Rajalakshmi Engineering College

Name: Harsha Vardhini S
Email: 240701180@rajalakshmi.edu.in
Roll no: 240701180
Phone: 9787756112
Branch: REC
Department: I CSE AH
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Moniksha, a chess coach organizing a tournament, needs a program to manage participant IDs efficiently. The program maintains a doubly linked list of IDs and offers two functions: Append to add IDs as students register, and Print Maximum ID to identify the highest ID for administrative tasks.

This tool streamlines tournament organization, allowing Moniksha to focus on coaching her students effectively.

Input Format

The first line consists of an integer n, representing the number of participant IDs to be added.

The second line consists of n space-separated integers representing the participant IDs.

Output Format

The output displays a single integer, representing the maximum participant ID.

If the list is empty, the output prints "Empty list!".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
163 137 155
Output: 163

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>

// Structure for Doubly Linked List Node
struct Node {
    int participantID;
    struct Node* prev;
    struct Node* next;
};

// Structure for Doubly Linked List
struct DoublyLinkedList {
    struct Node* head;
};

// Function to initialize the doubly linked list
void initList(struct DoublyLinkedList* list) {
    list->head = NULL;
}

// Function to append a node at the end of the list
void append(struct DoublyLinkedList* list, int participantID) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->participantID = participantID;
```

```
    newNode->next = NULL;

    if (list->head == NULL) {
        newNode->prev = NULL;
        list->head = newNode;
    } else {
        struct Node* temp = list->head;
        while (temp->next != NULL) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}

// Function to find the maximum participant ID in the list
int findMaxID(struct DoublyLinkedList* list) {
    if (list->head == NULL) {
        return -1;  // Return -1 to indicate empty list
    }

    int maxID = list->head->participantID;
    struct Node* temp = list->head;
    while (temp != NULL) {
        if (temp->participantID > maxID) {
            maxID = temp->participantID;
        }
        temp = temp->next;
    }
    return maxID;
}

int main() {
    int n;
    scanf("%d", &n);

    struct DoublyLinkedList list;
    initList(&list);

    if (n == 0) {
        printf("Empty list!\n");
        return 0;
```

```
    }

    int participantID;
    for (int i = 0; i < n; i++) {
        scanf("%d", &participantID);
        append(&list, participantID);
    }

    int maxID = findMaxID(&list);
    if (maxID == -1) {
        printf("Empty list!\n");
    } else {
        printf("%d\n", maxID);
    }

    return 0;
}
```

Status : Correct **Marks : 10/10**