BU CodeBreakers 2017

Python II: Functions, Libraries, Random Numbers

Today’s Schedule

09:00-10:15	Python II
10:30 - 11:45	Exercise II
01:00 - 01:45	Python III
02:00 - 03:00	Exercise III

We’re going climbing tomorrow so please dress comfortably

Review I: Guess the output

```
>>> month = ‘11’
>>> month = month + 1
TypeError: Can't convert 'int' object to str implicitly

>>> month = (int(month) + 1) % 2
>>> print (month)
0

>>> print (‘month’)
month

>>> print (day)
NameError: name 'day' is not defined

>>> print (‘day’)
day
```

Review II: Guess the output

```
>>> numbers = [2, 4, 8]
>>> numbers [-2 : 2]
[4]

>>> numbers [-2 : 1]
[]

>>> month = ‘December’
>>> ‘de’ in month
False

>>> list( ‘abc’ + str (123) )
['a', 'b', 'c', '1', '2', '3']

>>> month[0:2] = ‘de’
TypeError: 'str' object does not support item assignment
```

Running Python

- What are the 2 modes in which you can run python?

- Interactive Mode

  >>> Command

  Output

- **Python Script/Program → Collection/lines of commands (*.py)**

Running Python scripts

1. Open Python **IDLE** GUI
2. File > New File
3. Write a program named **helloworld.py** that prints “Hello World!!!”

   Hint: print (‘what should I print?’)
4. Save the program in your directory:

   Classroom > codebreakers > students > user_name
5. Run the program

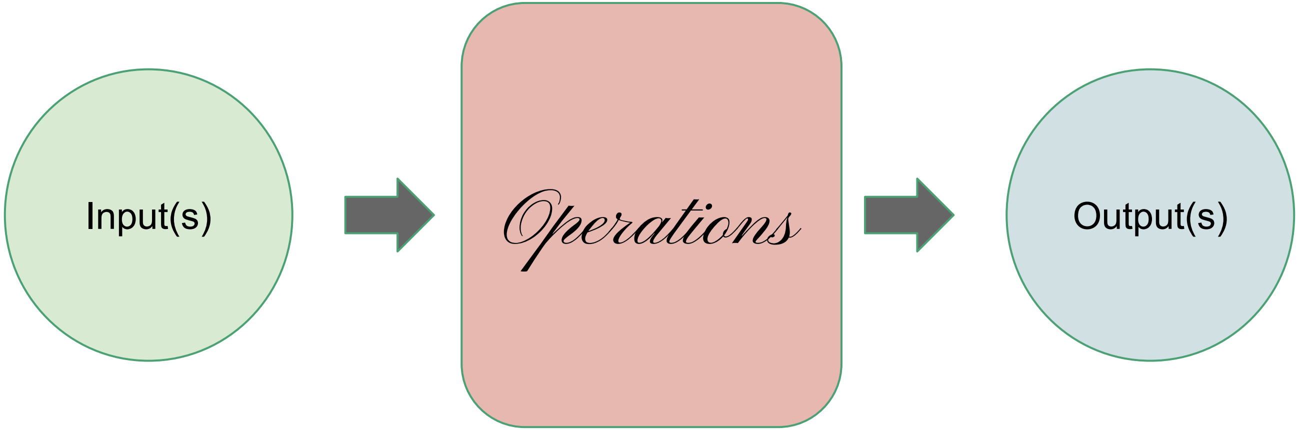
   Run> Run Module OR <F5>

What is a function?

