class _Node:
    """A node in a linked list.

    Note that this is considered a "private class", one
    which is only meant to be used in this module by the
    LinkedList class, but not by client code.
    ""

    == Attributes ==
    @type item: object
    The data stored in this node.
    @type next: _Node | None
    The next node in the list, or None if there are
    no more nodes in the list.

    """

def __init__(self, item):
    """Initialize a new node storing <item>, with no next node.
    ""
    @type self: _Node
    @type item: object
    @rtype: None

class LinkedList:
    """A linked list implementation of the List ADT.
    ""

    # == Private Attributes ==
    # @type _first: _Node | None
    #     The first node in the list, or None if the list is empty.

def __init__(self, items):
    """Initialize a new linked list containing the given items.
    ""
    The first node in the linked list contains the first item
    in <items>.

    @type self: LinkedList
    @type items: list
    @rtype: None

1. Suppose we create a linked list: `linky = LinkedList([1, 2, 3])`. Write an expression to access the second item in the list. (You can access private attributes for this question.)

   `>>>`

2. Suppose we create a linked list: `linky = LinkedList([1, 2, 3])`. Write an expression to access the third node in the list. (You can access private attributes for this question.)

   `>>>`
3. Write a method of the `LinkedList` class which returns the first item in the list, or raises an `IndexError` if the list is empty.

4. What is the asymptotic running time (another way of asking for the Big-Oh expression) of your method from Question 3?

5. Write a method of the `LinkedList` class which takes a number k and returns the item at index k in the list (indexing starts at 0), or raises an `IndexError` if the list has $\leq k$ elements.

6. What is the asymptotic running time of your method from Question 5? Think carefully about what determines how many times the loop runs.