

Chapter 4

Recursion

→ Recursive Implementation of a Factorial Function

```
def factorial(n):
```

```
    if n == 0:
```

```
        return 1
```

```
    else:
```

```
        return n * factorial(n-1)
```

Recursion Trace

factorial(3)

→ factorial(2)

→ factorial(1)

→ factorial(0)

← ret 3 * 2 = 6

← ret 2 * 1 = 2

← ret 1 * 1 = 1

← ret 1

→ Recursive Approach to Drawing a Ruler

- An interval with central tick length $L \geq 1$ is composed of:

- 1.) an interval w/ central tick length $L-1$
- 2.) single tick of length L
- 3.) interval w/ central tick length $L-1$

→ base case: $L=0$, no lines drawn

for $L \geq 1$, first and last steps are performed recursively.

// draw one line w/ given tick length

```
def draw_line(tick_length, tick_label=''):
    line = '-' * tick_length
```

```
    print(line)
```

// draw tick interval based on central tick length

```
def draw_interval(center_length):
```

```
    if center_length > 0:
```

```
        draw_interval(center_length-1)
```

```
        draw_line(center_length)
```

```
        draw_interval(center_length-1)
```

```
def draw_ruler(num_inch, major_len)
```

```
    draw_line(major_len, '0')
```

```
    for j in range(1, 1+num_inch):
```

```
        draw_interval(major_len-1)
```

```
        draw_line(major_len, str(j))
```

Recursion Trace

draw_interval(3)

→ draw_interval(2)

→ draw_interval(1)

→ draw_interval(0)

→ draw_line(1)

→ draw_interval(0)

→ draw_line(2)

→ draw_interval(1)

→ draw_interval(0)

→ draw_line(1)

→ draw_interval(0)

→ draw_line(3)

→ draw_interval(2)

etc