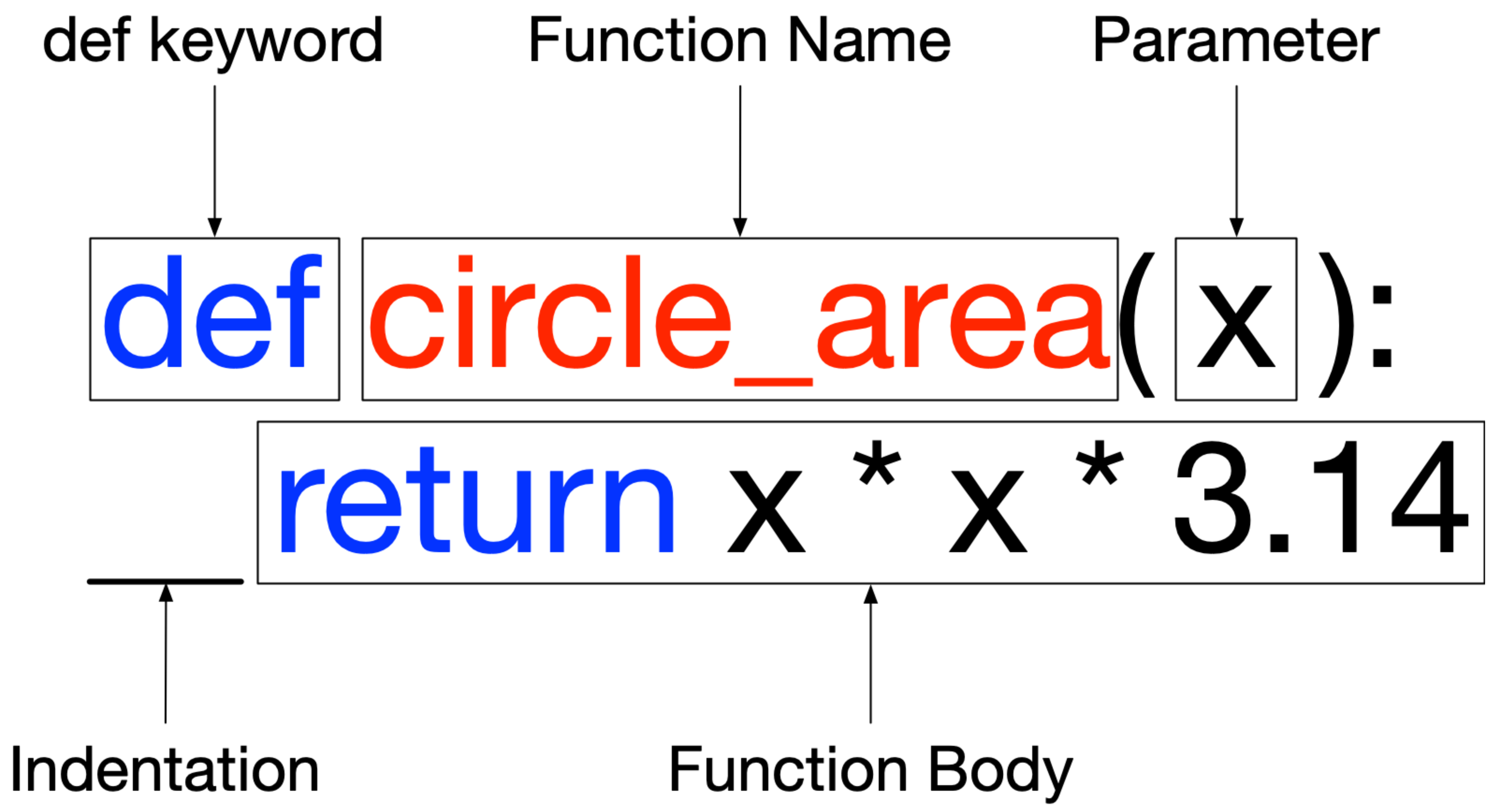
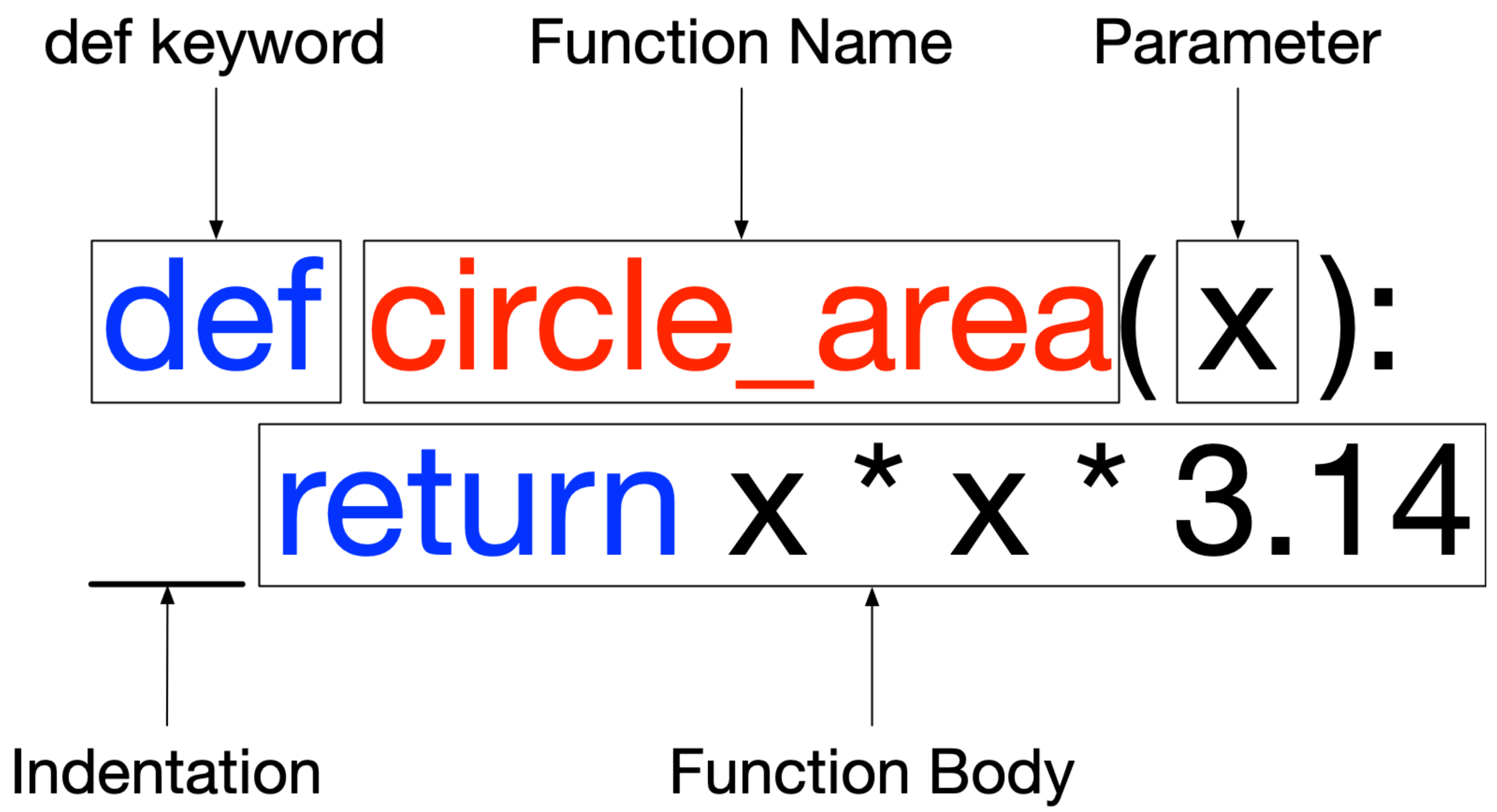
Anatomy of a Function Definition



```
def circle_area(x):
    return x * x * 3.14
```

