PIMPRI CHINCHWAD EDUCATION TRUST's.
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
(An Autonomous Institute)

Class : SY BTech **Acad. Yr. 2025-26** **Semester : I**

Name of the student: Hariom Shrikrishna Gundale **PRN : 124B1B036**

Department: Computer Engineering **Division : A**

Course Name : Data Structures Laboratory Course

Code:BCE23PC02 **Completion Date :8 /10/2025**

Assignment No. 8

Problem Statement: Simulate a ticketing system where customers raise support tickets and are added to a queue. The support team dequeues and resolves tickets. Allow urgent issues to be placed at the front.Write a program for above scenario.

Source Code :

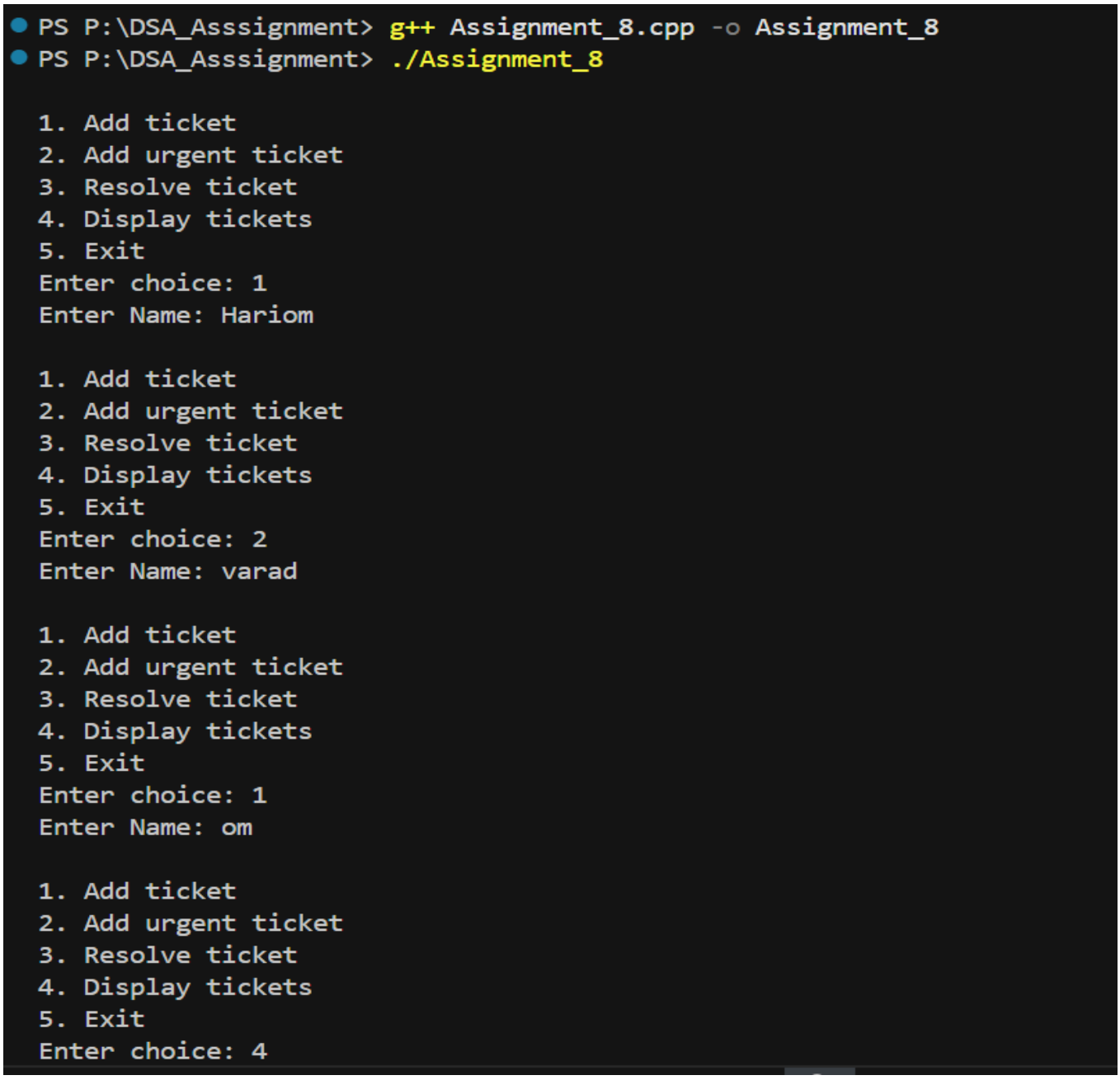
```
#include <bits/stdc++.h>
using namespace std;

class Qqueue
{
public:
   int ID;
   string customerName;
   Qqueue *next;

   Qqueue(string name, int id)
   {
      customerName = name;
      ID = id;
      next = NULL;
   }
};

class TicketingSystem
{
   Qqueue *front;
```

```
    Qqueue *rear;

public:
    TicketingSystem()
    {
        front = rear = NULL;
    }

    void enqueue(string name, int id)
    {
        Qqueue *newNode = new Qqueue(name, id);

        if (front == NULL)
        {
            front = rear = newNode;
            rear->next = front;
            return;
        }

        rear->next = newNode;
        rear = newNode;
        rear->next = front;
    }

    // Urgent Enqueue
    void urgentEnqueue(string name, int id)
    {
        Qqueue *newNode = new Qqueue(name, id);

        if (front == NULL)
        {
            front = rear = newNode;
            rear->next = front;
            return;
        }

        newNode->next = front;
        front = newNode;
        rear->next = front;
    }

    void dequeue()
    {
        if (front == NULL)
        {
            cout << "No tickets to resolve. ; Queue is empty \n";
            return;
        }

        Qqueue *temp = front;
```

```
        if (front == rear)
        {
            cout << "Resolving ticket: " << temp->ID << " " << temp->customerName << endl;
