The Art of the Function

- Give each function exactly one job
- Don’t reapeat yourself (DRY).
- Don’t reapeat yourself (DRY).
- Define functions generally

Generalizing Over Computational Processes

The common structure among functions may itself be a computational process, rather than a number.

\[
\sum_{k=1}^{5} k = 1 + 2 + 3 + 4 + 5 = 15
\]

\[
\sum_{k=1}^{5} k^3 = 1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225
\]

\[
\sum_{k=1}^{5} \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{33} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04
\]

Generalizing Patterns with Parameters

Regular geometric shapes relate length and area.

Shape:

- \( r \)
- \( \frac{3\sqrt{3}}{2} r \)
- \( r \)

Area:

- \( r^2 \)
- \( \frac{3\sqrt{3}}{2} r^2 \)

Finding common structure allows for shared implementation

Functions as Arguments

Function values can be passed as arguments

\[
def \text{cube}(k):
    \text{return pow}(k, 3)
\]

A formal parameter that will be bound to a function

\[
\text{def \ summation}(n, \text{term})
    \text{***Sum the first \( n \) terms of a sequence.}
\]

The cube function is passed as an argument value

\[
\text{def \ summation}(5, \text{cube})\quad 225
\]

The function bound to term gets called here

\[
\text{total, k = 0, 1, while k <= n:}
    \text{total, k = total + \text{term}(k), k + 1}
\]

Function of a single argument (not called term)
Function Values as Parameters

Parameters can be bound to function values

Example: http://goo.gl/e4YBH

Functions as Return Values

Locally defined functions can be returned
They have access to the frame in which they are defined

A function that returns a function

```python
def make_adder(n):
    """Return a function that adds n to its argument."""
    def adder(k):
        return add(n, k)
    return adder
```

The name `add_three` is bound to a function

Call Expressions as Operators

An expression that evaluates to a function value

```python
make_adder(1)(2)
```

Higher-Order Functions

Functions are first-class: they can be manipulated as values in Python

Higher-order function: a function that takes a function as an argument value or returns a function as a return value

Higher order functions:
- Express general methods of computation
- Remove repetition from programs
- Separate concerns among functions