Rajalakshmi Engineering College

Name: Vikashini M
Email: 241801314@rajalakshmi.edu.in
Roll no: 241801314
Phone: 9345747519
Branch: REC
Department: I AI & DS AF
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Ravi is developing a student registration system for a college. To efficiently store and manage the student IDs, he decides to implement a doubly linked list where each node represents a student's ID.

In this system, each student's ID is stored sequentially, and the system needs to display all registered student IDs in the order they were entered.

Implement a program that creates a doubly linked list, inserts student IDs, and displays them in the same order.

Input Format

The first line contains an integer N the number of student IDs.

The second line contains N space-separated integers representing the student IDs.

Output Format

The output should display the single line containing N space-separated integers representing the student IDs stored in the doubly linked list.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 20 30 40 50
Output: 10 20 30 40 50

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
struct Node{
    int studentID;
    struct Node* prev;
    struct Node* next;
};
struct DoublyLinkedList{
    struct Node* head;
    struct Node* tail;
};
struct Node* createNode(int studentID){
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if(newNode == NULL){
        printf("Memory allocation failed\n");
        exit(1);
   }
   newNode->studentID=studentID;
   newNode->prev=NULL;
   newNode->next=NULL;
   return newNode;
}
struct DoublyLinkedList* createDoublyLinkedList(){
    struct DoublyLinkedList*list = (struct DoublyLinkedList*)malloc(sizeof(struct
DoublyLinkedList));
    if(list==NULL){
        printf("Memory allocation failed\n");
        exit(1);
    }
    list->head =NULL;
    list->tail = NULL;
    return list;
}
void append(struct DoublyLinkedList* list,int studentID){
    struct Node* newNode = createNode(studentID);
    if(list->head == NULL){
        list->head = newNode;
        list->tail=newNode;
    }else{
        list->tail->next = newNode;
        newNode->prev = list->tail;
        list->tail=newNode;
    }
}
void displayList(struct DoublyLinkedList* list){
    struct Node* current = list->head;
    while(current != NULL){
        printf("%d ",current->studentID);
        current=current->next;
    }
    printf("\n");
}
int main(){
    int numStudents;
    scanf("%d",&numStudents);
    struct DoublyLinkedList* studentList = createDoublyLinkedList();
    for(int i=0;i<numStudents;i++){
        int studentID;
        scanf("%d",&studentID);
        append(studentList,studentID);
    }
    displayList(studentList);
    struct Node* current=studentList->head;
    while(current != NULL){
        struct Node* temp=current;
        current=current->next;
        free(temp);
    }
    free(studentList);
    return 0;
}
```

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