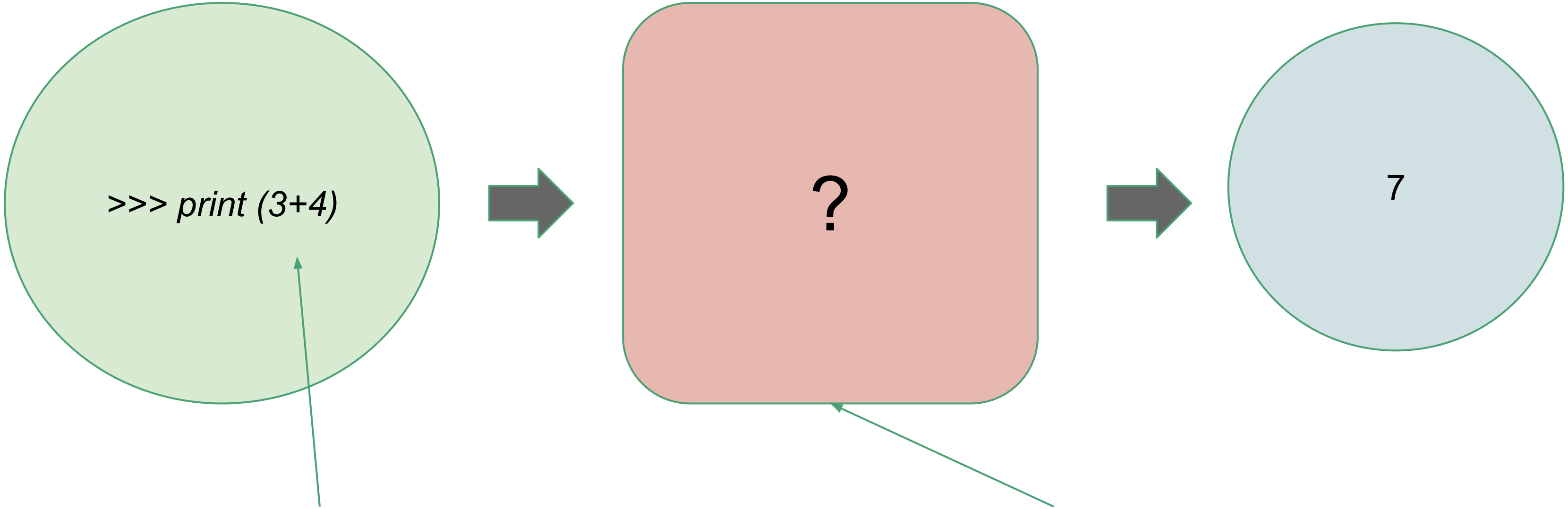
Function

- Sequence of statements/code to perform an operation on inputs to certain outputs
- *Syntax:*

return_value/function_output = function_name(function_inputs)



print() Function I



>>> print (3+4)

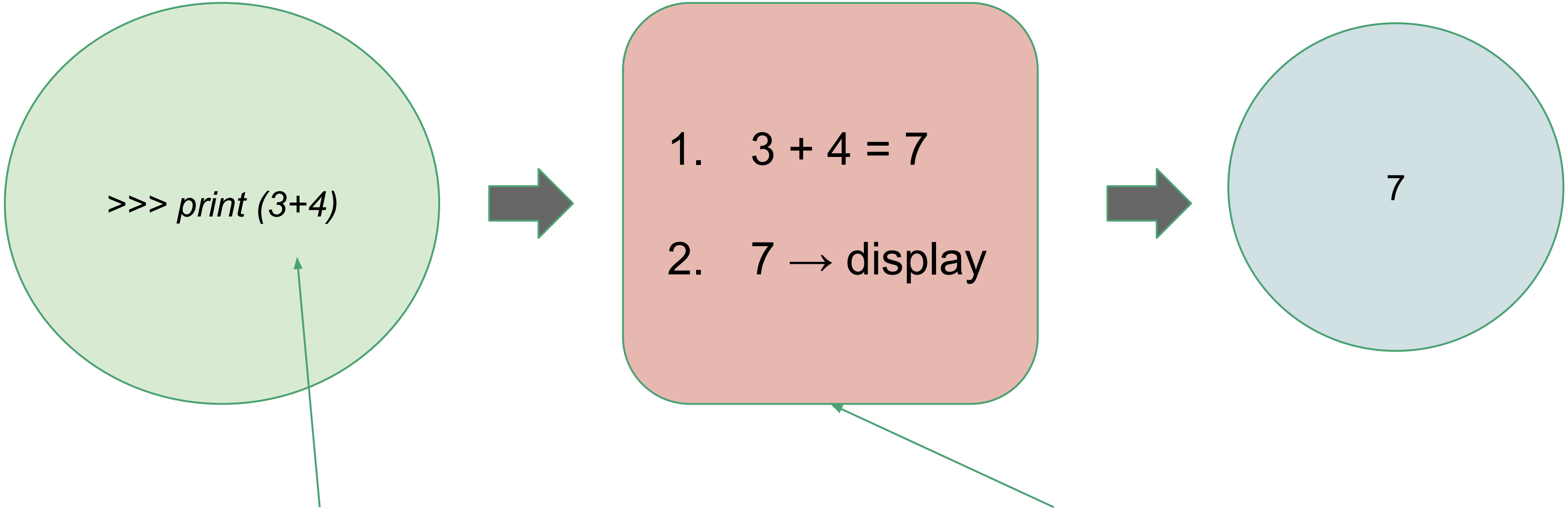
?

