class Solution {

public:

ListNode\* deleteDuplicates(ListNode\* head) {

if (head == nullptr || head->next == nullptr) {

return head; // Empty list or single element, no duplicates possible

}

ListNode\* current = head;

while (current != nullptr && current->next != nullptr) {

if (current->val == current->next->val) {

ListNode\* temp = current->next; // Store the duplicate node

current->next = current->next->next; // Skip the duplicate

delete temp; // Deallocate the memory of the duplicate node (important!)

} else {

current = current->next; // Move to the next node if no duplicate

}

}

return head;

}

};