Iteration Example

The Fibonacci Sequence

The next Fibonacci number is the sum of the current one and its predecessor

Discussion Question

Is this alternative definition of \texttt{fib} the same or different from the original \texttt{fib}?

```python
def fib(n):
    """Compute the n-th Fibonacci number, for \( n \geq 1 \).""
    pred, curr = 0, 1  # 0th and 1st Fibonacci numbers
    k = 1  # curr is the k-th Fibonacci number
    while k < n:
        pred, curr = curr, pred + curr
        k = k + 1
    return curr
```

Describing Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{fib}</td>
<td>( \text{Compute the } n \text{-th Fibonacci number, for } n \geq 1 )</td>
</tr>
<tr>
<td>\texttt{square}</td>
<td>( \text{Return } x \times x )</td>
</tr>
</tbody>
</table>

A function’s domain is the set of all inputs it might possibly take as arguments.

- \texttt{fib}: \( n \) is an integer greater than or equal to 1
- \texttt{square}: \( x \) is a real number

A function’s range is the set of output values it might possibly return.

- \texttt{fib}: returns a Fibonacci number
- \texttt{square}: returns a non-negative real number

A pure function’s behavior is the relationship it creates between input and output.

- \texttt{fib}: return value is the n-th Fibonacci number
- \texttt{square}: return value is the square of the input

A Guide to Designing Functions

Give each function exactly one job, but make it apply to many related situations

```python
>>> round(1.23)
1
>>> round(1.23, 0)
1
>>> round(1.23, 5)
1.23
```

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

Regular geometric shapes relate length and area.

Shape:

Area: \[ \frac{1}{3} \pi r^3 \]

Finding common structure allows for shared implementation

Higher-Order Functions

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

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

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

\[
\sum_{k=1}^{5} \frac{8}{k \cdot 4 - 3} \cdot \frac{8}{k \cdot 4 - 1} = 3.04
\]

Summation Example

```python
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
```

Functions as Return Values

```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
def compose1(f, g):
    """Return a function that composes f and g."
    def h(x):
        return f(g(x))
    return h
```

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 lambda k: k + n
```

Call Expressions as Operator Expressions

An expression that evaluates to a function

An expression that evaluates to its argument