Question02.cpp

```cpp
// <----Lab 06 - Queues---->

// Q2. Please implement the Generic Queue definition using singly linked list (you may use the
//     Singly Linked List that you already developed in Lab # 3), you may also add any
//     functions needed in the Singly Linked List definition given in Lab # 3. Your
//     implementation should work for the main function given below.
#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)
```

```
53      {
54          Node<T> *newnode = new Node<T>(value);
55          if (isFull())
56          {
57              cerr << "Cannot enqueue queue is full" << endl;
58              delete newnode;
59              return;
60          }
61          else if (isEmpty())
62          {
63              rear = front = newnode;
64          }
65          else
66          {
67              rear->next = newnode;
68              rear = newnode;
69              ++size;
70          }
71      }
72
73      void dequeue()
74      {
75          if (isEmpty())
76          {
77              cerr << "Queue is empty cannot dequeu." << endl;
78              return;
79          }
80          else if (rear == front)
81          {
82              rear = front = nullptr;
83          }
84          else
85          {
86              Node<T> *temp = front;
87              front = front->next;
88              delete temp;
89          }
90          --size;
91      }
92
93      T Get()
94      {
95          T value = front->data;
96          dequeue();
97          return value;
98      }
99  };
100 #endif
```