Rajalakshmi Engineering College

Name: KEERTHI PRIYA T
Email: 240701258@rajalakshmi.edu.in
Roll no: 2116240701258
Phone: 7397397221
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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

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

struct Node {
    char item;
    struct Node* next;
    struct Node* prev;
};

// You are using GCC
void insertAtEnd(struct Node** head, char item) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode ->item=item;
   newNode->next=NULL;
   newNode->prev=NULL;
   if (*head==NULL)
   {
       *head = newNode;
       return;
   }
   struct Node* temp= *head;
   while (temp-> next!=NULL)
   {
      temp = temp -> next;
   }
   temp-> next= newNode;
   newNode-> prev = temp;

}
void displayForward(struct Node* head) {
   printf("Forward playlist:");
   struct Node* temp=head;
   while (temp!= NULL)
   {
      printf("%c", temp->item);
      if (temp-> next == NULL) break;
      temp = temp-> next;
   }
   printf("\n");
}

void displayBackward(struct Node* tail) {
   printf("Backward Playlist:");
   while (tail-> next !=NULL)
      tail=tail->next;
   while (tail != NULL)
   {
      printf("%c",tail->item);
      tail=tail->prev;

   }
   printf("\n");
}

void freePlaylist(struct Node* head) {
   struct Node* temp ;
   while (head!= NULL)
   {
      temp = head;
      head = head -> next;
      free(temp);
```

```
    }
}

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

// Insert at end
void insertEnd(Node** head, char data) {
    struct Node* newNode = (struct Node*)malloc (sizeof(struct Node));
    newNode->data= item ;
    newNode->next = NULL;
    newNode->prev =NULL;
    if (*head == NULL) {
        *head = newNode;
        return;
    }
    struct Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;
    temp->next = newNode;
    newNode->prev = temp;
}

// Insert at front
void insertFront(Node** head, char data) {
    struct Node* newNode = createNode(data);
    if (*head != NULL) {
        newNode->next = *head;
        (*head)->prev = newNode;
    }
    *head = newNode;
}

// Print playlist
void printList(Node* head) {
    struct Node* temp = head;
    while (temp != NULL) {
        printf("%c ", temp->item);
        temp = temp->next;
    }
}

int main() {
    struct Node* playlist = NULL;
    char item;

    while (1) {
        scanf(" %c", &item);
        if (item == '-') {
            break;
        }
        insertAtEnd(&playlist, item);
    }

    struct Node* tail = playlist;
    while (tail->next != NULL) {
        tail = tail->next;
    }

    printf("Forward Playlist: ");
    displayForward(playlist);

    printf("Backward Playlist: ");
    displayBackward(tail);

    freePlaylist(playlist);

    return 0;
}
```

Status : Wrong ***Marks : 0/10***