Higher-Order Functions

Designing Functions

Describing Functions

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

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

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

A Guide to Designing Function

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

>>> round(1.23)   >>> round(1.23, 1)   >>> round(1.23, 0)   >>> round(1.23, 5)
1.2              1.2                 1.2                 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.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>circle</td>
<td>$\frac{1}{2} \pi r^2$</td>
</tr>
<tr>
<td>triangle</td>
<td>$\frac{1}{2}r^2$</td>
</tr>
<tr>
<td>pentagon</td>
<td>$\frac{5}{2}sr^2$</td>
</tr>
</tbody>
</table>

Finding common structure allows for shared implementation

Higher-Order Functions

Generalizing Patterns with Arguments

(Demo)
Lambda Expressions

Generalizing Over Computational Processes

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

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

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

\[ \sum_{k=1}^{n} (k-1) = 0 + 1 + 2 + 3 + 4 + 5 = 15 \]

Call Expressions as Operator Expressions

Lambda expressions in Python cannot contain statements at all!

Lambda expressions are not common in Python, but important in general

```
func make_adder(n)
    return lambda k: k + n
```

Functions as Return Values

```
def square(x):
    return x * x
```

```
square = lambda x: x * x
```

```
>>> square = x * x
```

```
>>> x = 10
```

```
An expression: this one evaluates to a number
```

```
An expression: evaluates to its argument
```

```
n = 10
```

```
10
```

```
func make_adder(n):
    return lambda k: k + n
```

```
func adder(k):
    return k + n
```

```
add_three = make_adder(3)
```

```
add_three(10)
```

```
A function that returns a function
```

```
A function that takes one argument k and returns k + n.
```

Locally Defined Functions

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

```
def make_adder(n):
    return lambda k: k + n
```

```
add_three = make_adder(3)
```

```
add_three(10)
```

```
A formal parameter that will be bound to a function
```

```
A function that returns a function
```

```
A function that takes one argument k and returns k + n.
```

```
A def statement within another def statement
```

```
Can refer to names in the enclosing function
```

Lambda Expressions

```
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 = total + term(k)
        k += 1
    return total
```

```
>>> summation(5, (k, pow(k, 3)))
```

```
>>> summation(5, (k, cube(k)))
```

```
>>> add_three = make_adder(3)
```

```
add_three(10)
```

```
Functions of a single argument
```

```
Can refer to names in the enclosing function
```

Lambda Expressions Versus Def Statements

```
def square(x):
    return x * x
```

```
square = lambda x: x * x
```

```
lambda ex: x * x
```

```
Lambda expressions are not common in Python, but important in general
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions are not common in Python, but important in general
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```

```
Lambda expressions in Python cannot contain statements at all!
```