Final Sample Questions (#2)

Complexity

1. Write a function **`func(n)`** such that the complexity of **`func`** is $O(n^2)$.

2. The following is a simple version of a function that determines if a number is prime:

```
def is_prime(n):
    for d in range(2,n):
        if n % d == 0:
            return False
        return True
```

It checks to see if the parameter **`n`** is divisible by all of the numbers less than **`n`**.

a) What is the complexity of **`is_prime(n)?`**

b) Suppose that we make a simple change to the function and only check to see if **`n`** is divisible by 2 and then all of the odd numbers up to **`n`**. How does that change the complexity of **`is_prime()`**? Why or why not?

Lists

3. Recall the definition of a linked list:

```
class LinkedList:
    def __init__(self):
        self._head = None

class Node:
    def __init__(self,value):
        self._value = value
        self._next = None
```

Write a function **`len_ll(alist)`** that returns the length of the linked list **`alist`**. Use recursion in your solution.

Note: Since a **`LinkedList`** object has the attribute **`_head`** and a **`Node`** object does not, you will need to introduce a second function for your solution.