7

How many values are you passing into print?

What operations happen here?

print() Function I



>>> print (3+4)

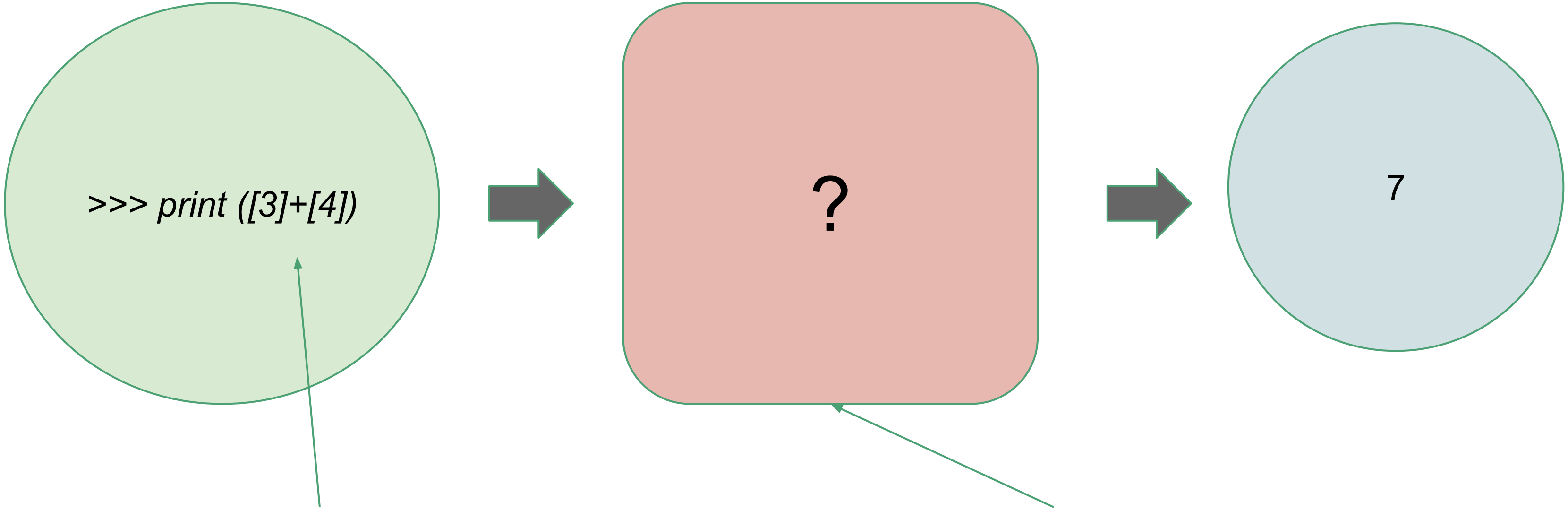
1. 3 + 4 = 7
2. 7 → display

7

How many values are you passing into print? = 1

What operations happen here?

print() Function II



>>> print ([3]+[4])

