

Extra Credit 2 - Linked Lists

In this extra credit assignment you will create your own **Stack** and **Queue** data structures based on the ideas we learned about **Linked Lists**.

Submission:

- Submit a file named `linkedlist.py` (use the template I have started for you).
- You can find the template and test file in the Extra Credit 2 Files folder in Week 6
- Put your name in a comment at the top of the file.
- You may turn this assignment in at any time.
- This assignment is worth an extra **4%** towards your final grade.

Part 1 - The Node Class

Before you build a Stack or Queue, you need to create a simple **Node** class. Here is the box model for the class:

```
+-----+
| Node   |
+-----+
| value  |
| next   |
| prev   |
+-----+
| __init__ |
+-----+
```

- The `value` attribute carries the actual value to be stored. This can be any type of variable.
- The `next` attribute points to the next `Node` in the list. If there is no next node (i.e. this node is the tail) then set `next` to `None`.
- The `prev` attribute points to the previous `Node` in the list. If there is not previous node (i.e. this node is the head) then set `prev` to `None`.
- The `__init__(self, value)` method should take a value as an argument. Set the `value` attribute to the passed in value, and initialize `next` and `prev` to `None`.

Part 2 - The Queue class

Now that you have nodes that can be string together in a list, create a Queue class. Here is the box model:

```
+-----+
| Queue  |
+-----+
| head   |
| tail   |
| length |
+-----+
| __init__ |
| push     |
| pop      |
+-----+
```

- The `head` and `tail` attributes contain to the `Node` objects in those positions.
- The `length` attribute keeps track of the total number of nodes.

- The pseudocode for the three methods is below:

__init__(self):

- Assign None to both head and tail
- Assign 0 to length

push(self, value):

- Create a new Node object using the value argument
- Check if tail is None:
 - If it is, then assign the new Node to both head and tail
 - Otherwise:
 - Assign the new node to the current tail node's next attribute
 - Then assign the new node to the Queue's tail attribute
- Increment the length attribute

pop(self):

- Assign the current head node to a temporary variable.
- Assign the current head's next node to the Queue's head.
- Decrement the length
- Return the value of the temporary node.

Part 3 - The Stack Class

- The box model for the Stack class is the same as the Queue class.
- The difference lies in the implementation of the pop() method.
 - Recall that a Queue is *First In, First Out*, whereas a Stack is *Last In, First Out*.
 - So a Queue pops off the head node, while a Stack pops off the tail node.
- Therefore we can simply make Stack a subclass of Queue, and then override the pop() method.

```
+-----+
| Stack |
+-----+
| head  |
| tail  |
| length|
+-----+
| __init__
| push
| pop
+-----+
```

pop(self):

- Assign the current tail node to a temporary variable.
- Assign the current tail's prev node to the Stack's tail.
- Now check the value of tail.
 - If it is None then assign None to head as well.
 - Otherwise, set the new tail node's next attribute to None.
- Decrement the length
- Return the value of the temporary node.

Testing

Now that you have the classes defined, you can try them out.

- Create a queue and start pushing values to it and check that the length attribute is correct.
- Then try popping values back out. Do they come out in the correct order?

I have written a test suite for you. To run it, do the following:

- Make sure your file is in the same directory as the test file.
- type: `python3 linkedlist_test.py`
- If everything is correct you should get something like this:

```
jgomez$ python3 linkedlist_test.py
....
```

```
-----
Ran 4 tests in 0.000s
```

OK

However, if you run it without implementing the methods, all the tests will fail, like so:

```
jgomez$ python linkedlist_test.py
EEEE
```

```
=====
ERROR: test_pop (__main__.TestQueue)
```

```
-----
Traceback (most recent call last):
```

```
  File "linkedlist_test.py", line 20, in test_pop
    self.assertEqual(q.length, exp_len)
```

```
AttributeError: 'Queue' object has no attribute 'length'
```

```
=====
ERROR: test_push (__main__.TestQueue)
```

```
-----
Traceback (most recent call last):
```

```
  File "linkedlist_test.py", line 11, in test_push
    self.assertEqual(q.length, 1)
```

```
AttributeError: 'Queue' object has no attribute 'length'
```

```
=====
ERROR: test_pop (__main__.TestStack)
```

```
-----
Traceback (most recent call last):
```

```
  File "linkedlist_test.py", line 42, in test_pop
    self.assertEqual(s.length, exp_len)
```

```
AttributeError: 'Stack' object has no attribute 'length'
```

```
=====
ERROR: test_push (__main__.TestStack)
```

```
-----
Traceback (most recent call last):
```

```
  File "linkedlist_test.py", line 33, in test_push
    self.assertEqual(s.length, 1)
```

```
AttributeError: 'Stack' object has no attribute 'length'
```

```
-----
Ran 4 tests in 0.001s
```

FAILED (errors=4)