)
14    {
15        return head;
16    }
17    even=head->next;
18    evenTail=even;
19
20    while(even!=NULL && even->next!=NULL)
21    {
22        odd->next=even->next;
23        odd=odd->next;
24        even->next=odd->next;
25        even=even->next;
26    }
27
28    odd->next=evenTail;
29    return head;
30 }
```