ASSIGNMENT NO.8

NAME: Payal Bhagvan Charvande

CLASS: S.Y. **DIV:** IT A

PRN NO: 125B2F002

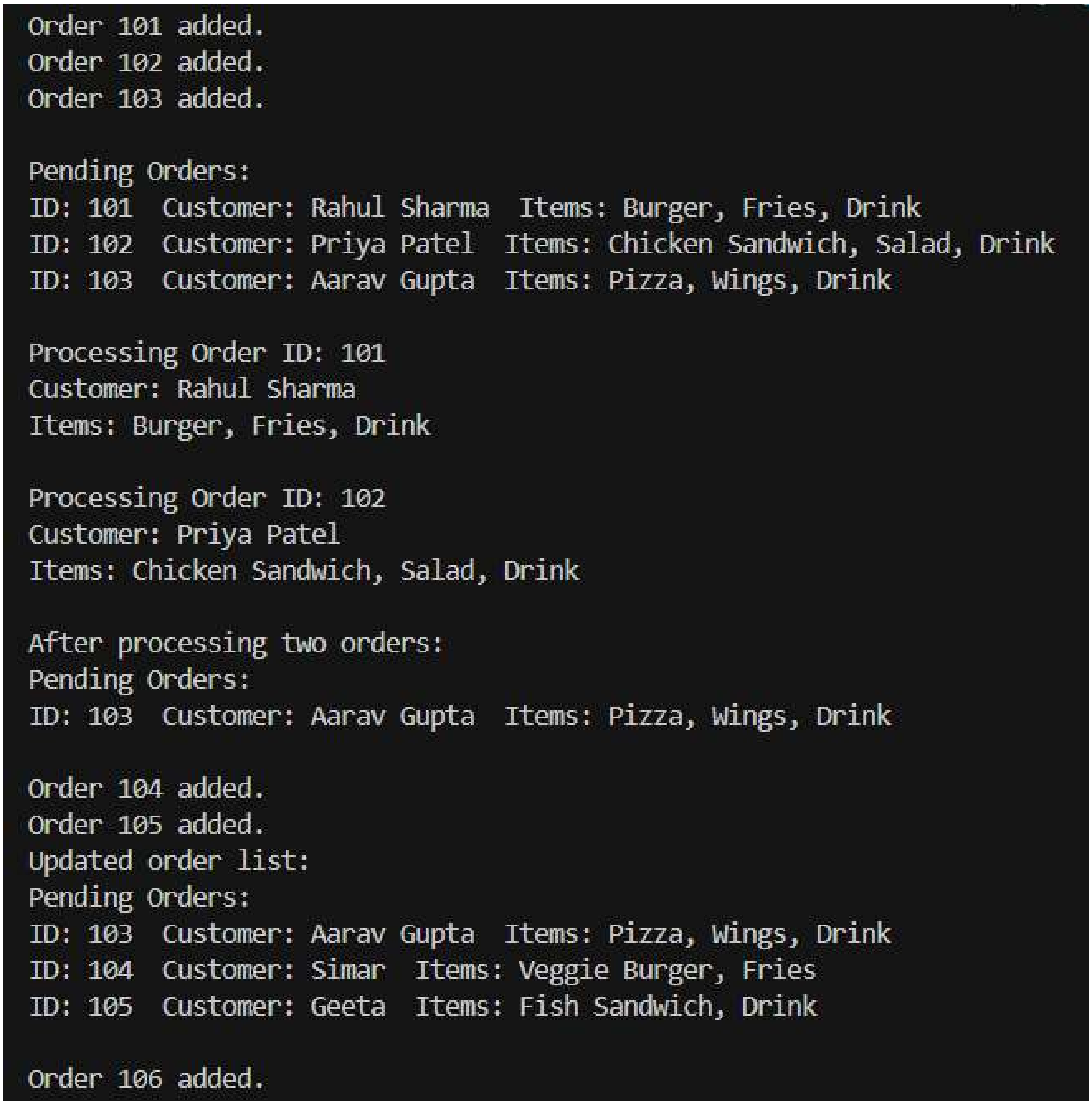
A)

CODE:-

```
#include <iostream>
using namespace std;

const int MAX = 5;

struct Order {
   int id;
   string items;
   string customer;
};
Order queue[MAX];
int front = -1, rear = -1;

// Check if queue is full
bool isFull() {
   return (front == 0 && rear == MAX - 1) || (rear + 1 == front);
}
// Check if queue is empty
bool isEmpty() {
   return front == -1;
}
// Add new order to queue
```

```
void addOrder(int id, string items, string customer) {
    if (isFull()) {
        cout << " Queue full! Can't add order " << id << "\n";
        return;
    }
    if (isEmpty()) {
        front = 0;
        rear = 0;
    } else {
        rear = (rear + 1) % MAX;
    }
    queue[rear].id = id;
    queue[rear].items = items;
    queue[rear].customer = customer;
    cout << "Order " << id << " added.\n";
}
// Process the order at front of queue
void processOrder() {
    if (isEmpty()) {
        cout << "No orders to process.\n";
        return;
    }
    cout << "Processing Order ID: " << queue[front].id << "\n";
    cout << "Customer: " << queue[front].customer << "\n";
    cout << "Items: " << queue[front].items << "\n\n";
    if (front == rear) {
        // Only one order, reset queue
        front = -1;
        rear = -1;
    } else {
        front = (front + 1) % MAX;
    }
}
// Display all current orders in queue
void displayOrders() {
    if (isEmpty()) {
        cout << "No pending orders.\n";
        return;
    }
    cout << "Pending Orders:\n";

    int i = front;
    while (true) {
        cout << "ID: " << queue[i].id << "  Customer: " << queue[i].customer
             << "  Items: " << queue[i].items << "\n";
        if (i == rear) break;
        i = (i + 1) % MAX;
    }
    cout << endl;
}
int main() {
    // Add initial orders
    addOrder(101, "Burger, Fries, Drink", "Rahul Sharma");
    addOrder(102, "Chicken Sandwich, Salad, Drink", "Priya Patel");
    addOrder(103, "Pizza, Wings, Drink", "Aarav Gupta");
    cout << "\n";

    displayOrders();
    // Process two orders
    processOrder();
    processOrder();

    cout << "After processing two orders:\n";
    displayOrders();

    // Add more orders
    addOrder(104, "Veggie Burger, Fries", "Simar");
    addOrder(105, "Fish Sandwich, Drink", "Geeta");

    cout << "Updated order list:\n";
    displayOrders();

    // Try adding order when full
    addOrder(106, "Maggi", "Neha ");

    return 0;
}
```

OUTPUT:-



```
Order 101 added.
Order 102 added.
Order 103 added.

Pending Orders:
ID: 101  Customer: Rahul Sharma  Items: Burger, Fries, Drink
ID: 102  Customer: Priya Patel  Items: Chicken Sandwich, Salad, Drink
ID: 103  Customer: Aarav Gupta  Items: Pizza, Wings, Drink

Processing Order ID: 101
Customer: Rahul Sharma
Items: Burger, Fries, Drink

Processing Order ID: 102
Customer: Priya Patel
Items: Chicken Sandwich, Salad, Drink

After processing two orders:
Pending Orders:
ID: 103  Customer: Aarav Gupta  Items: Pizza, Wings, Drink

Order 104 added.
Order 105 added.
Updated order list:
Pending Orders:
ID: 103  Customer: Aarav Gupta  Items: Pizza, Wings, Drink
ID: 104  Customer: Simar  Items: Veggie Burger, Fries
ID: 105  Customer: Geeta  Items: Fish Sandwich, Drink

Order 106 added.
```