

# Rajalakshmi Engineering College

Name: suriya SM

Email: 241801284@rajalakshmi.edu.in

Roll no: 241801284

Phone: 8110855156

Branch: REC

Department: I AI & DS FD

Batch: 2028

Degree: B.E - AI & DS

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>
```

```
#define MAX_SIZE 5
```

```
char orders[MAX_SIZE];
```

```
int front = -1;
```

```
int rear = -1;
```

```
void initializeQueue() {
```

```
    front = -1;
```

```
    rear = -1;
```

```
}
```

```
// You are using GCC
```

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

```
#define MAX 5
```

```
char queue[MAX];
int front = -1, rear = -1;
```

```
int isFull() {
    return rear == MAX - 1;
}
```

```
int isEmpty() {
    return front == -1 || front > rear;
}
```

```
void enqueue(char order) {
    if (isFull()) {
        printf("Queue is full. Cannot enqueue more orders.\n");
        return;
    }
    if (isEmpty()) {
        front = rear = 0;
    } else {
        rear++;
    }
    queue[rear] = order;
    printf("Order for %c is enqueued.\n", order);
}
```

```
void dequeue() {
    if (isEmpty()) {
        printf("No orders in the queue.\n");
        return;
    }
    printf("Dequeued Order: %c\n", queue[front]);
    front++;
    if (front > rear) {
```

```

    front = rear = -1;
}
}

void display() {
    if (isEmpty()) {
        printf("Queue is empty. No orders available.\n");
        return;
    }
    printf("Orders in the queue are: ");
    for (int i = front; i <= rear; i++) {
        printf("%c ", queue[i]);
    }
    printf("\n");
}

int main() {
    int choice;
    char order;

```

```

    while (scanf("%d", &choice) != EOF) {
        switch (choice) {
            case 1:
                scanf(" %c", &order);
                if (order == 'L' || order == 'E' || order == 'M' || order == 'O' || order == 'N') {
                    enqueue(order);
                } else {
                    printf("Invalid coffee order.\n");
                }
                break;

            case 2:
                dequeue();
                break;

            case 3:
                display();
                break;

            case 4:
                printf("Exiting program\n");

```

```

        return 0;

    default:
        printf("Invalid option.\n");
    }
}

return 0;
}

int main() {
    char order;
    int option;
    initializeQueue();
    while (1) {
        if (scanf("%d", &option) != 1) {
            break;
        }
        switch (option) {
            case 1:
                if (scanf(" %c", &order) != 1) {
                    break;
                }
                if (enqueue(order)) {
                }
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                printf("Exiting program");
                return 0;
            default:
                printf("Invalid option.\n");
                break;
        }
    }
    return 0;
}

```

**Status :** Correct

**Marks :** 10/10