Homework 2: Doubly Linked Lists

Due: Friday, February 7 at 5pm on Canvas

Concepts: python lists, linked lists, runtimes in practice

1. (18 points) Implement a doubly linked list that supports the following operations:

   - Initialization of an empty list
   - Push - add an element with a given value to the front of the list
   - Insert - insert a new node with a given value after a given node
   - Delete - delete a given node
   - Index - return a node at a given index in the list
   - Multiply all pairs - sum over the product of node values for every unique combination of different nodes (you may assume all node values are numeric). Examples: the result for $[1, 2, 3]$ should be $1*2+2*3+1*3 = 11$, and the result for $[1, 1]$ should be $1*1 = 1$.

   More documentation is given in the file `hw2.py` on Canvas. Your solution should also include test functions for all the included operations. Each operation and its associated test functions are worth 3 points.

2. (6 points) To analyze an algorithm's runtime in practice, we can use the `timeit` library in python. For documentation of this package see `https://docs.python.org/3/library/timeit.html#module-timeit`.

   (a) (3 points) Use the `timeit` package to find the average time to index a doubly linked list and python list for $n = 10$ to $10,000$, where n is the length of the list, and plot the results. Comment on what you observe.

   For reference, here is an example using `timeit` and the `random` package to pick a random input value.

```
n = 10
l = DLL()
for i in range(n):
    l.push(i)

t = timeit.Timer('l.index(random.randrange(n))', 'import random',
    globals=locals())
t.timeit(50) # times the operation 50 times and averages
```

   (b) (3 points) Now plot the time to multiply all pairs of a doubly linked list over the same range of n. Comment on what you observe.