CMPSCI 182 – Project 4
Queues
30 points total
Due 4/24/19

Write a reference-based implementation of a queue that uses a linear linked list to represent the items in a queue. You will need both a head reference and a tail reference. Be sure that the project includes all of the following:

**A Node class.** Your queue ADT will need this class in order to instantiate the Node objects which make up the linear linked list. You may use the Node class you wrote for Project 3.

**A Queue ADT.** This is a class which contains global, private **head** and **tail** reference variables as well as the Queue's "wall of operations". This "wall of operations" are the methods which are responsible for creating and managing the ADT's Linked List which. Queue ADT will use this Linked List to represent the queue. You may use the authors' QueueReferenceBased ADT (on page 418), or you may write your own ADT. Just be aware that if you use the authors' QueueReferenceBased ADT you will have to modify it slightly, because the ADT given in the textbook has only one reference variable, lastNode, and represents a circular queue.

**A Test class.** This class will contain a main method which instantiates an object from your Queue ADT class and invokes every Queue ADT method. What are some ways you can invoke the Queue ADT methods to demonstrate that those methods work in the way that they are intended to?

A Java ***package*** is a group of related Java classes. Since all of the source code files in this project are related, be sure that all of the source code files are in the same package. If any of your files are in a different package, you must use an ***import*** statement to let Netbeans know the package in which the file will be found. Also, if your source code files are contained in a package other than the <default> package, you must include a statement:

```
package packagename;
```

Where packagename is the name of the package in which all of your Project 4-related source code files are stored.

When you are done, compare your implementation to the one given in this chapter that uses a circular linked list with one external reference. **Answer the following analysis questions in the "Comments" textbox when you submit your project to Canvas:**

1. In your opinion, which implementation is easier to *write*?
2. In your opinion, which is easier to *understand*?
3. In your opinion, which is more *efficient*?