

Functions

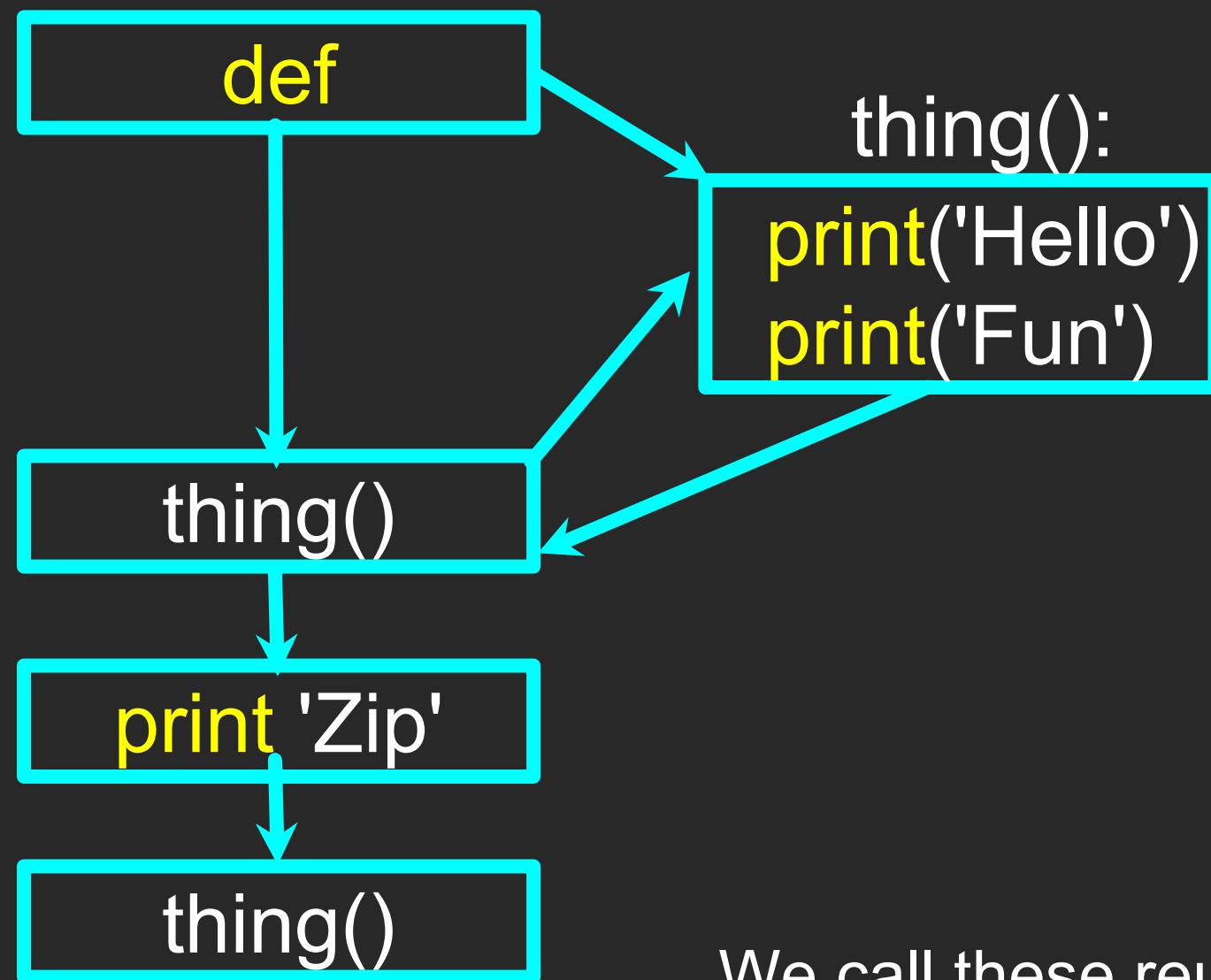
Chapter 4



Python for Everybody
www.py4e.com



Stored (and reused) Steps



Program	Output
<code>def thing(): print('Hello') print('Fun')</code>	
<code>thing()</code>	Hello Fun
<code>print('Zip')</code>	Zip
<code>thing()</code>	Hello Fun

We call these reusable pieces of code “functions”



