Assignment No: 06

Name: Prarthana Kumbhar
Class: SY-1
Batch: C
PRN: B25CE2014

Title: Coffee Shop Line (Simple Queue):
Arrival: Customers arrive at the coffee shop and stand in line. Order Processing: The first customer in line gets their order taken, and the barista starts making the coffee. Serving: Once the first customer is served, they leave the queue, and the next customer in line moves forward to be served. Write a program to implement a simple queue.

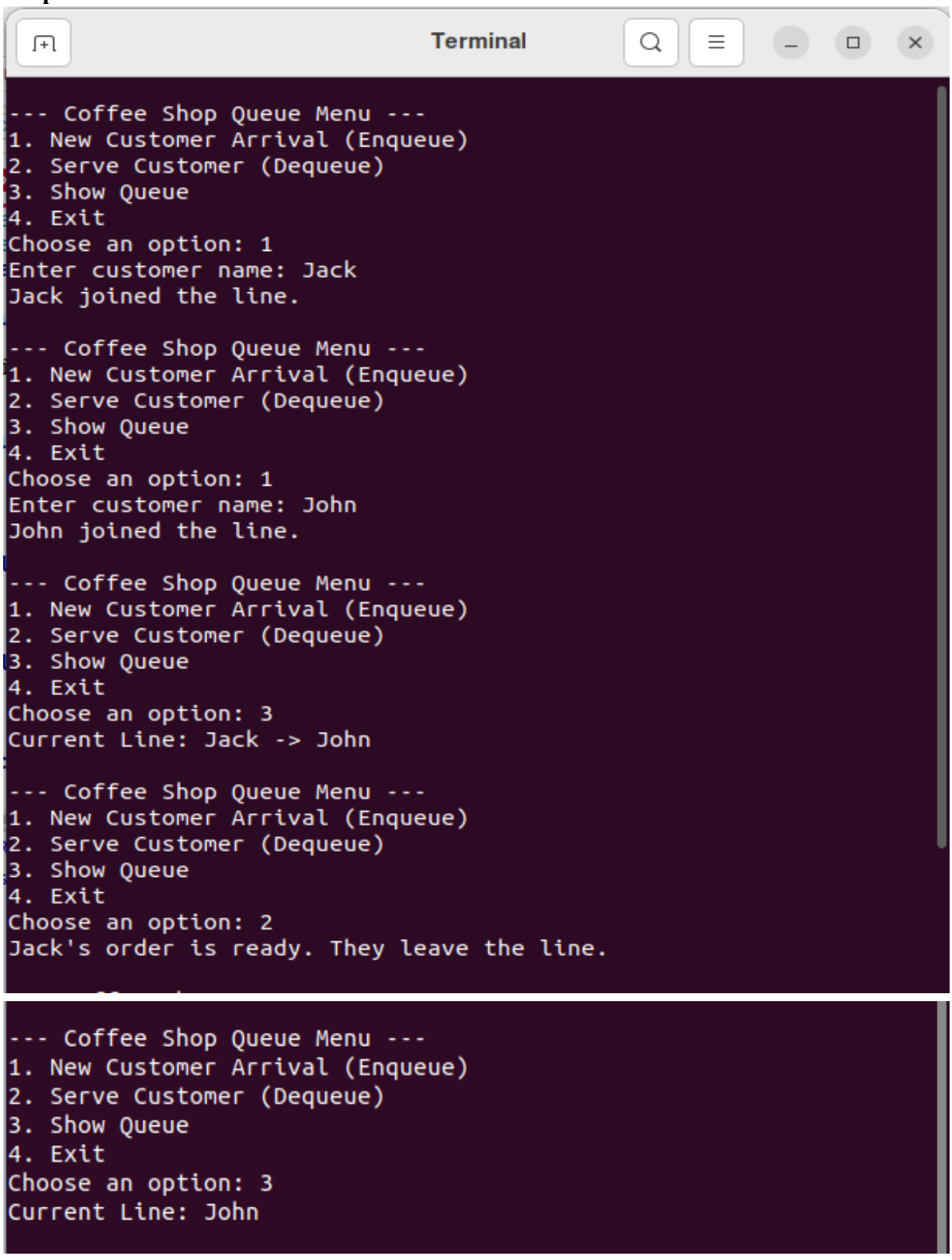
Program:

```
#include <iostream>
#include <string>
using namespace std;
#define MAX 10
class Queue
{
   string arr[MAX];
   int front, rear;
public:
   Queue()
   {
      front = 0;
      rear = -1;
   }
   bool isFull()
   {
      return (rear == MAX - 1);
   }
   bool isEmpty()
   {
      return (front > rear);
   }
   void enqueue(string name)
   {
      if (isFull())
      {
         cout << "Queue is full. Cannot add more customers.\n";
      }
      else
      {
         rear++;
         arr[rear] = name;
```

```
            cout << name << " joined the line.\n";
        }
    }
    void dequeue()
    {
        if (isEmpty())
        {
            cout << "No customers in line.\n";
        }
        else
        {
            cout << arr[front] << "'s order is ready. They leave the line.\n";
            front++;
        }
    }
    void display()
    {
        if (isEmpty())
        {
            cout << "The line is empty.\n";
        } else {
            cout << "Current Line: ";
            for (int i = front; i <= rear; i++)
            {
                cout << arr[i];
                if (i < rear) cout << " -> ";
            }
            cout << endl;
        }
    }
};
int main()
{
    Queue q;
    int choice;
    string name;
    do
    {
        cout << "\n--- Coffee Shop Queue Menu ---\n";
        cout << "1. New Customer Arrival (Enqueue)\n";
        cout << "2. Serve Customer (Dequeue)\n";
        cout << "3. Show Queue\n";
        cout << "4. Exit\n";
        cout << "Choose an option: ";
        cin >> choice;
        switch (choice)
```

```
        {
            case 1:
                cout << "Enter customer name: ";
                cin >> name;
                q.enqueue(name);
                break;
            case 2:
                q.dequeue();
                break;
            case 3:
                q.display();
                break;
            case 4:
                cout << "Exiting... Thank you!\n";
                break;
            default:
                cout << "Invalid option. Try again.\n";
        }
    } while (choice != 4);
    return 0;
}
```