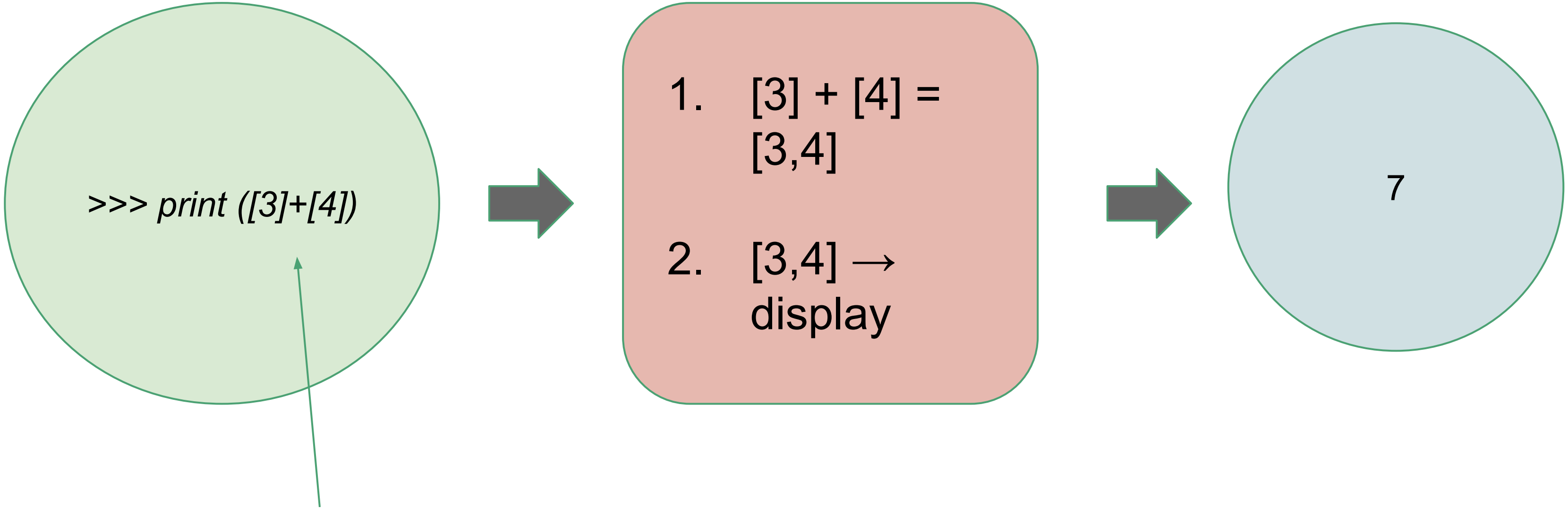
?

7

How many values are you passing into print?

What operations happen here?

print() Function II



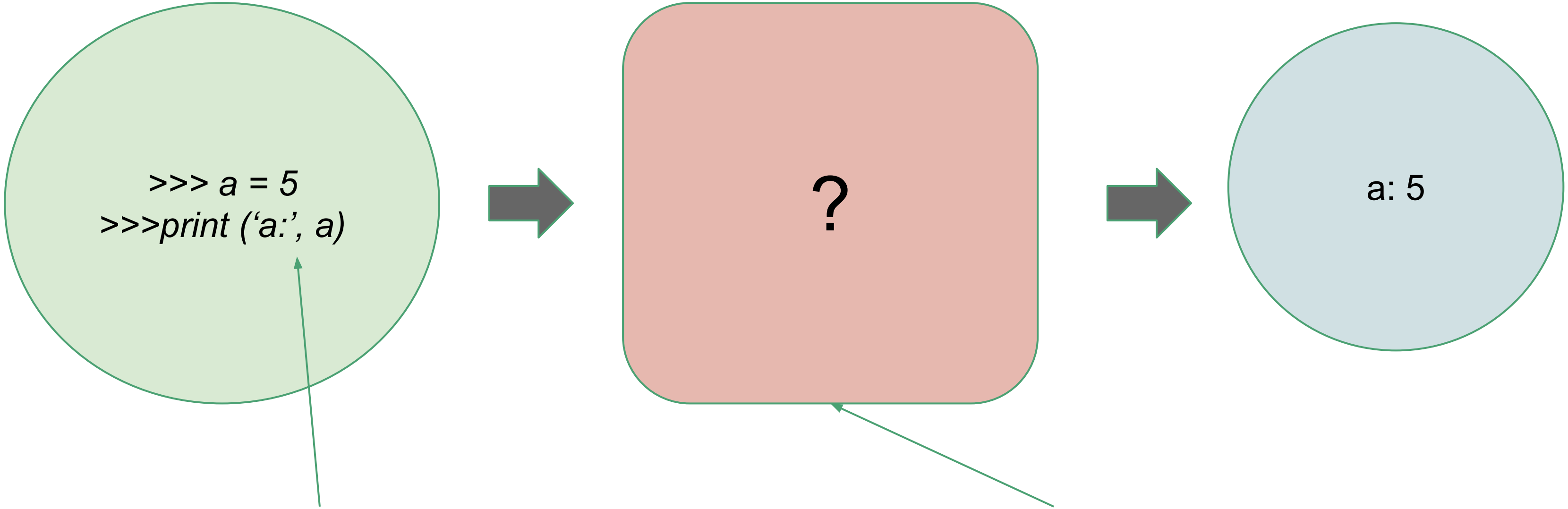
>>> print ([3]+[4])

