Think Python:
Chapter 3 Review

What does the code below output?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

What does the code below output?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

```
Hello! Hello! Hello! Hello! Hello!
```

What is the term for the "Hello! " and the 5 in the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

What is the term for the “Hello! “ and the 5 in the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

Arguments

What is the term for *msg* and *num_times* in the first line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

What is the term for *msg* and *num_times* in the first line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
```

Parameters

What is output by the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
Hello! Hello! Hello! Hello! Hello!
>>> print(msg)
>>>
```

What is output by the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
Hello! Hello! Hello! Hello! Hello!
>>> print(msg)
>>>
```

```
>>> print(msg)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'msg' is not defined
>>>
```

Why did we get an error message from executing the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
Hello! Hello! Hello! Hello! Hello!
>>> print(msg)
>>>
```

Why did we get an error message from executing the last line of code below?

```
>>> def print_msg(msg,num_times):
...     print(msg*num_times)
...
>>> print_msg("Hello! ",5)
Hello! Hello! Hello! Hello! Hello!
>>> print(msg)
>>>
```

Because parameters like ***msg*** and ***num_times*** are, like **local variables**, local to the function that contains them.

What is output by the code below?

```
>>> string = "Hush, hush sweet Charlotte!"
>>> print(print(string))
```

What is output by the code below?

```
>>> string = "Hush, hush sweet Charlotte!"
>>> print(print(string))
```

```
Hush, hush sweet Charlotte!
None
```

None is the value returned by a **void function**. **print()** is a void function.

What is output by the code below?

```
>>> def print_five_strings(s1, s2, s3, s4, s5):
...   print(s5, s4, s3, s2, s1)
...
>>> print_five_strings("penultimate", "means",
"next", "to", "last")
```

What is output by the code below?

```
>>> def print_five_strings(s1, s2, s3, s4, s5):
...   print(s5, s4, s3, s2, s1)
...
>>> print_five_strings("penultimate", "means",
"next", "to", "last")
```

```
last to next means penultimate
```

What two things have to be done in order to use a function from Python’s math module?

What two things have to be done in order to use a function from Python’s math module?

import math

Prepend **math.** to every call to a math module function, i.e., **math.floor(15.75)**

Which function can be used to convert its argument into an integer?

Which function can be used to convert its argument into an integer?

int

What is the value of the expression below?

```
>>> int(123.456)
```

What is the value of the expression below?

```
>>> int(123.456)
```

123

The **int** function truncates its floating-point argument in order to return an integer.

What is the value of the expression below?

```
>>> int('123')
```

What is the value of the expression below?

```
>>> int('123')
```

123

What is the value of the expression below?

```
>>> int(‘abc’)
```

What is the value of the expression below?

```
>>> int(‘abc’)
```

```
>>> int(‘abc’)
  File "<stdin>", line 1
    int(‘abc’)
            ^
SyntaxError: invalid character in identifier
>>>
```

What is the value of the expression below?

```
>>> int(‘abc’)
```

Curly (“smart”) quotes are NOT allowed in Python code!

Smart single quotes: ‘’

Plain single quotes: ''

What is the value of the expression below?

```
>>> int('abc')
```

What is the value of the expression below?

```
>>> int('abc')
```

```
>>> int('abc')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10:
'abc'
>>>
```

What is the value of the expression below?

```
>>> int('abc')
```

Quotation from your book at the beginning of Chapter 3: “The int function takes any value and converts it to an integer, if it can, or complains otherwise.”

In this case, a complaint was issued!

What function is used to convert from an integer to a floating point number?

What function is used to convert from an integer to a floating point number?

float

```
>>> float(145)
145.0
>>>
```

What function is used to convert from an integer to a string?

What function is used to convert from an integer to a string?

str

```
>>> str(14.67)
'14.67'
>>>
```

What (if anything) is output by the code below?

```
>>> def mystery():
...   print(26**2)
```

What (if anything) is output by the code below?

```
>>> def mystery():
... print(26**2)
```

```
  File "<stdin>", line 2
    print(26**2)
        ^
IndentationError: expected an indented block
>>>
```

What (if anything) is output by the code below?

```
>>> def mystery2
```

What (if anything) is output by the code below?

```
>>> def mystery2




  File "<stdin>", line 1
    def mystery2
               ^
SyntaxError: invalid syntax
>>>
```

What (if anything) is output by the code below?

```
>>> def mystery2
```

Function definitions have to end with a (possibly empty) set of parenthesis around the function's parameters and then a colon (:).

What is output by the last line of code below?

```
>>> def print_msg(msg,num_times):
...     result = msg*num_times
...     print(result)
...
>>> print_msg("Hello! ",3)
Hello! Hello! Hello!
>>> print(result)
```

What is output by the last line of code below?

```
>>> def print_msg(msg,num_times):
...     result = msg*num_times
...     print(result)
...
>>> print_msg("Hello! ",3)
Hello! Hello! Hello!
>>> print(result)
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'result' is not defined
```

What is output by the last line of code below?

```
>>> def print_msg(msg,num_times):
...     result = msg*num_times
...     print(result)
...
>>> print_msg("Hello! ",3)
Hello! Hello! Hello!
>>> print(result)
```

result is a local variable of function *print_msg*. This means it's only available inside that function, not outside it.