CIS 313, Intermediate Data Structures
Winter 2019

CIS 313 Lab 1

Due: Friday at 4:55 pm, Feb. 1, 2019

This lab involves implementing linear data structures - stacks and queues.

Overview

Fill out all of the methods in the provided skeleton code. You may add additional methods, but should NOT add additional public data to Stack or Queue. You also should not change the name of any of the classes or files.

You may not use Python built-in data abstractions like lists or dictionaries. You may use string, and string functions if you think they will help.

In main, read each line, and call isPalindrome.

In isPalindrome, iterate over the characters in the line, and process with Stack and Queue.
You should think about how to use a Stack and Queue to determine if a string is a palindrome.

In the Stack and Queue classes, fill out each of the public methods. Do NOT change the arguments or return types of the public methods. You may, however, add private methods.

Input Description

The input will be a text file, for example *inSample.txt* below will be provided. The first line will contain an integer N, which is the number of lines to follow. Each of the N lines contains a string of numbers not separated by spaces.

For each line, use your stack and queue to determine if the input represents a palindrome.
The input strings are made up of non-negative numbers of different sizes.

```
5
30325
1133311
613373316
44
56
```

Note: Feel free to create your own test input files.

Output Description

For each line of the input, if the string of numbers reads the same way forward as backwards output "This is a Palindrome.", otherwise output: "Not a Palindrome." For example, using the sample input above, your program should output:

```
Not a Palindrome.
This is a Palindrome.
This is a Palindrome.
This is a Palindrome.
Not a Palindrome.
```

Testing Protocol

We strongly suggest you test your program in the following ways:

- While creating it
- With inSample.txt
- With your own imagined test cases.

Grading

This assignment will be graded as follows:

Correctness 50%

Your program compiles without errors (including the submitted files NOT containing package names at the top. Delete the package name from the top of the files before you submit them): 10%
Your program runs without errors: 10%
Your program produces the correct output: 30%

Implementation 50%

Your Stack class implements all of the proper methods in O(1): 20%
Your Queue class implements all of the proper methods in O(1): 20%
Your palindrome test uses a Stack and a Queue, to achieve linear running time O(n) 10%

To earn points for implementation, your code must be clear enough for us to understand

Further, you may not use any data structures from the python library including lists, and dictionaries

Extra Credit 20%

To receive points for the extra credit you must implement a queue using two stacks:

- Create a new class file called TwoStackQueue, which has two stacks as member variables
- The signature of this class should be the same as the Queue class.
  - It should have the same public methods
  - Each public method should match in argument types (argument number, etc), and return type.
    - Note that this is not the same as having the same methods.
    - In particular, you may modify the bodies of public methods, and change add or remove private methods.
  - Amortized complexity $O(1)$ for each operation.
- create an additional public class file called EC
  - All the same functionality as lab1.py
  - use a TwoStackQueue instead of a Queue in the isPalindrome method

Testing Extra Credit

In order to test your TwoStackQueue class, simply test it in all the same ways you tested your original program.