Depamaylo, Ma. Samantha Elizabeth M.
BSITWMA – AW12

Code

```
#include <iostream> // Input and output operations
#include <iomanip> // Manipulating input and output
#include <windows.h> // Windows-specific functions
using namespace std;

// Node class representing each customer request as a node in the queue
class Node {
public:
    int data; // Data stored in the node, representing the ticket number
    Node* next; // Pointer to the next node in the queue

    // Constructor to initialize a node with a given value
    Node(int value) {
        data = value;
        next = NULL;
    }
};

// Queue class implementing a queue using linked list nodes
class Queue {
private:
    Node* front; // Pointer to the front node of the queue
    Node* rear; // Pointer to the rear node of the queue

    // Function to set the console text color based on the ticket number
    void setConsoleColor(int value) {
        HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
        switch (value) {
            case 1:
                SetConsoleTextAttribute(hConsole, 9); // Blue
                break;
            case 2:
                SetConsoleTextAttribute(hConsole, 12); // Red
                break;
            case 3:
                SetConsoleTextAttribute(hConsole, 14); // Yellow
                break;
            case 4:
                SetConsoleTextAttribute(hConsole, 13); // Purple
                break;
            case 5:
                SetConsoleTextAttribute(hConsole, 11); // Mint
                break;
            default:
                SetConsoleTextAttribute(hConsole, 7); // Default color
                break;
        }
    }

    // Function to reset the console text color to default
    void resetConsoleColor() {
        setConsoleColor(0);
    }

public:
    // Constructor to initialize an empty queue
    Queue() {
        front = NULL;
        rear = NULL;
    }

    // Function to check if the queue is empty
    bool isEmpty() {
        return front == NULL;
    }

    // Function to add a customer request (number) to the queue
    void enqueue(int value) {
        Node* newNode = new Node(value); // Create a new node with the given value
        if (isEmpty()) {
            front = rear = newNode; // If the queue is empty, both front and rear point to the new node
        } else {
            rear->next = newNode; // Add the new node at the end of the queue
            rear = newNode; // Update the rear pointer to the new node
        }
        cout << "\nPatient " << setfill('0') << setw(3) << value << " arrives." << endl;
        cout << "\n--------------------------------\n";
        cout << "\nTicket " << setfill('0') << setw(3) << value << " is issued." << endl;
        setConsoleColor(value);
        cout << "Ticket " << setfill('0') << setw(3) << value << " added to the queue." << endl;
        resetConsoleColor();
        cout << "\n--------------------------------\n";
        display();
        Sleep(1000);
    }
```

Depamaylo, Ma. Samantha Elizabeth M.
BSITWMA – AW12

```
    // Function to remove and serve the customer request at the front of the queue
    void dequeue() {
        if (isEmpty()) {
            cout << "No patient to serve." << endl;
        } else {
            display();
            cout << "\n--------------------------------\n";
            cout << "\nServing patient ";
            for (int i = 0; i < 3; ++i) {
                cout << ".";
                cout.flush();
                Sleep(500);
            }
            cout << endl;

            Node* temp = front; // Store the front node
            front = front->next; // Update the front pointer to the next node

            cout << "\n--------------------------------\n";
            cout << "\nServing patient ";
            setConsoleColor(temp->data);
            cout << setfill('0') << setw(3) << temp->data;
            resetConsoleColor();
            cout << endl;
            cout << "\n--------------------------------\n";

            delete temp; // Delete the old front node
            if (front == NULL) {
                rear = NULL; // If the queue is now empty, update the rear pointer
            }
            Sleep(1000);

            if (temp->data == 5 && isEmpty()) {
                cout << "\nQueue is empty!" << endl;
            }
        }
    }

    // Function to display the customer request currently being served
    int peek() {
        if (!isEmpty()) {
            return front->data; // Return the data of the front node
        } else {
            cout << "No patient to serve.\n" << endl;
            return -1; // Return -1 if the queue is empty
        }
    }

    // Function to display all customer requests in the queue
    void display() {
        if (isEmpty()) {
            cout << "Queue is empty!" << endl;
            cout << "--------------------------------\n \n";
            return;
        }

        cout << "\nCurrent queue:\n";
        Node* temp = front; // Start from the front node
        while (temp != NULL) {
            setConsoleColor(temp->data);
            cout << "Ticket " << setfill('0') << setw(3) << temp->data << endl;
            resetConsoleColor();
            temp = temp->next; // Move to the next node
        }
    }
};

// Function to display the menu options to the user
void displayMenu() {
    cout << "\n--------------------------------\n";
    cout << "   \n   Samantha's Medical Center\n\n";
    cout << "--------------------------------\n";
    cout << "1. Get Ticket Number\n";
    cout << "2. Start the system\n";
    cout << "3. Exit\n";
    cout << "--------------------------------\n";
}

int main() {
    Queue q; // Create a Queue object
    int choice, ticketNumber = 1; // Initialize choice and ticketNumber
    bool systemStarted = false; // Initialize systemStarted flag

    while (true) {
        if (!systemStarted) {
            displayMenu(); // Display the menu if the system is not started
        }
```

Depamaylo, Ma. Samantha Elizabeth M.
BSITWMA – AW12

```
        cout << "Enter your choice: ";
        cin >> choice;
        cout << "--------------------------------\n";

        switch (choice) {
            case 1:
                q.enqueue(ticketNumber++); // Get a ticket number and enqueue it
                break;
            case 2:
                systemStarted = true; // Start the system
                break;
            case 3:
                cout << "\nNo patient to serve.\n";
                return 0; // Exit the program
            default:
                cout << "Invalid choice! Please try again.\n";
                break;
        }

        if (systemStarted) {
            while (!q.isEmpty()) {
                q.dequeue(); // Serve all customers in the queue
            }
            systemStarted = false; // Reset the systemStarted flag
        }
    }

    return 0;
}
```