Recall - Week 1

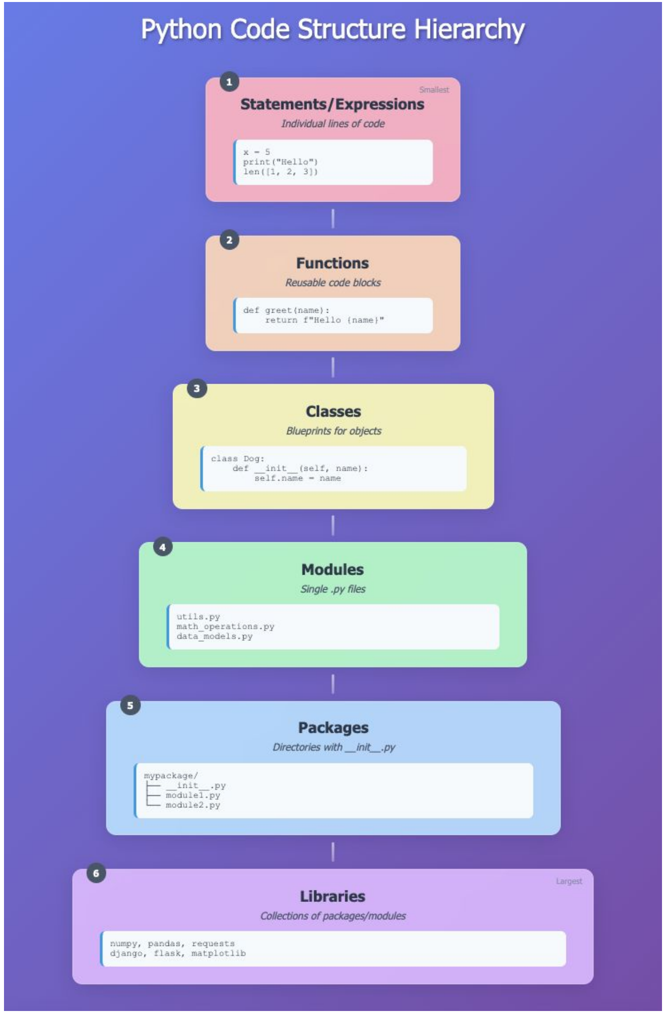
You should be able to:

- Set up a local Python development environment using VSCode.
- Write basic Python programs to use and manipulate the primitive data types (numerical, string, None)
- Apply valid name binding and related conventions to Python objects
- Build expressions and statements using Python operators (Arithmetic, Boolean, Relational, Logical) that accord to operator precedence

Week 2

You should be able to:

- Write Python functions to solve a problem
- Distinguish between global and local scope as well as namespace
- Pass positional, keyword, and default arguments to functions in proper order and with proper syntax
- Orchestrate file execution and module imports using “if __name__ == ‘__main__:”



Python Code Structure Hierarchy

1 Statements/Expressions

Smallest

Individual lines of code

```
x = 5
print("Hello")
len([1, 2, 3])
```

2 Functions

Reusable code blocks

```
def greet(name):
    return f"Hello {name}"
```

3 Classes

Blueprints for objects

```
class Dog:
    def __init__(self, name):
        self.name = name
```

4 Modules

Single .py files

```
utils.py
math_operations.py
data_models.py
```

5 Packages

Directories with __init__.py

```
mypackage/
├── __init__.py
├── module1.py
└── module2.py
```

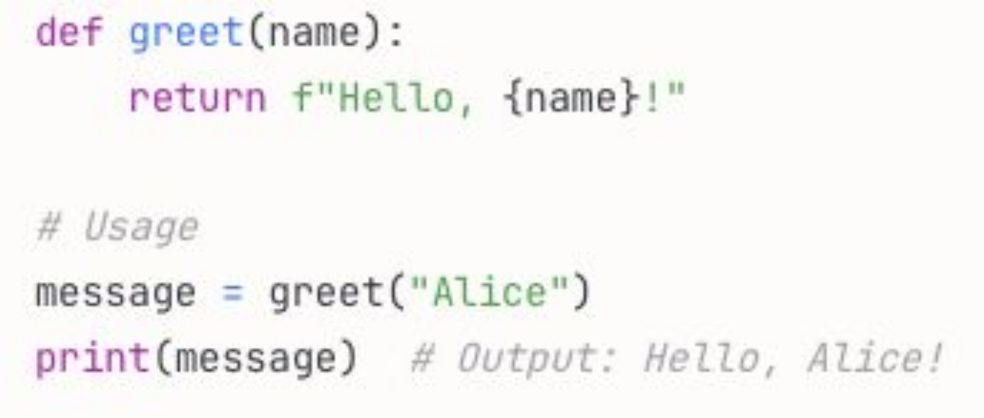
6 Libraries

Largest

Collections of packages/modules

```
numpy, pandas, requests
django, flask, matplotlib
```

Python Function - a block of reusable code that performs a specific, well-defined task

