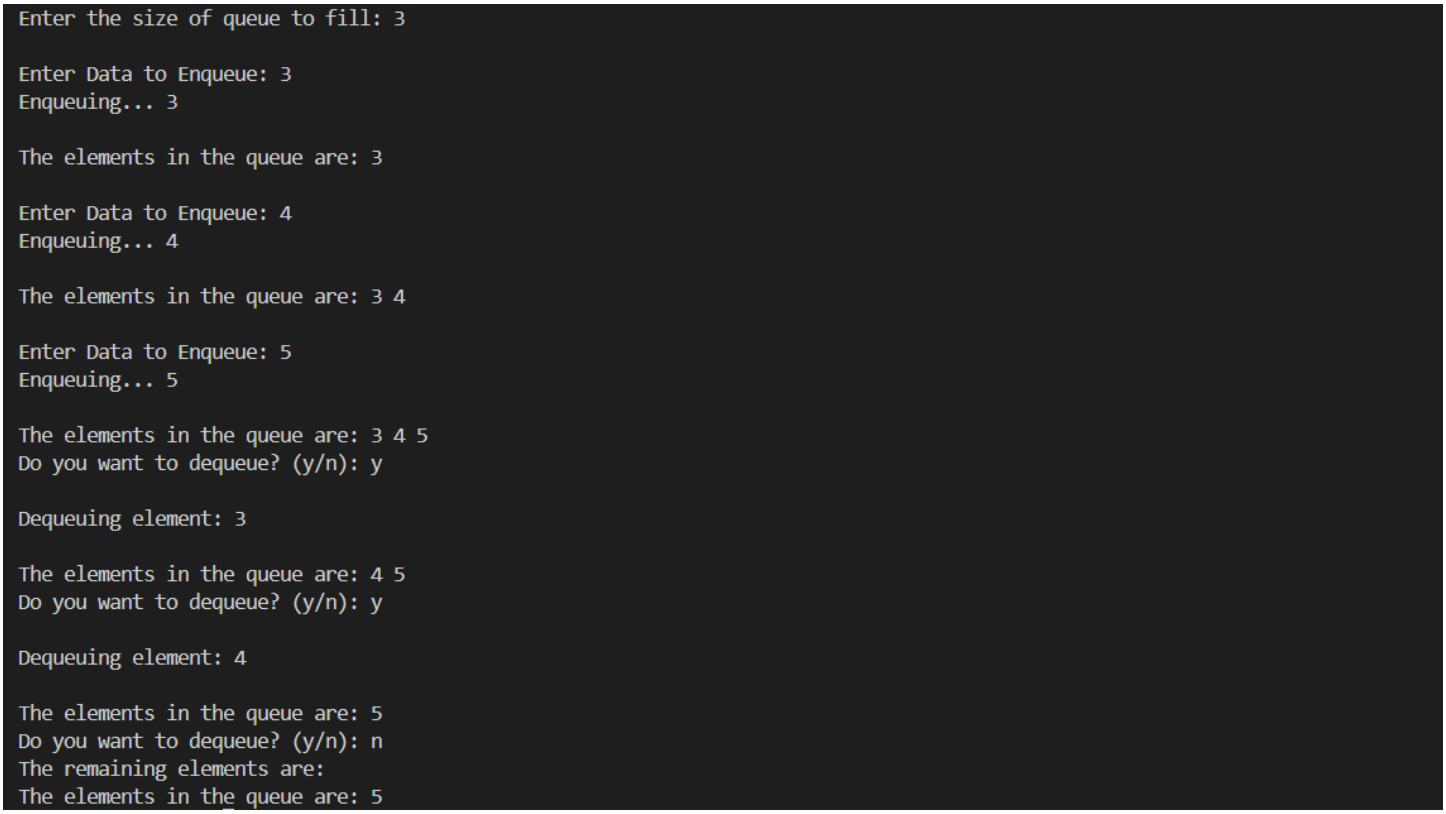
```cpp
#include <iostream>
using namespace std;

class QueueOper
{
public:
    static const int MAX = 50;
    int queue[MAX];
    int front, rear;

    QueueOper(){
        front = -1;
        rear = -1;
    }

    void enqueue(int data){
        if (rear == MAX - 1)
        {
            cout << "\nQueue is Full!";
        }
        else{
            rear++;
            queue[rear] = data;

            if (front == -1){
                front = 0;
            }
        }
    }

    void dequeue(){
        if (front == -1 || front > rear){
            cout << "\nQueue is Empty! Cannot Dequeue.";
            return;
        }
        else{
            cout << "\nDequeuing element: " << queue[front] << endl;
            front++;

            if (front > rear){
                front = -1;
                rear = -1;
            }
        }
    }

    void printQueue()
    {
        cout << "\nThe elements in the queue are: ";
        if (front == -1){
            cout << "Queue is Empty";
        }
```

```
        else{
            for (int i = front; i <= rear; i++){
                cout << queue[i] << " ";
            }
        }
        cout << endl;
    }
};

int main(){
    QueueOper q;
    int data, size;

    cout << "Enter the size of queue to fill: ";
    cin >> size;

    if (size > QueueOper::MAX){
        cout << "Size exceeds maximum limit of 50.";
        return 0;
    }

    for (int i = 0; i < size; i++){
        cout << "\nEnter Data to Enqueue: ";
        cin >> data;

        cout << "Enqueuing... " << data << endl;
        q.enqueue(data);
        q.printQueue();
    }

    char ans = 'n';
cout << "Do you want to dequeue? (y/n): ";
cin >> ans;

// Loop as long as the user wants to AND the queue is not empty.
while(ans == 'y' && size > 0) {
    q.dequeue();
    q.printQueue();
    size--;
    if (size > 0) { // Only ask again if there are elements left
        cout << "Do you want to dequeue? (y/n): ";
        cin >> ans;
    }
}

    cout << "The remaining elements are: ";
    q.printQueue();

    return 0;
}
```



This method removes an element from the front of the queue. It checks if the queue is empty (front == -1 or front > rear). If not empty, it prints the element at the front and then increments the front index. This "removes" the element by making it inaccessible. After dequeuing, it checks if the queue has become empty (front > rear). If so, it resets front and rear to -1.

The main part of the program controls how the user works with the queue. First, it makes a QueueOper object. Then it starts the enqueue, where it asks the user how many items to put in the queue. It makes sure this number is not bigger than the queue's MAX length. After that, it repeats the steps for that number: ask for data, add it to the queue, and show the queue after each addition.

When the queue is full, the program starts the dequeue. It asks the user if they want to take out an item. A while loop runs this process as long as the user says "y" and there are still items in the queue. Inside the loop, it removes one item, shows the new queue, and asks again only if the queue is not empty. When the loop ends, the program shows the final queue and stops.