None Indicates that Nothing is Returned

The special value None represents nothing in Python. A function that does not explicitly return a value will return None. Careful: None is not displayed by the interpreter as the value of an expression.

```python
>>> def does_not_return_square(x):
    ... x * x
    ...

>>> does_not_return_square(4)

>>> sixteen = does_not_return_square(4)

>>> sixteen + 4
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    TypeError: unsupported operand type(s) for +: 'NoneType' and 'int'
```

No return None value is not displayed.

Pure Functions & Non-Pure Functions

- Pure Functions: just return values
- Non-Pure Functions: have side effects

Nested Expressions with Print

```python
None, None
```

Life Cycle of a User-Defined Function

Def statement: Name bound to that function in the current frame

Call expression: Function (value of operator) called on arguments (values of operands)

Calling/Rejecting: A new frame is created! Parameters bound to arguments

Multiple Environments

Print and None
Multiple Environments in One Diagram!

An environment is a sequence of frames.

- The global frame alone
- A local, then the global frame

Interactive Diagram

Names Have No Meaning Without Environments

Every expression is evaluated in the context of an environment.

- The global frame alone
- A local, then the global frame

Interactive Diagram

Names Have Different Meanings in Different Environments

A call expression and the body of the function being called are evaluated in different environments.

Every expression is evaluated in the context of an environment.

A name evaluates to the value bound to that name in the earliest frame of the current environment in which that name is found.

Interactive Diagram

Miscellaneous Python Features

- Division
- Multiple Return Values
- Source Files
- Doctests
- Default Arguments
- (Demo)

Interactive Diagram

Statements

A statement is executed by the interpreter to perform an action

Conditional Statements

The first header determines a statement’s type

The header of a clause "controls" the suite that follows

def statements are compound statements

Interactive Diagram
Compound Statements

A suite is a sequence of statements.

To "execute" a suite means to execute its sequence of statements, in order.

Execution Rule for a sequence of statements:
• Execute the first statement
• Unless directed otherwise, execute the rest

Conditional Statements

def absolute_value(x):
    """Return the absolute value of x."""
    if x < 0:
        return -x
    elif x == 0:
        return 0
    else:
        return x

Execution Rule for Conditional Statements:
1. Always starts with "if" clause.
2. Zero or more "elif" clauses.
3. Zero or one "else" clause, always at the end.

Syntax Tips:
1. Always starts with "if" clause.
2. Zero or more "elif" clauses.
3. Zero or one "else" clause, always at the end.

Boolean Contexts

False values in Python:
False, 0, '', None

True values in Python:
Anything else (True)

(more to come)

Read Section 1.5.4!

Iteration

While Statements

Execution Rule for While Statements:
1. Evaluate the header's expression.
2. If it is a true value, execute the (whole) suite, then return to step 1.