2-6-08 Lab 4 CS III Name:_____

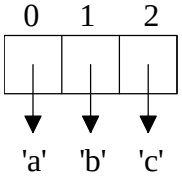
Objective: To experiment with stack implementations in Python to obverse the importance of big-oh analysis of operations. To gain experience implementing a linked Stack implementation.

The Assignment Overview

For this lab, you will design an experiment to test the conjecture that Python's list implementation is an array of references to list items.

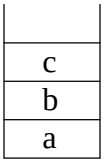
Additionally, you are to implement and test a Stack class using a linked structure.

Activity 1: Recall that I conjectured in class (after reading on the web) that Python's built-in list is implemented as an array of references (pointers) to list items. For example, the list ['a', 'b', 'c'] would really be implemented as the array.



This has performance ramifications for possible Stack implementations using a list representation. The two possible list representation of a Stack being:

"Abstract" Stack



I. Stack with Bottom at list [0] — 0 1 2 ['a', 'b', 'c'] — P:\810-063-CSIII\LABS\LAB_4 (stackClass.py)

II. Stack with Top at list [0] — 0 1 2 ['c', 'b', 'a'] — (stackClassAlt.py)

Complete the expected big-oh notation for each Stack implementation

	push(item)	pop( )
Stack with Bottom at list [0]		
Stack with Top at list [0]		

Your job for this activity is to design a test program(s) using the Stack representations to see if my conjecture that list are implemented as arrays is correct.

After you have designed you test program, raise your hand and explain your experimental design.

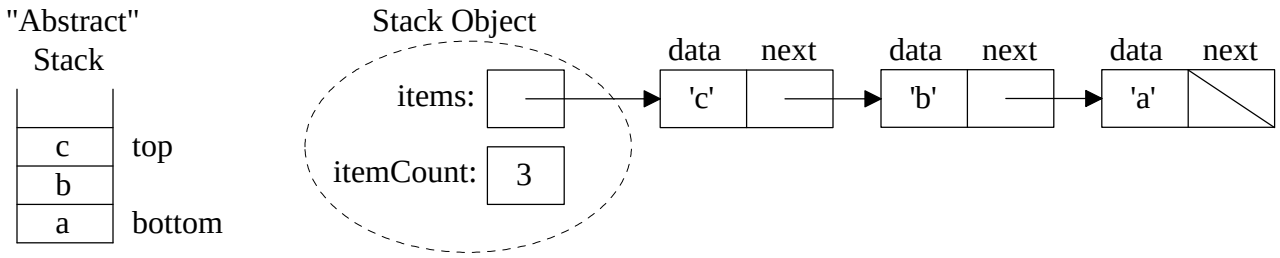
After you receive some feedback on your design, implement your test program(s). To time your program's execution call the "`times()`" function after first issuing "`from os import times`". The "`times()`" function returns a tuple whose [0] item contains the accumulated user-time in seconds for the program, and [1] contains the accumulated system-time in seconds for the program. To time some section of your program

1. record the starting time,
2. run the code to be timed,
3. record the ending time, and
4. calculate the difference between the starting and ending times.

The execution time of the program is approximately the sum of the user-time and system-time.

After you have implemented AND fully tested your program(s), raise your hand and demonstrate the it.

Activity 2: Copy the Node class implementation defined in the P:\810-063-CSIII\LABS\LAB_4\nodeClass.py. Use this Node class to implement a Stack class using a linked representation as discussed in class. Conceptually, a Stack object would look like:



Include in your Stack class the following methods: `__init__, isEmpty, push, pop, peek, size,` and `show`. The `show` method should print the stack items from top to bottom.

After you have implemented AND fully tested your Stack class, raise your hand and demonstrate the it.

NOTE: If you complete all of the activities within the lab period, you do not need to hand anything it. However, if you need to finish activities outside of the lab period, then hand in:

Activity 1:

- a hard-copy of the file(s) containing your Python program(s) to test the list implementation.
- the output produced by running the program(s). In Windows, IDLE has a File | Print Window menu options that you can use, and you can use the <Alt> and <Print Screen> to capture the Python window(s).

Activity 2:

- a hard-copy of the file(s) containing your Python Stack class using a linked representation (stackClassLinked.py) and the test script to demonstrate this Stack implementation.
- the output produced by running the test script. In Windows, IDLE has a File | Print Window menu options that you can use, and you can use the <Alt> and <Print Screen> to capture the Python window(s).