1. [3] + [4] = [3,4]
2. [3,4] → display

7

How many values are you passing into print? = 1

print() Function III



```
>>> a = 5
>>>print (‘a:’, a)
```

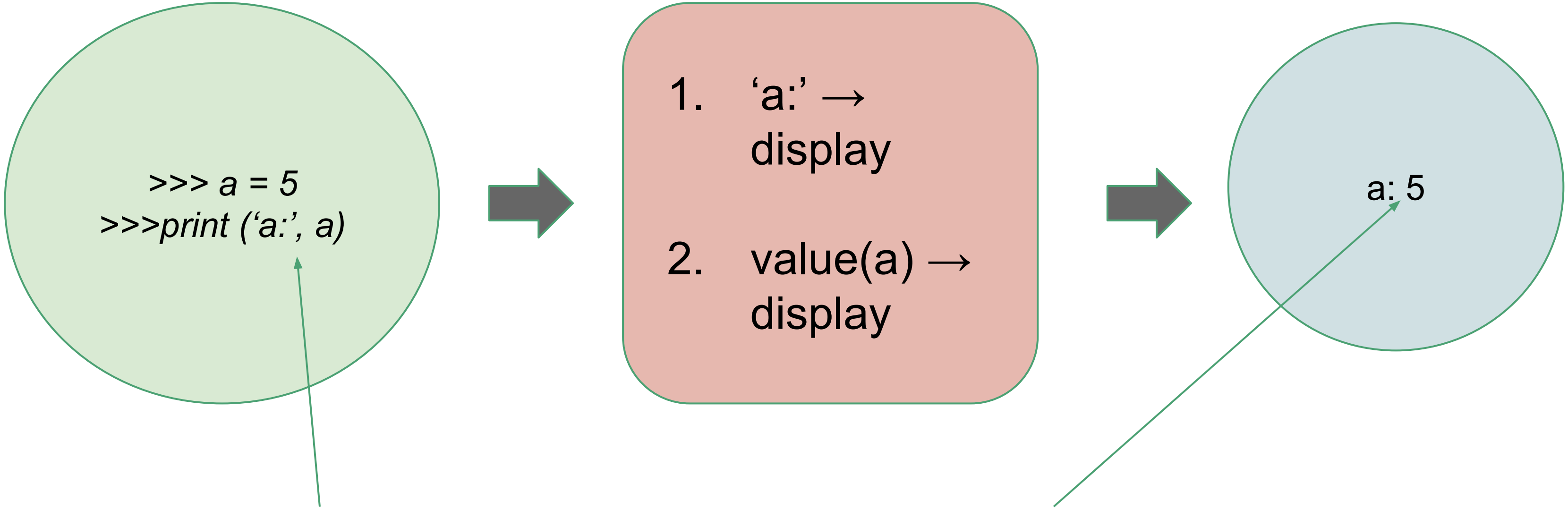
```
a: 5
```

How many values are you passing into print?

What operations happen here?

print() Function III

- *Multiple inputs/outputs are separated by commas*



```
>>> a = 5
>>>print (‘a:’, a)
```

1. ‘a:’ → display
2. value(a) → display

```
a: 5
```

How many values are you passing into print? = 2

Default space

Guess the output

```
>>> my_string = ‘Code-Breakers’
>>> print (‘My string is’, my_string)
```

```
My string is Code-Breakers
```

```
>>> print (‘My string is’, ‘my_string’)
```

```
My string is my_string
```

