Week 6 Quiz: Linked Lists

```python
def mystery(self, i, j):
    # This is a method in class LinkedList
    before = self._first
    curr_index = 0
    while curr_index < i - 1 and before is not None:
        before = before.next
        curr_index += 1
    after = before
    while curr_index < j and after is not None:
        after = after.next
        curr_index += 1
    hold = before.next
    before.next = after
    answer = LinkedList([])
    answer._first = hold
    return answer
```

You are about to trace the call below to method `mystery`. You will probably need some scrap paper for rough work.

```python
>>> linky = LinkedList([0, 1, 2, 3, 4, 5, 6, 7])
>>> s = linky.mystery(2, 6)
```

1. Draw the state of the linked list and the variables `before` and `curr_index` the first time we reach line 4.

2. Draw the state of the linked list and the variables `before` and `curr_index` when we reach line 7.

3. Draw the state of the linked list and the variables `before` and `curr_index` and `after` when we reach line 11.

4. Draw the state of the linked list and all the local variables when we reach line 13.

Continued on the back of this page.
5. Look at the final steps in this method. What do you think it is trying to do? It may remind you of a method on simply Python lists.

6. Do you notice anything odd about what the method does?

We have some great news for you: The midterm won’t have anything nearly as difficult as this about linked lists!