LAB 3 - (1)Qns - removing_duplicates

removing_duplicates

Bob works as a software engineer at a company that is rapidly expanding its workforce. The HR department has been using a software tool to manage employee records, which includes information employee IDs. However, due to some data integration issues, there are duplicate entries in the employee records, causing confusion and inefficiency.

Bob's task is to develop a program that takes a list of employee IDs as input. Each employee ID corresponds to a unique employee in the company. The program should then construct doubly linked list of employee IDs, removing any duplicate IDs.

Input Format

1. First Line consist of a number N showing the number of entries in the record.
2. Second Line consist of N space-separated sorted list of employee IDs, where each employee ID is a positive integer.

Constraints

N<100

Output Format

The program will print the space separated list of employes without duplicates.

Sample Input 0

```
5 
1 1 2 2 3
```

Sample Output 0

```
1 2 3
```

Sample Input 1

```
9 
4 5 5 6 6 6 7 8 8
```

Sample Output 1

`4 5 6 7 8`

💡 answer :

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

struct Node {
    int value;
    struct Node* next;
    struct Node* prev;
};

struct Node* createNode(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->value = value;
    newNode->next = NULL;
    newNode->prev = NULL;
    return newNode;
}

struct Node* createSortedDoublyLinkedList(int input[], int n) {
    if (n == 0) {
        return NULL;
    }

    struct Node* head = createNode(input[0]);
    struct Node* tail = head;

    for (int i = 1; i < n; i++) {
        struct Node* newNode = createNode(input[i]);
        tail->next = newNode;
        newNode->prev = tail;
        tail = newNode;
    }

    return head;
}

struct Node* deleteDuplicates(struct Node* head) {
    struct Node* current = head;

    while (current != NULL && current->next != NULL) {
        if (current->value == current->next->value) {
            struct Node* duplicate = current->next;
            current->next = duplicate->next;

            if (duplicate->next != NULL) {
                duplicate->next->prev = current;
            }
            free(duplicate);
        } else {
            current = current->next;
        }
    }

    return head;
}

void printDoublyLinkedList(struct Node* head) {
    struct Node* current = head;
    while (current) {
        printf("%d ", current->value);
        current = current->next;
    }
}

int main() {
    int n;

    scanf("%d", &n);

    int input[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &input[i]);
    }

    struct Node* head = createSortedDoublyLinkedList(input, n);

    struct Node* newHead = deleteDuplicates(head);

    printDoublyLinkedList(newHead);

    return 0;
}
```