Guess the output

```
>>> out = print (3 + 3*2)

9



>>> print (out)
```

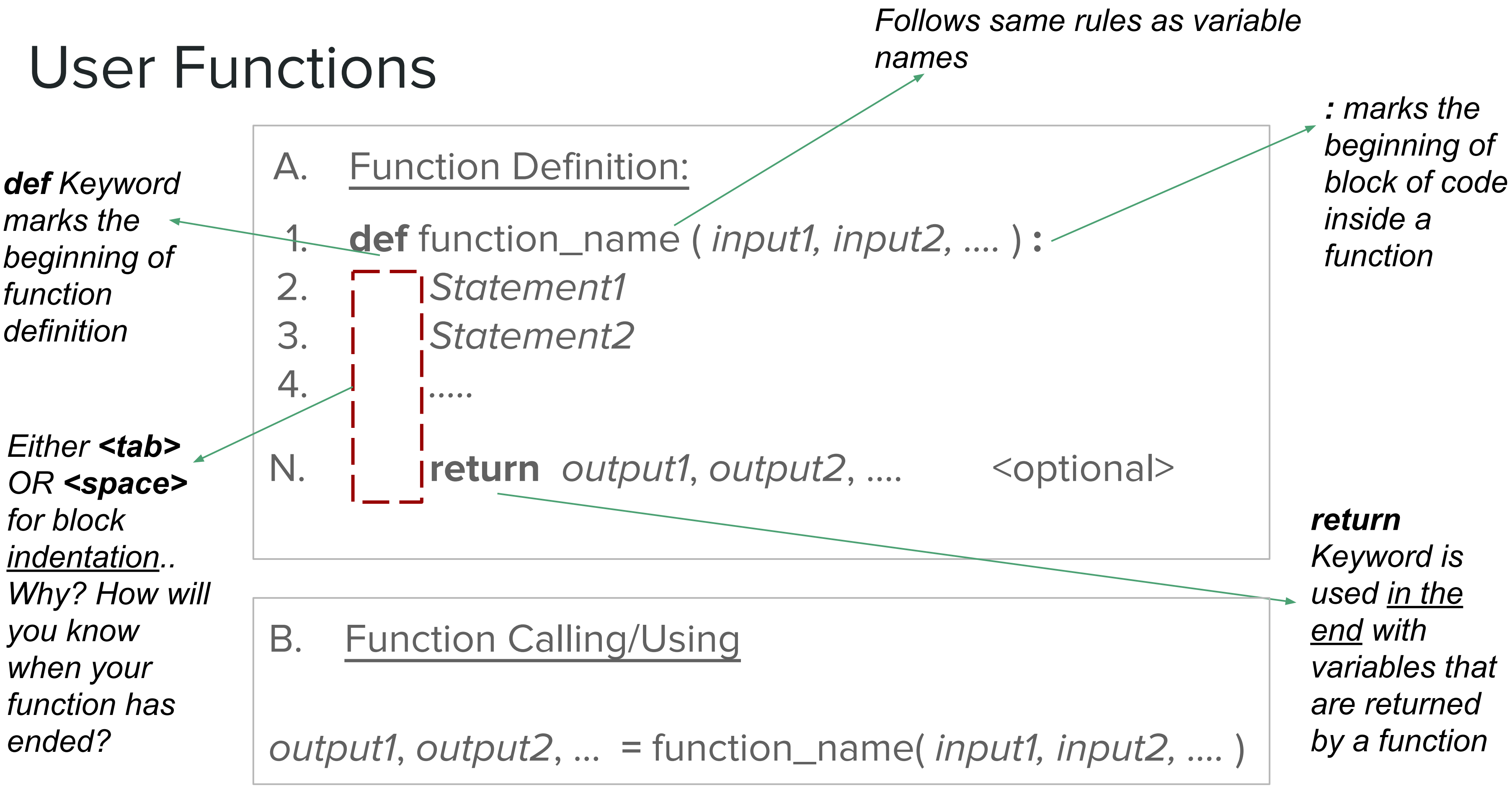
Why? *→ print() doesn’t return any value, it only displays its input values*

```
None

>>> out = len ( ‘string’ )
>>> print (out)

5
```

Why? *→ len() returns the length of the string*

How to write your own functions?

User Functions



Follows same rules as variable names

***def** Keyword marks the beginning of function definition*

*Either **<tab>** OR **<space>** for block <u>indentation</u>.. Why? How will you know when your function has ended?*

A. <u>Function Definition:</u>

1. **def** function_name (*input1, input2, ...*) **:**
2. *Statement1*
3. *Statement2*
4. .....

N. **return** *output1*, *output2*, ... <optional>

<b>:</b> marks the beginning of block of code inside a function

***return** Keyword is used <u>in the end</u> with variables that are returned by a function*

B. <u>Function Calling/Using</u>

output1, *output2*, ... = function_name(*input1, input2*, ...)

Example

```
>>> def my_function(x):
...     print ( x[0] + x[1] )
...
```

Q) What does my_function do?

Prints the sum/concatenation of 1st 2 elements of input

```
>>> my_function( [1, 3, 6, 10] )
4

>>> my_function( 'hello' )
he

>>> my_function( ['hello', '-bye'] )
hello-bye
```

Return Value?

→ *None*

<u>BUT</u>

```
>>> def my_function(x):
...     return ( x[0] + x[1] )
...
```

Return Value?

