CSCA48 Exercise 4

Due: June 9, 2017. 5:00pm

Linked List Functions

This week, we'll be getting some more linked list practice. In order to keep things simple, we're going to build functions that work with Nodes directly. The Node class that we will be using is given below.

```
class Node():
    '''A node in a linked list'''
    def __init__(self, data, next_node=None):
        self.data = data
        self.next_node = next_node
```

Your functions will take in the head of a linked list, but that linked list may be empty. So your type contracts will look something like `(Node or NoneType) -> int`.

sum

Write a function called `sum` that takes in the head of a linked list, and returns the sum of all elements in that linked list (you may assume that all data values are integers

reverse

Write a function called `reverse` that takes in the head of a linked list, and returns the head of a linked list with the same values as the one input, but in reverse order.

(Optional): Try to do this without changing the `.data` value of any nodes, only changing the pointers

splice

Write a function called `splice` that takes in the head of a linked list, and and integer `i`, and returns the head of a linked list with the same values as the input linked list, except the values before index `i` and the values after index `i` have been swapped.

For example, if the linked list `A -> B -> C -> D -> E -> F` and the integer `3` were passed in, the resulting linked list would look like: `E -> F -> D -> A -> B -> C`

(Optional): Try to do this once by creating a new list, and once by mutating the existing list... which is easier? which is more efficient?

What to hand in

Include the `Node` and your functions in a file called `ex4.py`.