Announcements

- Quiz today!
  - Only worth two points, so don’t worry!

- Hog project
  - Get started early!
  - If you still don’t have a partner (and want one), find one on Piazza
    - Use existing post; don’t make a new one
Give each function exactly one job

Don’t reapeat yourself (DRY).

Define functions generally
Generalizing Patterns with Parameters

Regular geometric shapes relate length and area.

Shape:

Area:

Finding common structure allows for shared implementation.
Generalizing Over Computational Processes

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

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

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

\[
\sum_{k=1}^{5} \frac{8}{(4k-3) \cdot (4k-1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04
\]
def cube(k):
    return pow(k, 3)

def summation(n, term):
    '''Sum the first n terms of a sequence.'''
    total, k = 0, 1
    while k <= n:
        total, k = total + term(k), k + 1
    return total

>>> summation(5, cube)
225

0 + 1³ + 2³ + 3³ + 4³ + 5⁵
Function Values as Parameters

Parameters can be bound to function values

Example:

```python
1 def cube(k):
2     return pow(k, 3)

4 def summation(n, term):
5     total, k = 0, 1
6     while k <= n:
7         total, k = total + term(k), k + 1
8     return total
9
10 result = summation(5, cube)
```

Example: [http://goo.gl/e4YBH](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

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

>>> add_three = make_adder(3)
>>> add_three(4)
7
"""
```

A function that returns a function

The name add_three is bound to a function

A local def statement

Can refer to names in the enclosing function
Call Expressions as Operators

make_adder(1)(2)

make_adder(1)  (       2       )

Operator  Operand 0

An expression that evaluates to a function value

An expression that evaluates to any value

def make_adder(n):
    def adder(k):
        return add(n, k)
    return adder
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