CS 160
CS Orientation
Programming Structure
Chap. 4

Remember Our Triangle

```python
print("   *   ")
print("  ***  ")
print(" ***** ")
print("*******")
```

Making it w/ a Loop

```python
sp=3
x=1
for b in range(sp):
    print(" " , end="")
for b in range(x):
    print("*",end="")
for b in range(sp):
    print(" ", end="")
print("")
```

```python
sp=2
x=3
for b in range(sp):
    print(" " , end="")
for b in range(x):
    print("*",end="")
for b in range(sp):
    print(" ", end="")
print("")
```
Make the Original w/ a Loop

```
sp=5
x=5
for b in range(sp):
    print('', end='')
for b in range(x):
    print('*',end='')
for b in range(sp):
    print('', end='')
print('')
sp=7
x=7
for b in range(sp):
    print('', end='')
for b in range(x):
    print('*',end='')
for b in range(sp):
    print('', end='')
print('')
```

Where's the code the same?

```
for b in range(sp):
    print(' ', end='')
for b in range(x):
    print('*',end='')
for b in range(sp):
    print('', end='')
```
Let’s call the function

draw_splats(3, 1)
draw_splats(2, 3)
draw_splats(1, 5)
draw_splats(0, 7)

• Now, what’s repetitive?
• What is the pattern?

The call to the function!!!

for x in range(3, -1, -1):
    draw_splats(x, 7-(x*2))

• Why have these three arguments to range?
  – Why use -1 instead of 0?
• What are the arguments to draw_splats?

Decomposition/Modularity

![Diagram of decomposition/modularity]
Scope of Local and Global Variables

Terminology

- Function vs. Method vs. Procedure vs. Module
  - Refer to the same thing!
- Actual Parameters vs. Arguments
  - Refer to the same thing!
- Formal Parameters vs. Parameters
  - Refer to the same thing!
- Pass by value vs. pass by reference
  - VERY, VERY different!!!

Decomposition

*indicates call-by-reference parameter; those parameters without an asterisk are call-by-value parameters.
Quiz #5

- Get into groups of 4 to 5.
- Did we pass the number of spaces and splats to `draw_splats()` by reference or value?
- How could you decompose this problem?

```python
def draw_splats(sp, x):
    for b in range(sp):
        print(" ", end="")
    for b in range(x):
        print("*", end="")
    for b in range(sp):
        print(" ", end="")
    print("")
```