Output:


Terminal
--- Coffee Shop Queue Menu ---
1. New Customer Arrival (Enqueue)
2. Serve Customer (Dequeue)
3. Show Queue
4. Exit
Choose an option: 1
Enter customer name: Jack
Jack joined the line.
--- Coffee Shop Queue Menu ---
1. New Customer Arrival (Enqueue)
2. Serve Customer (Dequeue)
3. Show Queue
4. Exit
Choose an option: 1
Enter customer name: John
John joined the line.
--- Coffee Shop Queue Menu ---
1. New Customer Arrival (Enqueue)
2. Serve Customer (Dequeue)
3. Show Queue
4. Exit
Choose an option: 3
Current Line: Jack -> John
--- Coffee Shop Queue Menu ---
1. New Customer Arrival (Enqueue)
2. Serve Customer (Dequeue)
3. Show Queue
4. Exit
Choose an option: 2
Jack's order is ready. They leave the line.
--- Coffee Shop Queue Menu ---
1. New Customer Arrival (Enqueue)
2. Serve Customer (Dequeue)
3. Show Queue
4. Exit
Choose an option: 3
Current Line: John

Title: Printer Spooler (Circular Queue):
In a multi-user environment, printers often use a circular queue to manage print jobs. Each print job is added to the queue, and the printer processes them in the order they arrive. Once a print job is completed, it moves out of the queue, and the next job is processed, efficiently managing the flow of print tasks. Implement the Printer Spooler system using a circular queue without using built-in queues.

Program:

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

#define SIZE 5

string jobs[SIZE];
int front = -1, rear = -1;

// Function to insert job
void insertJob() {
    if ((front == 0 && rear == SIZE - 1) || (front == rear + 1)) {
        cout << "\nSorry... Spooler is FULL!" << endl;
        return;
    }
    string job;
    cout << "\nEnter print job name: ";
    cin >> job;

    if (front == -1) front = 0;
    rear = (rear + 1) % SIZE;
    jobs[rear] = job;

    cout << "Print job \"" << job << "\" added to the spooler." << endl;
}

// Function to delete job
void deleteJob() {
    if (front == -1) {
        cout << "\nSorry... Spooler is EMPTY!" << endl;
        return;
    }
    cout << "Processing print job: \"" << jobs[front] << "\"" << endl;

    if (front == rear) {
```

```
        front = -1;
        rear = -1;
    } else {
        front = (front + 1) % SIZE;
    }
}

// Function to show jobs
void showJobs() {
    if (front == -1) {
        cout << "\nSorry... Spooler is EMPTY!" << endl;
        return;
    }
    cout << "\nCurrent Print Queue: ";
    int i = front;
    while (true) {
        cout << "\"" << jobs[i] << "\"";
        if (i == rear) break;
        cout << " -> ";
        i = (i + 1) % SIZE;
    }
    cout << endl;
}

int main() {
    int choice;

    while (1) {
        cout << "\n--- Printer Spooler Menu ---\n";
        cout << "1. Add Print Job\n";
        cout << "2. Process Print Job\n";
        cout << "3. Show All Print Jobs\n";
        cout << "4. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1: insertJob(); break;
            case 2: deleteJob(); break;
            case 3: showJobs(); break;
            case 4: exit(0);
            default: cout << "\nInvalid choice!" << endl;
        }
    }
```

```
    return 0;
}
```

Output: