203. Remove Linked List Elements

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

struct ListNode {
    int val;
    struct ListNode* next;
};

struct ListNode* removeElements(struct ListNode* head, int val) {
    // Remove nodes from the beginning if needed
    while (head != NULL && head->val == val) {
        struct ListNode* temp = head;
        head = head->next;
        free(temp);
    }

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

```
    return head;
}

void printList(struct ListNode* head) {
    struct ListNode* p = head;
    while (p != NULL) {
        printf("%d ", p->val);
        p = p->next;
    }
    printf("\n");
}

int main() {
    int n, val;

    printf("Enter number of nodes: ");
    scanf("%d", &n);

    struct ListNode* head = NULL;
    struct ListNode* tail = NULL;

    printf("Enter %d values:\n", n);
    for (int i = 0; i < n; i++) {
        int x;
        scanf("%d", &x);

        struct ListNode* newNode = (struct ListNode*)malloc(sizeof(struct ListNode));
        newNode->val = x;
        newNode->next = NULL;
```

```
        if (head == NULL) {
            head = tail = newNode;
        } else {
            tail->next = newNode;
            tail = newNode;
        }
    }

    printf("Enter value to remove: ");
    scanf("%d", &val);

    head = removeElements(head, val);

    printf("List after removal:\n");
    printList(head);

    return 0;
}
```

```
Enter number of nodes: 5
Enter 5 values:
1
2
3
2
5
Enter value to remove: 2
List after removal:
1 3 5
```