## CC 2.1.1: Scope Rules

Scope Rules: Question 1

1/1 point (graded)

Consider the following code:

```
def increment(n):
    n += 1
    print(n)

n = 1
increment(n)
print(n)
```

What will this print? Note that here, we separate lines of output with "`;`"

`1`; `1`

`2`; `1`
correct

`1`; `2`

`2`; `2`

**Explanation**
While `n` is defined globally on line 3, it is also defined locally in `increment`. Therefore, the function will print one greater than `1` due to the function and its argument, but the local `n` will not change.

### Scope Rules: Question 2

1/1 point (graded)

Consider the following code:

```
def increment(n):
    n += 1
    #blank#

n = 1
while n < 10:
    n = increment(n)
print(n)
```

Fill in the `#blank#` to ensure this prints `10`.

Enter your code here.

[return n]

## CC 2.1.2: Classes and Object-Oriented Programming

### Classes and Object-Oriented Programming: Question 1

1/1 point (graded)

Consider the following code:

```python
class NewList(list):
    def remove_max(self):
        self.remove(max(self))
    def append_sum(self):
        self.append(sum(self))

x = NewList([1,2,3])
while max(x) < 10:
    x.remove_max()
    x.append_sum()

print(x)
```

What will this print?

`10`

`[5,8,10]`

Nothing: this code contains an error.

Nothing: this program will never halt.
correct

**Explanation**
Upon first removing the maximal element (`3`) and adding then sum of the `MyList` object (`3`), the object has the same elements in the same locations as before (`[1, 2, 3]`). Therefore, the `while` condition will always be met, and the program will run as long as it can.