Rajalakshmi Engineering College

Name: Nachiyappan Muthuraman
Email: 240801211@rajalakshmi.edu.in
Roll no: 240801211
Phone: 6381571350
Branch: REC
Department: I ECE AF
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 : 10

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;
};
```

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

// Define a node in the doubly linked list
typedef struct Node {
    char data;
    struct Node* prev;
    struct Node* next;
} Node;
```

```
// Function to create a new node
Node* createNode(char data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->prev = newNode->next = NULL;
    return newNode;
}

// Function to insert at the end
void insertAtEnd(Node** head, char data) {
    Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }
    Node* temp = *head;
    while (temp->next)
        temp = temp->next;
    temp->next = newNode;
    newNode->prev = temp;
}

// Function to insert at the front
void insertAtFront(Node** head, char data) {
    Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }
    newNode->next = *head;
    (*head)->prev = newNode;
    *head = newNode;
}

// Function to display the list
void displayList(Node* head) {
    while (head) {
        printf("%c ", head->data);
        head = head->next;
    }
    printf("\n");
}
```

240801211

```
int main() {
    Node* forwardPlaylist = NULL;
    Node* backwardPlaylist = NULL;

    char ch;
    while (1) {
        scanf(" %c", &ch);
        if (ch == '-')
            break;
        if (ch >= 'a' && ch <= 'z') {
            insertAtEnd(&forwardPlaylist, ch);
            insertAtFront(&backwardPlaylist, ch);
        }
    }

    printf("Forward Playlist: ");
    displayList(forwardPlaylist);

    printf("Backward Playlist: ");
    displayList(backwardPlaylist);

    return 0;
}
```

```
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: ");
```

240801211

```
    displayForward(playlist);

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

    freePlaylist(playlist);

    return 0;
}
```

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

240801211