Queue.h

```cpp
#ifndef QUEUE_H
#define QUEUE_H

#include <iostream>
using namespace std;

template <class T>
class Node
{
public:
    T data;
    Node *next;
    Node(T data)
    {
        this->data = data;
        next = nullptr;
    }
};

template <class T>
class Queue
{
private:
    Node<T> *rear;
    Node<T> *front;
    int size;
    int capacity;

public:
    Queue(int capacity) : rear(nullptr), front(nullptr)
    {
        this->capacity = capacity;
        size = 0;
    }

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

    bool isFull()
    {
        return (size == capacity);
    }

    void Put(T value)
    {
        Node<T> *newnode = new Node<T>(value);
        if (isFull())
        {
            cerr << "Cannot enqueue queue is full" << endl;
            delete newnode;
            return;
        }
        else if (isEmpty())
        {
            rear = front = newnode;
        }
        else
        {
            rear->next = newnode;
            rear = newnode;
            ++size;
        }
    }

    void dequeue()
    {
        if (isEmpty())
        {
            cerr << "Queue is empty cannot dequeu." << endl;
            return;
        }
        else if (rear == front)
        {
            rear = front = nullptr;
        }
        else
        {
            Node<T> *temp = front;
            front = front->next;
            delete temp;
        }
        --size;
    }

    T Get()
    {
        T value = front->data;
        dequeue();
        return value;
    }
};
#endif
```