CSCI 241 Fall 2015
Project 3: Deques, Stacks, and Queues

DUE 0500 Monday 16 November 2015

AT NO POINT MAY YOU WORK ON THIS PROJECT WITHOUT THE PHYSICAL PRESENCE OF YOUR TEAMMATE(S)

It is imperative that before you begin this project, you have a fully functional Linked List implementation. Select the easiest to fix of your two (or three) options and correct the problems identified by the test cases. If you have a working Linked List, this project becomes significantly easier.

In this project, your team will implement three data structures: a deque, a stack, and a queue. This project emphasizes what we call class composition, where one class includes an instance of another class as an attribute. To accomplish this, you will build your Deque first. The inner storage medium for your Deque should be a (now fully functional) Linked List. Once your Deque is complete, you will use it as the storage medium for your Stack implementation and your Queue implementation.

To use a class from one file in another file, you begin with a line similar to:

```
from file import Type_I_Need
```

This line opens the file `file.py` and imports the `Type_I_Need` class so that you can declare objects of that class in your new file. I have included the import line in the skeleton file for your Deque class. Use it as an example to include your Deque class in your Stack and Queue implementations.

We have discussed the operations for Deques and Stacks and Queues in class extensively, so I will not repeat those details here. The expected methods, however, are included in skeleton files for you to implement. Because the files reference each other, you must place them all (including the fixed `Linked_List.py` file) in the same folder, otherwise you would have to configure the system to go look for them elsewhere.

In the main section of each of your Python files, provide some test cases to ensure that your Deque, Stack, and Queue implementations function correctly. Though this is not an exhaustive list, some things to consider are:

1. Does your Stack operate exclusively in LIFO mode? It should not be possible to access the data in any other pattern from outside of the class.
2. Does your Queue operate exclusively in FIFO mode? It should not be possible to access the data in any other pattern from outside of the class.
3. Does your Deque implementation allow operations only at the front and back?
4. Do peek operations leave the contents of the structures unchanged?
5. Does `len(my_object)` return the number of items in `my_object`?

A few days after this project is released, a slight extension will be added. Begin by implementing the Deque, Stack, and Queue class methods and performing your initial tests. Soon, I will ask you to use your implementation to solve a specific problem. Your three Python scripts should contain your implementations, tests, and solution to the problem that will be released soon. You will also submit a jointly created writeup for your team that includes

1. a description of any issues your team encountered during the implementation;

2. a justification for each of your test cases, and an explanation of why those test cases are complete; and
3. a brief explanation of how you could use an array as the storage medium instead of a Linked List or Deque for each of the three structures.

Your submission should consist of a single ZIP file containing your Python scripts and your team's writeup. If your team name is Yoda, then your ZIP file should be called Yoda.zip, and it should contain files called Deque.py, Stack.py, and Queue.py, and a file called Yoda.docx or Yoda.pdf.