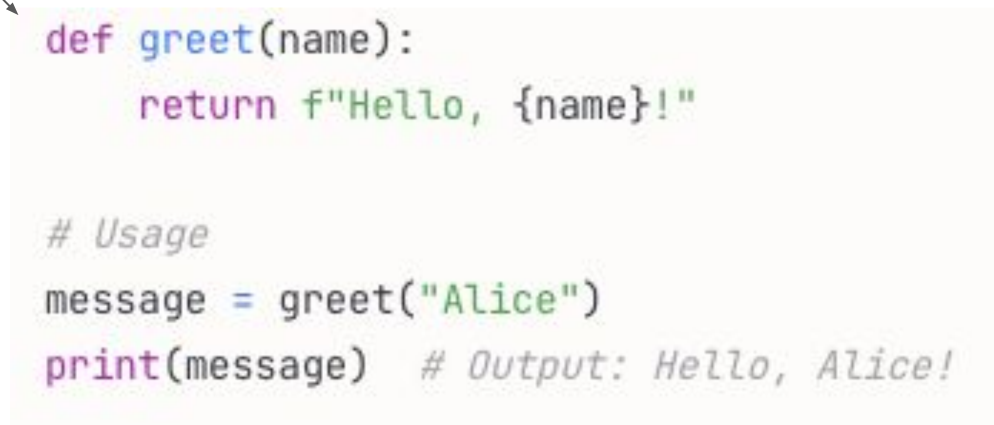


```python
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Python Function

Definition



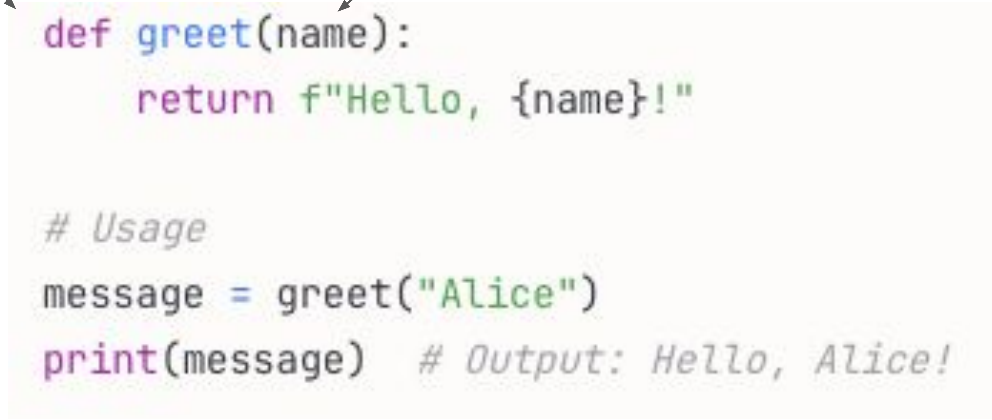
```python
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Python Function

Definition

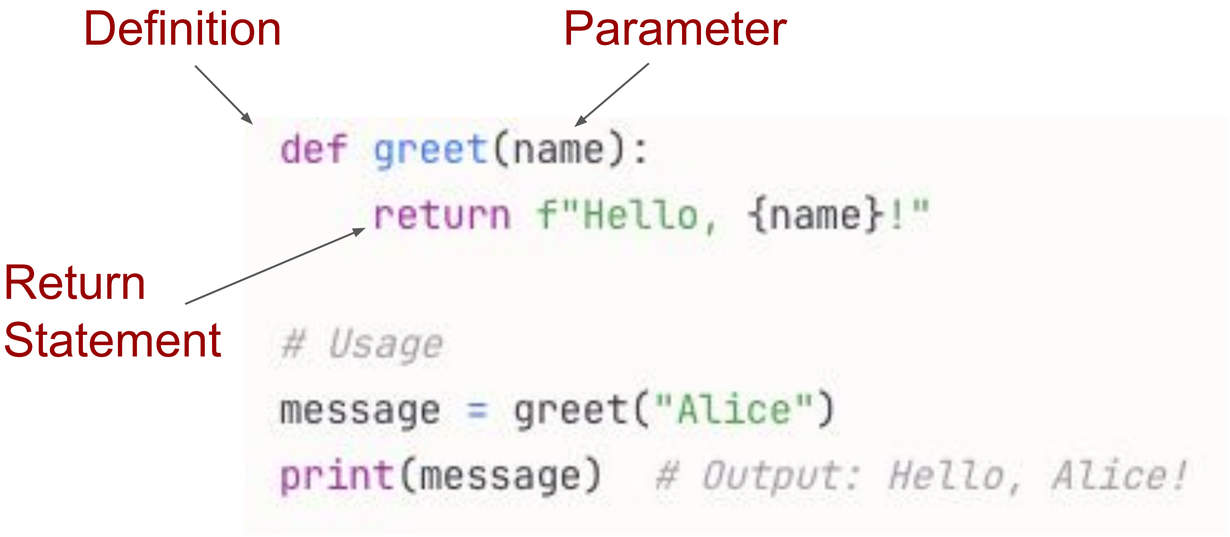
Parameter



```python
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Python Function



Definition

Parameter

Return Statement

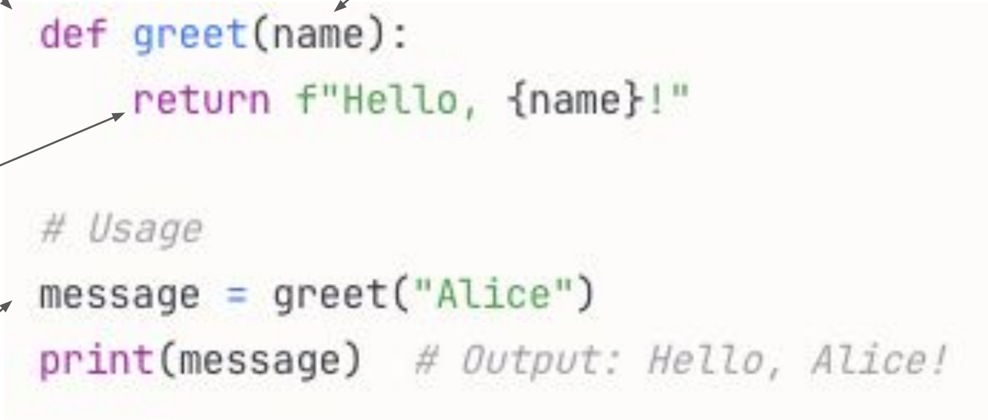
```python
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Python Function

