Announcements
Iteration Example
def fib(n):
    """Compute the nth Fibonacci number, for N >= 1."""
    pred, curr = 0, 1  # Zeroth and first Fibonacci numbers
    k = 1  # curr is the kth Fibonacci number
    while k < n:
        pred, curr = curr, pred + curr
        k = k + 1
    return curr

The next Fibonacci number is the sum of the current one and its predecessor.
Discussion Question 1

What does pyramid compute?

```python
def pyramid(n):
    a, b, total = 0, n, 0
    while b:
        a, b = a + 1, b - 1
        total = total + a + b
    return total
```

\[
\begin{align*}
    &n^2 \\
    &(n + 1)^2 \\
    &2 \cdot (n + 1) \\
    &n^2 + 1 \\
    &n \cdot (n + 1)
\end{align*}
\]
Designing Functions
### Characteristics of Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Domain</th>
<th>Range</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>def square(x):</code></td>
<td>&quot;&quot;&quot;Return ( X \times X. )&quot;&quot;&quot;&quot;</td>
<td>( x ) is a real number</td>
<td>returns a non-negative real number</td>
<td>return value is the square of the input</td>
</tr>
<tr>
<td><code>def fib(n):</code></td>
<td>&quot;&quot;&quot;Compute the ( n )th Fibonacci number, for ( N \geq 1 ).&quot;&quot;&quot;&quot;</td>
<td>( n ) is an integer greater than or equal to 1</td>
<td>returns a Fibonacci number</td>
<td>return value is the ( n )th Fibonacci number</td>
</tr>
</tbody>
</table>
A Guide to Designing Function

Give each function exactly one job.

Don’t repeat yourself (DRY). Implement a process just once, but execute it many times.

Define functions generally.
Generalization
Generalizing Patterns with Arguments

Regular geometric shapes relate length and area.

Finding common structure allows for shared implementation

(Demo)
Higher-Order Functions
Generalizing Over Computational Processes

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

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

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

\[
\sum_{k=1}^{5} \frac{8}{(4k - 3) \cdot (4k - 1)} = \frac{8}{3} + \frac{8}{35} + \frac{8}{99} + \frac{8}{195} + \frac{8}{323} = 3.04
\] (Demo)
Summation Example

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

"""
Functions as Return Values

(Demo)
Locally Defined Functions

Functions defined within other function bodies are bound to names in a local frame.

A function that returns a function

```python
def make_adder(n):
    """Return a function that takes one argument k and returns k + n."
    def adder(k):
        return k + n
    return adder
```

The name `add_three` is bound to a function

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

A def statement within another def statement

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

An expression that evaluates to a function

An expression that evaluates to its argument

1

make_adder(1)

(func adder(k)

make_adder(1)

func make_adder(n)

def adder(k):
    return k + n

func adder(k)