Programming Assignment #7

Due: 12/25/2022 23:59:59

Topic: Designing a template class for ADT queue with exception and stream I/O

Objective: Use template to create a generic queue and test the container data structure with data from a file.

Description:
You are asked to design a general-purpose queue class called `Queue` by means of template. You have to protect this data structure with good error checking, and the class should throw exceptions when needed. In addition to the "Big Three", the `Queue` class exports at least two functions, `enqueue` and `dequeue`, which have the usual meanings. The class should be implemented in its own .h and .cc files. In the test routine, you should instantiate the queue with two data types: `int` and `char`. The sources of these data are from two separate files (or from the standard input as explained later). You should call `enqueue` with these data, duplicate your queue objects, and then `dequeue` from these new objects to show that your class works. After the operations, you should print the results of `dequeue` to the console screen.

The queue itself should store its data in a linked list. As such the queue will have no fixed length. The queue will store a pointer to the head and the tail of the linked list, respectively, such that the `enqueue` and `dequeue` operations can be accomplished in constant time.

Your class should throw exceptions on at least three occasions:
1. when an internal error, such as incorrect head or tail pointers in the queue,
2. when you can not allocate more memory in enqueue operations,
3. when you try to dequeue an empty queue.

Assignment Directory: `/usr/local/class/oop/assign/assign7`

Sample Input and Output Files:
A sample but incomplete test file, `QueueTest.cc`, is given to you in the assignment directory. In this file, you should test your class with complete exercises consisting of at least the following code segment.

```
// test integer queue
Queue<int> *myIntQPtr=new Queue<int>;
// enqueue exercises on myIntQPtr
…
// test copy constructor
Queue<int> myIntQ = *myIntQPtr;
// test destructor
delete myIntQPtr;
// dequeue exercises on myIntQ
…
// test character queue
Queue<char> myCharQ, myCharQ1;
// enqueue exercises on myCharQ
…
// test assignment operator
myCharQ1=myCharQ;
// dequeue exercise on myCharQ1
…
// dequeue an empty queue to throw an exception
…
```

In addition, two sample data files: `intq.dat` and `charq.dat` are given to you for testing.

```
li@oop 12:09pm >./Queue_x64-sample_i86pc intq.dat charq.dat
Integer Queue:
1
2
3
4
5
Character Queue:
A
B
C
d
Attempt to dequeue empty queue
ID: 86701000
li@oop 12:09pm >
```

The first and second command-line arguments are the input file names with integer and character data in them, respectively. When the arguments are missing, standard input `cin` is assumed. For example, "**`./Queue_x64-sample`** `intq.dat < charq.dat`**"** should produce the same result as in the previous example.

Files to hand in:
You need to write the `Queue.h` and `Queue.cc` files and complete the `QueueTest.cc` file. As usual, you have to hand in the whole package electronically.