Rajalakshmi Engineering College

Name: magesh v
Email: 240801187@rajalakshmi.edu.in
Roll no: 240801187
Phone: 6381572897
Branch: REC
Department: I ECE FB
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Your task is to create a program to manage a playlist of items. Each item is represented as a character, and you need to implement the following operations on the playlist.

Here are the main functionalities of the program:

Insert Item: The program should allow users to add items to the front and end of the playlist. Items are represented as characters.Display Playlist: The program should display the playlist containing the items that were added.

To implement this program, a doubly linked list data structure should be used, where each node contains an item character.

Input Format

The input consists of a sequence of space-separated characters, representing the items to be inserted into the doubly linked list.

The input is terminated by entering - (hyphen).

Output Format

The first line of output prints "Forward Playlist: " followed by the linked list after inserting the items at the end.

The second line prints "Backward Playlist: " followed by the linked list after inserting the items at the front.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: a b c -

Output: Forward Playlist: a b c
Backward Playlist: c b a

Answer

-

Status : Skipped ***Marks : 0/10***

Rajalakshmi Engineering College

Name: magesh v
Email: 240801187@rajalakshmi.edu.in
Roll no: 240801187
Phone: 6381572897
Branch: REC
Department: I ECE FB
Batch: 2028
Degree: B.E - ECE

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>
#include <limits.h>

typedef struct Node {
   int data;

   struct Node *prev;
   struct Node *next;
} Node;

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

void append(Node **head, int data) {
   Node *newNode = createNode(data);
   if (*head == NULL)
   {
      *head = newNode;
```

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

void printMaxID(Node *head)
{
    if (head == NULL) {
        printf("Empty list!\n");
        return;
    }

    int maxID = INT_MIN;
    Node *temp = head;
    while (temp != NULL) {
        if (temp->data > maxID) {
            maxID = temp->data;
        }
        temp = temp->next;
    }
    printf("%d\n", maxID);
}

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

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

    Node *head = NULL;
    for (int i = 0; i < n; i++) {
        int id;
        scanf("%d", &id);
        append(&head, id);
    }

    printMaxID(head);

    Node *temp;
    while (head != NULL) {
        temp = head;
        head = head->next;
        free(temp);
    }

    return 0;
}
```

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