Question3.cpp

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

template <class T>
class Queue
{
private:
    int front, rear;
    unsigned capacity;
    T *array;

public:
    Queue(int capacity)
    {
        array = new T[capacity];
        front = -1;
        rear = -1;
        this->capacity = capacity;
    }

    bool isEmpty()
    {
        return (rear == -1 && front == -1);
    }

    bool isFull()
    {
        return ((rear + 1) % capacity == front);
    }

    void enqueue(T element)
    {
        if (isFull())
        {
            cerr << "Queue is full cannot enqueue." << endl;
            return;
        }
        else if (isEmpty())
        {
            front = rear = 0;
        }
        else
        {
            rear = (rear + 1) % capacity;
        }
        array[rear] = element;
        cout << array[rear] << " is enqueued in the queue." << endl;
    }

    void dequeue()
    {
        if (isEmpty())
        {
            cerr << "Queue is Empty cannot dequeue." << endl;
            return;
        }
        else if (front == rear)
        {
            cout << array[front] << " is dequeued from the queue." << endl;
            rear = front = -1;
        }
        else
        {
            cout << array[front] << " is dequeued from the queue." << endl;
            front = (front + 1) % capacity;
        }
    }

    T peek()
    {
        if (isEmpty())
        {
            cerr << "Queue is empty." << endl;
        }
        return array[front];
    }

    int queue_size()
    {
        if (isEmpty())
            return 0;
        else if (rear >= front)
        {
            return rear - front + 1;
        }
        else
        {
            return (capacity - front + rear) % capacity;
        }
    }
};

int main()
{
    Queue<char> q(10);
    q.enqueue('a');
    q.enqueue('b');
    q.enqueue('c');
    q.enqueue('d');
    q.enqueue('e');
    q.enqueue('f');
    q.enqueue('g');
    q.enqueue('h');
    q.enqueue('h');
    q.enqueue('h');
    q.enqueue('h');
    q.enqueue('h');
    q.enqueue('h');
    q.dequeue();
    q.dequeue();
    cout << "Front element " << q.peek() << endl;
    cout << "Size of queue " << q.queue_size() << endl;
    return 0;
}

```