            delete temp;
            front = rear = NULL;
            return;
        }

        cout << "Resolving ticket: " << temp->ID << " " << temp->customerName << endl;

        front = front->next;
        rear->next = front;
        delete temp;
    }

    void display()
    {
        if (front == NULL)
        {
            cout << "Queue is empty\n";
            return;
        }

        Qqueue *temp = front;
        cout << "\n Tickets in Queue:\n";
        do
        {
            cout << "Ticket ID: " << temp->ID << " || Customer Name: " << temp->customerName << endl;
            temp = temp->next;
        } while (temp != front);
        cout << endl;
    }
};

int main()
{
    int GlobalID = 1;
    TicketingSystem ts;

    while (true)
    {
        int choice;
        cout << "\n1. Add ticket"
             << "\n2. Add urgent ticket"
             << "\n3. Resolve ticket"
             << "\n4. Display tickets"
             << "\n5. Exit\n"
             << "Enter choice: ";
        cin >> choice;
```

```
        switch (choice)
        {
        case 1:
        {
            string name;
            cout << "Enter Name: ";
            cin.ignore();
            getline(cin, name);
            ts.enqueue(name, GlobalID++);
            break;
        }
        case 2:
        {
            string name;
            cout << "Enter Name: ";
            cin.ignore();
            getline(cin, name);
            ts.urgentEnqueue(name, GlobalID++);
            break;
        }
        case 3:
            ts.dequeue();
            break;
        case 4:
            ts.display();
            break;
        case 5:
            return 0;
        default:
            cout << "Enter valid choice!\n";
        }
    }
}
```

Screen Shot of Output :



Conclusion:

Thus, we have successfully implemented the C++ program toSimulate a ticketing system where customers raise support tickets and are added to a queue. The support team dequeues and resolves tickets. Allow urgent issues to be placed at the front.