K.R. MANGALAM UNIVERSITY
THE COMPLETE WORLD OF EDUCATION

NAME: BADAL PRASAD
ROLL NO:2501351020
COURSE: BTECH CSE(FSD)
SUBJECT: DSA
TOPIC: LAB ASSIGNMENT 2

Q The Customer Support Ticket System is a software solution designed to manage incoming customer tickets using queues, stacks, and linked lists. This system enables dynamic addition of tickets, prioritizes urgent tickets, supports undo operations, and processes.

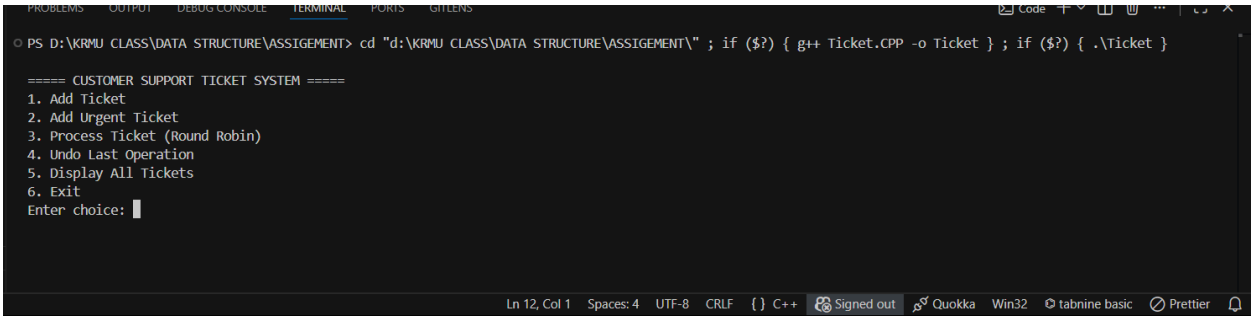
```
#include <iostream>
#include <string>
using namespace std;

//--------------------------------------
// Ticket Node (Linked List)
//--------------------------------------
struct Ticket {
    int id;
    string issue;
    Ticket* next;

    Ticket(int i, string iss) {
        id = i;
        issue = iss;
        next = nullptr;
    }
};

//--------------------------------------
// Queue class for normal tickets
//--------------------------------------
class TicketQueue {
public:
    Ticket* front;
    Ticket* rear;
```

```
    TicketQueue() {
        front = rear = nullptr;
    }

    // enqueue normal ticket
    void enqueue(int id, string issue) {
        Ticket* temp = new Ticket(id, issue);
        if (!rear) {
            front = rear = temp;
            return;
        }
        rear->next = temp;
        rear = temp;
    }

    // dequeue ticket
    Ticket* dequeue() {
        if (!front) return nullptr;

        Ticket* temp = front;
        front = front->next;

        if (!front) rear = nullptr;
        return temp;
    }

    // display queue
    void display() {
        Ticket* temp = front;
        while (temp) {
            cout << "[ID: " << temp->id << "] Issue: " << temp->issue << endl;
            temp = temp->next;
        }
    }
};

//-------------------------------------
// Stack class for UNDO
//-------------------------------------
```

```
class UndoStack {
public:
    Ticket* top;

    UndoStack() {
        top = nullptr;
    }

    void push(Ticket* t) {
        t->next = top;
        top = t;
    }

    Ticket* pop() {
        if (!top) return nullptr;
        Ticket* temp = top;
        top = top->next;
        return temp;
    }
};

//---------------------------------------
// MAIN SYSTEM
//---------------------------------------
int main() {
    TicketQueue normalQueue;
    TicketQueue urgentQueue;
    UndoStack undo;

    int choice;
    int id = 1;

    while (true) {
        cout << "\n===== CUSTOMER SUPPORT TICKET SYSTEM =====\n";
        cout << "1. Add Ticket\n";
        cout << "2. Add Urgent Ticket\n";
        cout << "3. Process Ticket (Round Robin)\n";
        cout << "4. Undo Last Operation\n";
        cout << "5. Display All Tickets\n";
        cout << "6. Exit\n";
```

```
cout << "Enter choice: ";
cin >> choice;

if (choice == 1) {
    string issue;
    cout << "Enter issue: ";
    cin.ignore();
    getline(cin, issue);

    normalQueue.enqueue(id, issue);
    cout << "Ticket Added (Normal). ID = " << id << endl;

    undo.push(new Ticket(id, issue));
    id++;
}

else if (choice == 2) {
    string issue;
    cout << "Enter urgent issue: ";
    cin.ignore();
    getline(cin, issue);

    urgentQueue.enqueue(id, issue);
    cout << "Urgent Ticket Added. ID = " << id << endl;

    undo.push(new Ticket(id, issue));
    id++;
}

else if (choice == 3) {
    Ticket* t = nullptr;

    // Round Robin: urgent tickets first
    if (urgentQueue.front)
        t = urgentQueue.dequeue();
    else
        t = normalQueue.dequeue();

    if (t)
        cout << "Processing Ticket ID: " << t->id << " | Issue: " << t->issue << endl;
```

```
            else
                cout << "No tickets to process!\n";
        }

        else if (choice == 4) {
            Ticket* undone = undo.pop();
            if (!undone) {
                cout << "Nothing to undo!\n";
            } else {
                cout << "Undo: Last added ticket removed (ID = " << undone->id << ")\n";
            }
        }

        else if (choice == 5) {
            cout << "\n--- Urgent Tickets ---\n";
            urgentQueue.display();

            cout << "\n--- Normal Tickets ---\n";
            normalQueue.display();
        }

        else if (choice == 6) {
            cout << "Exiting System...\n";
            break;
        }

        else {
            cout << "Invalid choice!\n";
        }
    }

    return 0;
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
Code
PS D:\KRMU CLASS\DATA STRUCTURE\ASSIGEMENT> cd "d:\KRMU CLASS\DATA STRUCTURE\ASSIGEMENT\" ; if ($?) { g++ Ticket.CPP -o Ticket } ; if ($?) { .\Ticket }
===== CUSTOMER SUPPORT TICKET SYSTEM =====
1. Add Ticket
2. Add Urgent Ticket
3. Process Ticket (Round Robin)
4. Undo Last Operation
5. Display All Tickets
6. Exit
Enter choice:
Ln 12, Col 1
Spaces: 4
UTF-8
CRLF
{} C++
Signed out
Quokka
Win32
tabnine basic
Prettier