Project 1

points
Due January 25th 8:00pm
***Reference:** Author of this project is Erika Lustig*

This is not a team work, do not copy somebody else's work.

Assignment Overview

For this project, you will be implementing a Linked List class for singly linked lists. As learned in previous classes, singly linked list nodes have only a 'next' node pointer and a value. The Node class has already been implemented for you, in addition to a few functions of the Linked List class. Your assignment is to complete the remaining functions of the Linked List class.
A review can be found on pages 256-260 of your textbook.
Note: the project implementation may vary from the textbook.

Assignment Deliverables

LinkedList.py

Be sure to use the specified file name(s) and to submit your files for grading **via D2L Dropbox** before the project deadline.

Assignment Specifications

Complete the functions included in the Linked List class.

The **Node class** is included in the skeleton file with the following functions and **should not be edited.**

- __init__(self, value, next_node=None)
  This function initializes a node with a given value. Optional argument is the next_node reference, which will default to None if not passed in.
- __eq__(self, other)
  This provides comparison ('==') for nodes. If nodes have the same value, they are considered equal.
- __repr__(self)
  A node is represented in string form as 'value'. Use str(node) to make into a string.

The **Linked List class** is partially completed in the skeleton file. **Function signatures or provided functions may not be edited in any way.**
The following functions are provided and may be used as needed.

- __init__(self)
  A Linked List class has variables head, tail, and size. Self.head is the first node in the linked list, self.tail is the last node in the linked list, and size is the number of nodes.
- __eq__(self, other)
  This function is used to compare two Linked Lists. This will primarily be used for grading purposes - DO NOT CHANGE.
- __repr__(self)
  This function will return the string form of a list of all of the node values, in order from head to tail. Please make sure you fully understand how to traverse Linked Lists, even though this function is provided to you in this project.

You must complete and implement the following functions. Take note of the specified return values and input parameters. **Do not change the function signatures.**

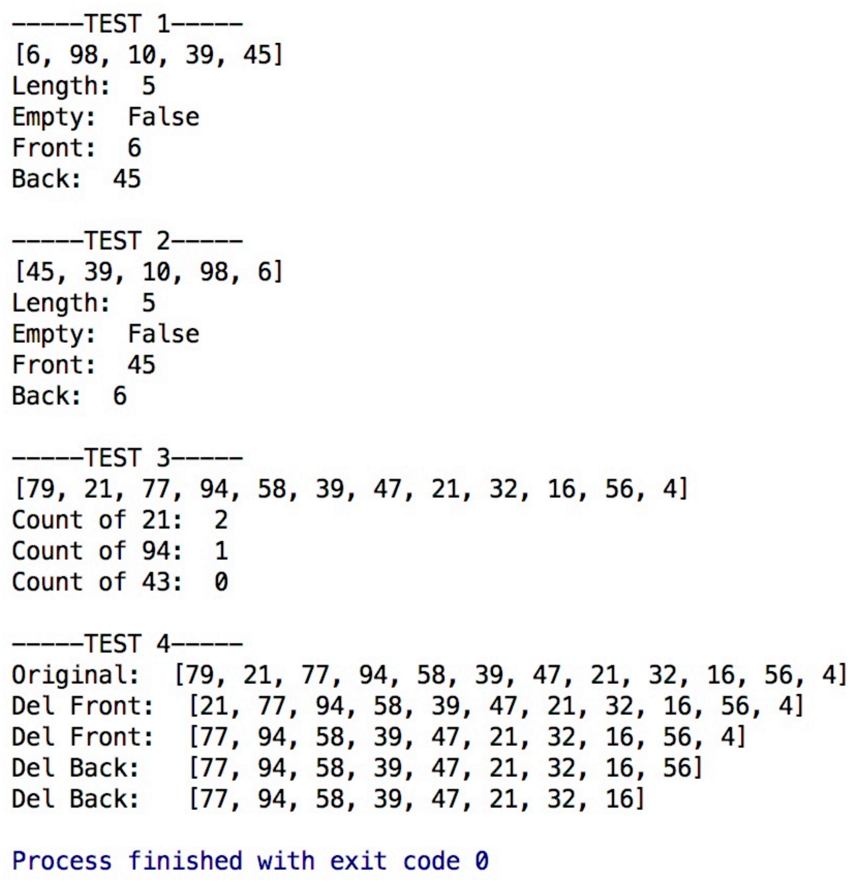
- **length(self)**
  Returns the number of nodes in the list.
- **is_empty(self)**
  Returns true if the linked list is empty, false if it is not empty.
- **front_value(self)**
  Returns the value of the front (head) node.
- **back_value(self)**
  Returns the value of the front (tail) node.
- **count(self, val)**
  Takes a value 'val' and counts how many times that value occurs in the linked list. Returns the number of occurrences.
- **find(self, val)**
  Takes a value 'val' and returns true if the value is found in the list, false if the value is not found.
- **push_front(self, val)**
  This function takes a parameter 'val' and inserts a node with value 'val' at the front (head) of the linked list.
- **push_back(self, val)**
  This function takes a parameter 'val' and inserts a node with value 'val' at the back (tail) of the linked list.
- **pop_front(self)**
  This function removes the front (head) node of the linked list.
- **pop_back(self)**
  This function removes the back (tail) node of the linked list.

Assignment Notes

Points will be deducted if your solution has any warnings of type.

- Do not modify the Node class.
- For this project, you have been provided the docstrings for each function (the part in triple quotes). You will be required to add and complete the docstrings in future projects.
- You are provided skeleton code for the LinkedList class and must complete the included methods. You may add more functions in LinkedList class than what is provided**, but may not alter the function signatures in anyway**.
- To test your classes, Project1_Main.py is provided and uses some of the required functions. Compare your results to the output below.
- Errors when using your class that cause the grading script to fail will result in a 25% deduction.

Results of Project1_Main.py:



```
-----TEST 1-----
[6, 98, 10, 39, 45]
Length:  5
Empty:  False
Front:  6
Back:  45

-----TEST 2-----
[45, 39, 10, 98, 6]
Length:  5
Empty:  False
Front:  45
Back:  6

-----TEST 3-----
[79, 21, 77, 94, 58, 39, 47, 21, 32, 16, 56, 4]
Count of 21:  2
Count of 94:  1
Count of 43:  0

-----TEST 4-----
Original:  [79, 21, 77, 94, 58, 39, 47, 21, 32, 16, 56, 4]
Del Front:  [21, 77, 94, 58, 39, 47, 21, 32, 16, 56, 4]
Del Front:  [77, 94, 58, 39, 47, 21, 32, 16, 56, 4]
Del Back:   [77, 94, 58, 39, 47, 21, 32, 16, 56]
Del Back:   [77, 94, 58, 39, 47, 21, 32, 16]

Process finished with exit code 0
```