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

struct Queue {
    int front, rear, capacity;
    int* queue;
    Queue(int c)
    {
        front = rear = 0;
        capacity = c;
        queue = new int;
    }

    ~Queue() { delete[] queue; }

    void queueEnqueue(int data)
    {

        if (capacity == rear) {
            printf("\nQueue is full\n");
            return;
        }


        else {
            queue[rear] = data;
            rear++;
        }
        return;
    }

    void queueDequeue()
    {

        if (front == rear) {
            printf("\nQueue is  empty\n");
            return;
        }

        else {
            for (int i = 0; i < rear - 1; i++) {
                queue[i] = queue[i + 1];
            }

            rear--;
        }
        return;
    }

```

```
50
51      void queueDisplay()
52      {
53          int i;
54          if (front == rear) {
55              printf("\nQueue is Empty\n");
56              return;
57          }
58
59
60          for (i = front; i < rear; i++) {
61              printf(" %d <-- ", queue[i]);
62          }
63          return;
64      }
65
66
67      void queueFront()
68      {
69          if (front == rear) {
70              printf("\nQueue is Empty\n");
71              return;
72          }
73          printf("\nFront Element is: %d", queue[front]);
74          return;
75      }
76  };
77
78  int main(void)
79  {
80      // Create a queue of capacity 4
81      Queue q(4);
82
83      // print Queue elements
84      q.queueDisplay();
85
86      // inserting elements in the queue
87      q.queueEnqueue(20);
88      q.queueEnqueue(30);
89      q.queueEnqueue(40);
90      q.queueEnqueue(50);
91
92      // print Queue elements
93      q.queueDisplay();
94
95      // insert element in the queue
96      q.queueEnqueue(60);
97
98      // print Queue elements
```

```
 99      q.queueDisplay();
100
101      q.queueDequeue();
102      q.queueDequeue();
103
104      printf("\n\nafter two node deletion\n\n");
105
106      // print Queue elements
107      q.queueDisplay();
108
109      // print front of the queue
110      q.queueFront();
111
112      return 0;
113  }
```