Commenting

Single Line:

```
# Function to print sum
# of 1st 2 elements
```

Multiple Lines: 3 single quotation mark

```
’’’ Function to return the
sum of 1st 2 input elements ’’’
```

Exercise (~3 mins)

Implement and test following functions on IDLE Python Shell

Don’t forget about indentation

```
def sum1(x):
        print ( x[0] + x[1] )

def sum2(x):
        print ( x[0] + x[1] )
        return ( x[0] + x[1] )
```

What are built-in functions? Examples?

LEN(), PRINT(), STR(), INT(), TYPE(),....

Where are the built-in functions?

Python Standard Library (PSL)

What else is in the Python Standard library?

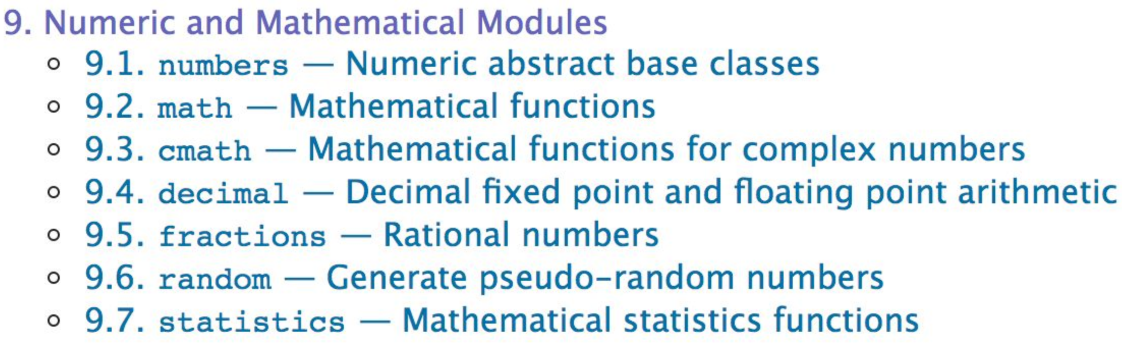
Python Standard Library

- Contains all built-in and installed object/data types and functions in Python
- What are some of the examples of built-in datatypes?
- What are some examples of built-in functions?
- More: https://docs.python.org/3/library/index.html

Modules

Python program in PSL that aren’t loaded by default and need to be **imported**

1. Open: https://docs.python.org/3/library/index.html
2. Look at #9



9. Numeric and Mathematical Modules
   - 9.1. numbers — Numeric abstract base classes
   - 9.2. math — Mathematical functions
   - 9.3. cmath — Mathematical functions for complex numbers
   - 9.4. decimal — Decimal fixed point and floating point arithmetic
   - 9.5. fractions — Rational numbers
   - 9.6. random — Generate pseudo-random numbers
   - 9.7. statistics — Mathematical statistics functions

3. What does it contain?

Using modules

1. (i) Import module:

   ```
   >>> import module_name
   ```

   (ii) Now you can use any functions in the module using:

   ```
   >>> module_name.module_function()
   ```

2. (i) Import only required function from the module:

   ```
   >>> from module_name import module_function_name
   ```

   (ii) Now you can use the imported function directly

   ```
   >>> module_function()
   ```

How to use random module to generate random numbers in python?

Random Module: Example

1.
```
>>> import random
>>> random.random()
```

2.
```
>>> from random import random
>>> random()
```

Exercise (~3min)

Use *factorial()* function inside **math** module to print factorial of 5 using both imports

1.
```
>>> import math
>>> print ( math.factorial(5) )
```

2.
```
>>> from math import factorial
>>> print ( factorial(5) )
```

In *math.factorial(x)*, what is **math**? What is **factorial()**? What is **x**?

In random.random(), what is **random**? What is **random**()? What does it do?

More Random numbers

- Generate random numbers/floats between 0-10

```
>>> random.random() * 10
```

- Generate random integers between 0-10

```
>>> int ( random.random() * 10 )
```

Review Questions

1. What does Python Standard Library contain?
   ➜ all built-in and installed functions/data-types and modules
2. What is the difference between built-in types vs. non-built in types in Python?
   ➜ built-in types can be used directly without **import**ing
3. What are modules?
   ➜ Python files with specialized functions and/or types definition
4. Can you give an example of a module?
   ➜ math, random
5. What does random.random() do?
   ➜ returns random float between 0-1

Exercise II