Definition

Parameter



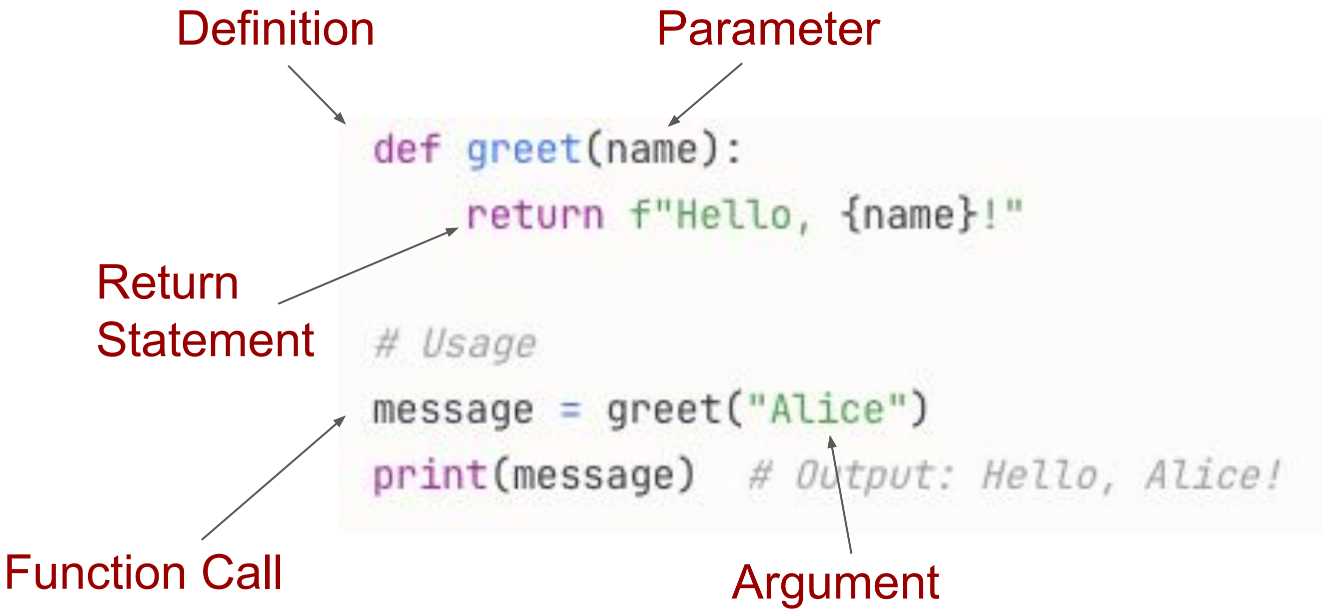
```python
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Return Statement

Function Call

Python Function



```
def greet(name):
    return f"Hello, {name}!"

# Usage
message = greet("Alice")
print(message)  # Output: Hello, Alice!
```

Definition

Parameter

Return Statement

Function Call

Argument

Week 1

You should also be able to:

- Distinguish between Python modules, packages, and libraries
- Identify and explain what a Python object is as well as describe object properties
- Discuss how Python allocates memory to object data (immutable types)

Review Questions

1. Are x and y equal?

x = 0.1 + 0.1 + 0.1
y = 0.3

Review Questions

2. Are these statements True or False?

x == y → this is a measure of object identity

x is y → this is a measure of value equality

Review Questions

2. In what order will the following calculation be performed?

- final_rating = baseline_score + follow_up_score / 2 * num_sessions % 5

Review Questions

2. In what order will the following calculation be performed?

- final_rating = (baseline_score + follow_up_score) / 2 * num_sessions % 5