Module 6

Topic 1

```
#include <iostream>
using namespace std;
int n, queue[0];
void sizer()
{
    cout << "Enter your Queue Size : ";
    cin >> n;
    queue[n];
}
void menu()
{
    cout << "***Simple Circular Queue Menu***" << endl;
    cout << "1. Enqueue\n2. Dequeue\n3. Display\n4. Exit" << endl;
    cout << "Enter Your Option -> ";
}
class Queue
{
private:
    int rear = -1, front = -1;

public:
    void enqueue(int y)
    {
        if ((rear == n - 1 && front == 0) || (front == rear + 1))
        {
            printf("Queue is full.\n");
        }
        else
        {
            if (front == -1)
                front = 0;
            rear = (rear + 1) % n;
            queue[rear] = y;
        }
    }

    void dequeue()
    {
        int x, y = 0;
        if (front == -1)
            printf("Queue is empty..\n");
        else
        {
```

```
            if (front == rear)
            {
                y = queue[front];
                front = -1;
                rear = -1;
                cout << "Your Dequeued Value is :" << y << endl;
                cout << "his work (" << y << ") is done" << endl;
                cout << "Queue was emptied so it is initializing ...." << endl;
            }
            else
            {
                x = queue[front];
                front = (front + 1) % n;
                cout << "Your Dequeued Value is :" << x << endl;
                cout << "his work (" << x << ") is done" << endl;
            }
        }
    }
    void display()
    {
        int i = front;
        cout << "front = " << front << endl;
        cout << "rear  = " << rear << endl;
        cout << "| TC |-> ";
        if (front == -1)
        {
            cout << "Queue is empty..." << endl;
        }
        else
        {
            while (i != rear)
            {
                cout << "| " << queue[i] << "  ";
                i = (i + 1) % n;
            }
            cout << "| " << queue[i] << "  ";
            cout << "|" << endl;
        }
    }
};
int main()
{
    Queue test;
    int choice, r, x;
    while (choice != 4)
```

```
    {
        menu();
        cin >> choice;
        switch (choice)
        {
        case 1:
            if (n == 0)
                sizer();
            cout << "Enter your Enqueued Value : ";
            cin >> x;
            test.enqueue(x);
            break;
        case 2:
            test.dequeue();
            break;
        case 3:
            cout << "Your Displayed Queue is :" << endl;
            test.display();
            break;
        default:
            break;
        }
    }
}
```

Topic 2

```
#include <iostream>
using namespace std;
class quelink
{
public:
    int data;
    quelink *next;
};
class TicketCounter
{
private:
    quelink *front = NULL, *rear = NULL;

public:
    void enqueue(int x)
    {
        quelink *temp = new quelink;
        temp->data = x;
        temp->next = NULL;
```

```
        if (rear == NULL)
        {
            front = temp;
            rear = temp;
        }
        else
        {
            rear->next = temp;
            rear = temp;
        }
    }
    void dequeue()
    {
        if (front == NULL)
        {
            cout << "Queue is Empty" << endl;
        }
        else
        {
            quelink *temp = front;
            if (temp->data == rear->data)
            {
                front = NULL;
                rear = NULL;
                cout << "Line was Emptied" << endl;
            }
            else
            {
                front = front->next;
                cout << "the data ( " << temp->data << " ) has been destroyed" <<
endl;
                delete (temp);
            }
        }
    }
    void display()
    {
        if (front != NULL && rear != NULL)
        {
            cout << "Front =" << front->data << endl;
            cout << "Rear = " << rear->data << endl;
        }
        quelink *iterate = front;
        if (iterate == NULL)
        {
```

```
            cout << "Counter is empty" << endl;
        }
        else
        {
            while (iterate != NULL)
            {
                cout << "| " << iterate->data << "  ";
                cout << iterate->next << endl;
                iterate = iterate->next;
            }
            cout << "|" << endl;
        }
    }
};
void menu()
{
    cout << "***Simple Queue Menu***" << endl;
    cout << "1. Enqueue\n2. Dequeue\n3. Display\n4. Exit" << endl;
    cout << "Enter Your Option -> ";
}
int main()
{
    int choice, r, x;
    TicketCounter test;
    while (choice != 4)
    {
        menu();
        cin >> choice;
        switch (choice)
        {
        case 1:
            cout << "Enter your Enqueued Value : ";
            cin >> x;
            test.enqueue(x);
            break;
        case 2:
            test.dequeue();
            break;
        case 3:
            cout << "Your Displayed Queue is :" << endl;
            test.display();
            break;
        }
    }
}
```

Topic 3

```
#include <iostream>
using namespace std;
long long ackermann(int m, int n)
{
    if (m == 0)
    {
        return n + 1;
    }
    else if ((m > 0) && (n == 0))
    {
        return ackermann(m - 1, 1);
    }
    else if ((m > 0) && (n > 0))
    {
        return ackermann(m - 1, ackermann(m, n - 1));
    }
    return 0;
}
int main()
{
    int A, x, y;
    cout << "Enter the value of m and n : ";
    cin >> x >> y;
    A = ackermann(x, y);
    cout << "The ackermann functional value is " << A << endl;
    return 0;
}
```

Topic 4

```
#include <iostream>
using namespace std;
int cnt = 0;
int towerOfHanoi(int n, char src, char des, char help)
{
    if (n != 0)
    {
        cnt++;
        towerOfHanoi(n - 1, src, help, des);
        cout << "Move Disk " << n << " from Source " << src
             << " to Destination " << des << endl;
        towerOfHanoi(n - 1, help, des, src);
    }
    return cnt;
}
int main()
```

```
{
    int N;
    cout << "Enter your Disk Numbers : ";
    cin >> N;
    int res = towerOfHanoi(N, 'A', 'C', 'B');
    cout << "Total moves taken is :  " << res << endl;
    return 0;
}
```