Rajalakshmi Engineering College

Name: Praveen Kumar
Email: 240801247@rajalakshmi.edu.in
Roll no: 240801247
Phone: 7550385160
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 4_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Imagine a bustling coffee shop, where customers are placing their orders for their favorite coffee drinks. The cafe owner Sheeren wants to efficiently manage the queue of coffee orders using a digital system. She needs a program to handle this queue of orders.

You are tasked with creating a program that implements a queue for coffee orders. Each character in the queue represents a customer's coffee order, with 'L' indicating a latte, 'E' indicating an espresso, 'M' indicating a macchiato, 'O' indicating an iced coffee, and 'N' indicating a nabob.

Customers can place orders and enjoy their delicious coffee drinks.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Enqueue the coffee order into the queue. If the choice is 1, the following input is a space-separated character ('L', 'E', 'M', 'O', 'N').

Choice 2: Dequeue a coffee order from the queue.

Choice 3: Display the orders in the queue.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the queue:

If the choice is 1:

1. Insert the given order into the queue and display "Order for [order] is enqueued." where [order] is the coffee order that is inserted.
2. If the queue is full, print "Queue is full. Cannot enqueue more orders."

If the choice is 2:

1. Dequeue a character from the queue and display "Dequeued Order: " followed by the corresponding order that is dequeued.
2. If the queue is empty without any orders, print "No orders in the queue."

If the choice is 3:

1. The output prints "Orders in the queue are: " followed by the space-separated orders present in the queue.
2. If there are no orders in the queue, print "Queue is empty. No orders available."

If the choice is 4:

1. Exit the program and print "Exiting program"

If any other choice is entered, the output prints "Invalid option."

Refer to the sample output for the exact text and format.

Sample Test Case

Input: 1 L
1 E
1 M
1 O
1 N
1 O
3
2
3
4
Output: Order for L is enqueued.
Order for E is enqueued.
Order for M is enqueued.
Order for O is enqueued.
Order for N is enqueued.
Queue is full. Cannot enqueue more orders.
Orders in the queue are: L E M O N
Dequeued Order: L
Orders in the queue are: E M O N
Exiting program

Answer

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

typedef struct Queue {
   char data;
   struct Queue* next;
}node;

int length(node* head) {
   int len = 0;
   while(head != NULL) {
      len++;
      head = head->next;
```

```
    }
    return len;
}

void enqueue(node** head, node** tail, char data) {
    node* newnode = (node*) malloc(sizeof(node));
    newnode->data = data;
    newnode->next = NULL;

    if(*head == NULL) {
        *head = newnode;
        *tail = newnode;
        return;
    }

    (*tail)->next = newnode;
    *tail = newnode;
}

char dequeue(node** head, node** tail) {
    if(*head == NULL) {
        printf("No orders in the queue.\n");
        return NULL;
    }
    if(*head == *tail) {
        *tail = NULL;
    }

    char data = (*head)->data;
    node* temp = *head;
    *head = (*head)->next;
    free(temp);
    return data;
}

void display(node* head) {
    if(head == NULL) {
        printf("Queue is empty. No orders available.\n");
        return;
    }
    printf("Orders in the queue are: ");
    while(head != NULL) {
        printf("%c ", head->data);
        head = head->next;
    }
    printf("\n");
}

int main() {
    int n;
    char m;
    node* head = NULL;
    node* tail = NULL;

    input:
        scanf("%d", &n);

    if(n == 1) {
        scanf("%c", &m);
        if(length(head) >= 5 && (int)m != 32 && m != '\n') {
            printf("Queue is full. Cannot enqueue more orders.\n");
        }
        else if (m != ' ' && m != '\n')   {
            printf("Order for %c is enqueued.\n", m);
            enqueue(&head, &tail, m);
        }
    }
    else if(n == 2) {
        m = dequeue(&head, &tail);
        if(m != NULL) {
            printf("Dequeued Order: %c\n", m);
        }
    }
    else if(n == 3) {
        display(head);
    }
    else if(n == 4) {
        printf("Exiting program");
        return 0;
    }
    else {
        printf("Invalid option.\n");
    }
    goto input;
    node* temp;
    while(head != NULL) {
        temp = head;
        head = head->next;
        free(temp);
    }
    head = tail = NULL;

}
```

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