- A function definition begins with the keyword **def** followed by the **function name** and **parentheses**.
- Any **parameters** should be placed within the *parentheses*.
- Each parameter should be separated by **commas** (**,**).
- The **function body** within every function starts with a **colon** (**:**), **new line**, and **indentation**.

The *return* Statement



```
def circle_area(x):
    return x * x * 3.14
```

- The **return** statement passes (*outputs*) a value from the function to the caller.
- The *return* statement format:
  - return <expression>
- On the left, *x * x * 3.14* is the expression.
- Any statements after *return* are ignored.

Function Definitions vs. Calls

Parameter

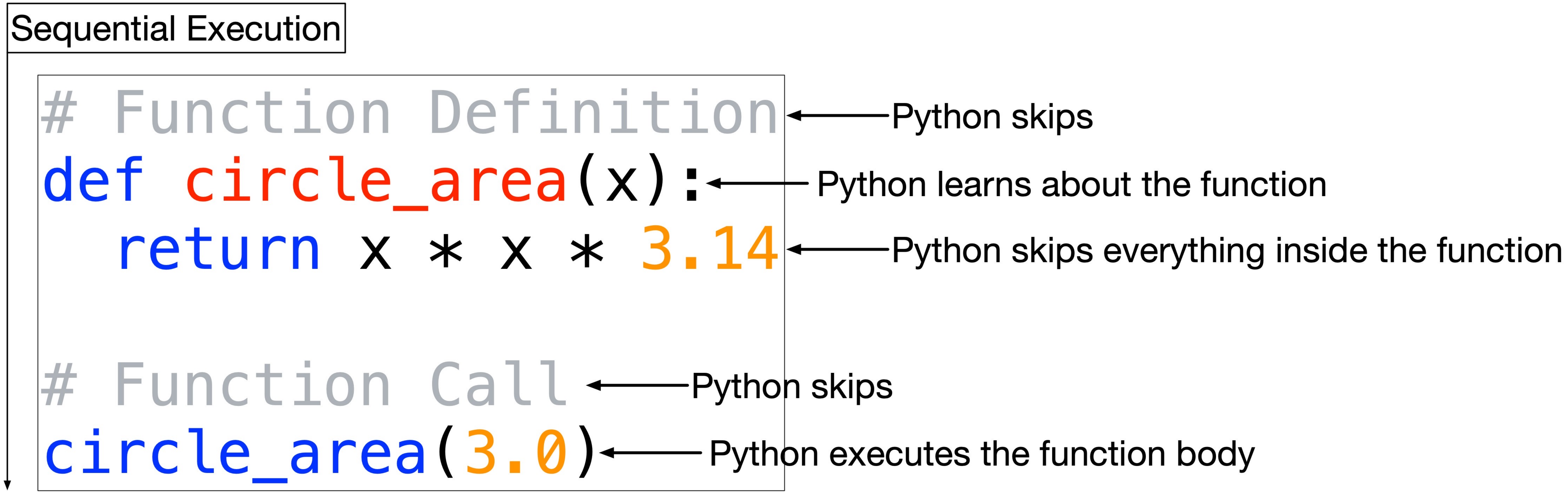
```python
# Function Definition
def circle_area(x):
    return x * x * 3.14

# Function Call
circle_area(3.0)
```

Argument

- Function Definition:
  - Defines what function **does**.
  - Declares parameter **x**.
  - **Parameter**: the *variable* that is listed within the parentheses of a function header.
- Function Call:
  - Command to **do** the function.
  - An **argument** to assign to **x**.
  - **Argument**: a *value* to assign to the function parameter when it is called.

Order of execution

- Function definition must come before the function call.


Sequential Execution
Python skips
Python learns about the function
Python skips everything inside the function
Python skips
Python executes the function body

```
# Function Definition
def circle_area(x):
    return x * x * 3.14

# Function Call
circle_area(3.0)
```

Order of execution: For-Loop

Code

Output

```
print('Welcome!')


for x in range(3):
    print(x)


print('Good Bye!')
```

Order of execution: For-Loop

Code

```
print('Welcome!')


for x in range(3):
    print(x)


print('Good Bye!')
```

Output

```
Welcome!
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
0
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
0
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
0
1
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)


print('Good Bye!')
```

Output

```
Welcome!
0
1
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
0
1
2
```

Order of execution: For-Loop

Code

```
print('Welcome!')

for x in range(3):
    print(x)

print('Good Bye!')
```

Output

```
Welcome!
0
1
2
Good Bye!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')

print('Welcome!')

hello()

print('Good Bye!')
```

Output

```
Welcome!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

```
Welcome!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

```
Welcome!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

```
Welcome!
Hello, World!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

```
Welcome!
Hello, World!
```

Order of execution: Function

Code

```
def hello():
    print('Hello, World!')


print('Welcome!')


hello()


print('Good Bye!')
```

Output

```
Welcome!
Hello, World!
Good Bye!
```