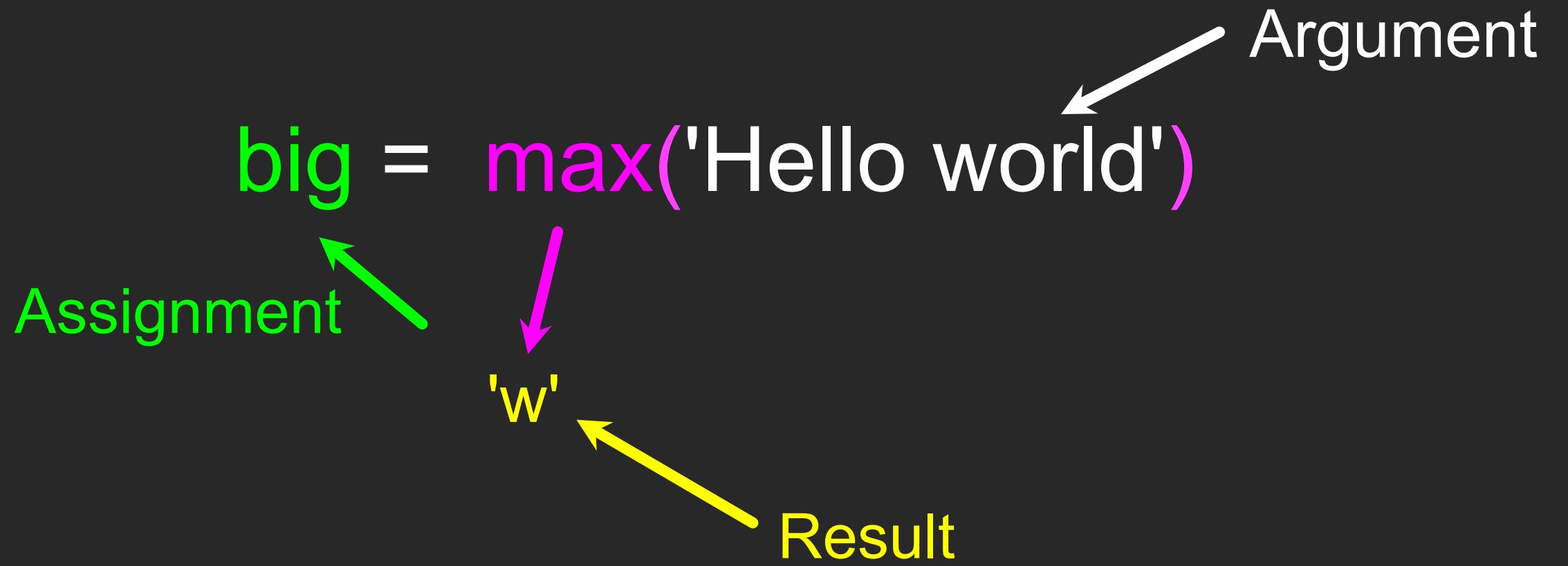
Python Functions

- There are two kinds of functions in Python.
 - **Built-in functions** that are provided as part of Python - `print()`, `input()`, `type()`, `float()`, `int()` ...
 - **Functions that we define ourselves** and then use
- We treat the built-in function names as “new” **reserved words** (i.e., we avoid them as variable names)



Function Definition

- In Python a **function** is some reusable code that takes **arguments**(s) as input, does some computation, and then returns a result or results
- We define a **function** using the **def** reserved word
- We call/invoke the **function** by using the function name, parentheses, and **arguments** in an expression



```
>>> big = max('Hello world')
>>> print(big)
w
>>> tiny = min('Hello world')
>>> print(tiny)
>>>
```

Max Function

```
>>> big = max('Hello world')  
>>> print(big)
```

w



A **function** is some stored code
that we use.

A function takes some **input** and
produces an **output**.

Guido wrote this code

Max Function

```
>>> big = max('Hello world')
>>> print(big)
```

w

'Hello world'
(a string)

A function is some stored code
that we use.

A function takes some input and
produces an output.

```
def max(inp):
    blah
    blah
    for x in inp:
        blah
        blah
```

'w'
(a string)

Guido wrote this code



Type Conversions

- When you put an integer and floating point in an expression, the integer is **implicitly** converted to a float
- You can control this with the built-in functions `int()` and `float()`

```
>>> print float(99) / 100
0.99
>>> i = 42
>>> type(i)
<class 'int'>
>>> f = float(i)
>>> print(f)
42.0
>>> type(f)
<class 'float'>
>>> print(1 + 2 * float(3) / 4 - 5)
-2.5
>>>
```

String Conversions

- You can also use `int()` and `float()` to convert between strings and integers
- You will get an `error` if the string does not contain numeric characters

```
>>> sval = '123'  
>>> type(sval)  
<class 'str'>  
>>> print(sval + 1)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
TypeError: cannot concatenate 'str'  
and 'int'  
>>> ival = int(sval)  
>>> type(ival)  
<class 'int'>  
>>> print(ival + 1)  
124  
>>> nsv = 'hello bob'  
>>> niv = int(nsv)  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
ValueError: invalid literal for int()
```



A Function of Our Own



Acknowledgements / Contributions



These slides are Copyright 2010- Charles R. Severance
(www.dr-chuck.com) of the University of Michigan School of Information and open.umich.edu and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

...

Initial Development: Charles Severance, University of Michigan School of Information

... Insert new Contributors and Translators here