

203.Remove Linked List Elements

```
#include <stdio.h>
#include <stdlib.h>

/* Definition for singly-linked list */
struct ListNode {
    int val;
    struct ListNode *next;
};

/* Function to create a new node */
struct ListNode* createNode(int val) {
    struct ListNode* newNode = (struct ListNode*)malloc(sizeof(struct ListNode));
    newNode->val = val;
    newNode->next = NULL;
    return newNode;
}

/* Function to display the linked list */
void displayList(struct ListNode* head) {
    struct ListNode* temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->val);
        temp = temp->next;
    }
    printf("NULL\n");
}

/* Function to remove all elements with a specific value */
struct ListNode* removeElements(struct ListNode* head, int val) {
    /* Remove nodes from the beginning if they match val */
```

```

while (head != NULL && head->val == val) {
    struct ListNode* temp = head;
    head = head->next;
    free(temp);
}

struct ListNode* current = head;

/* Traverse the list and remove matching nodes */
while (current != NULL && current->next != NULL) {
    if (current->next->val == val) {
        struct ListNode* temp = current->next;
        current->next = current->next->next;
        free(temp);
    } else {
        current = current->next;
    }
}

return head;
}

/* Function to create a linked list from user input */
struct ListNode* createList(int n) {
    int val;
    struct ListNode *head = NULL, *tail = NULL;

    for (int i = 0; i < n; i++) {
        printf("Enter element %d: ", i + 1);
        scanf("%d", &val);
        struct ListNode* newNode = createNode(val);

```

```
if (head == NULL) {  
    head = tail = newNode;  
}  
else {  
    tail->next = newNode;  
    tail = newNode;  
}  
  
return head;  
}  
  
int main() {  
    int n, val;  
  
    printf("Enter number of elements in the list: ");  
    scanf("%d", &n);  
  
    printf("Enter elements of the list:\n");  
    struct ListNode* head = createList(n);  
  
    printf("Original list:\n");  
    displayList(head);  
  
    printf("Enter value to remove: ");  
    scanf("%d", &val);  
  
    head = removeElements(head, val);  
  
    printf("List after removing %d:\n", val);  
    displayList(head);
```

```
    return 0;  
}
```

OUTPUT:

The screenshot shows a terminal window titled "Leetcode Problems" with three separate code snippets running sequentially. Each snippet asks for the number of elements in a list and then prompts for individual element values. The first two snippets show the original list and the list after removing a specified value. The third snippet shows the list after removing the last element.

```
c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems>cd "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems" && gcc 203_RemoveLLElements.c -o 203_RemoveLLElements  
Enter number of elements in the list: 6  
Enter elements of the list:  
Original list:  
1 -> 2 -> 6 -> 3 -> 4 -> 5 -> NULL  
Enter value to remove: 6  
List after removing 6:  
1 -> 2 -> 3 -> 4 -> 5 -> NULL  
c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems>cd "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems" && gcc 203_RemoveLLElements.c -o 203_RemoveLLElements  
Enter number of elements in the list: 4  
Enter elements of the list:  
Original list:  
1 -> 2 -> 3 -> 2 -> NULL  
Enter value to remove: 2  
List after removing 2:  
1 -> 3 -> NULL  
c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems>cd "c:\Users\Mohammed Javeed\OneDrive\Desktop\Javeed\DS-Lab\Leetcode_Problems" && gcc 203_RemoveLLElements.c -o 203_RemoveLLElements  
Enter number of elements in the list: 2  
Enter elements of the list:  
Original list:  
1 -> 1 -> NULL  
Enter value to remove: 1  
List after removing 1:  
NULL
```