# **PH 366 Day 2:**
# Functions and the Euler Method

8 Jan 2025

# Reviewing Functions – Syntax

```
def function(arg1, arg2):
    implementation
    return output
```

Function definition

```
function(5, 2.3)
```

Function call

# Purpose of Functions

```
def function(arg1, arg2):
    implementation
    return output
```

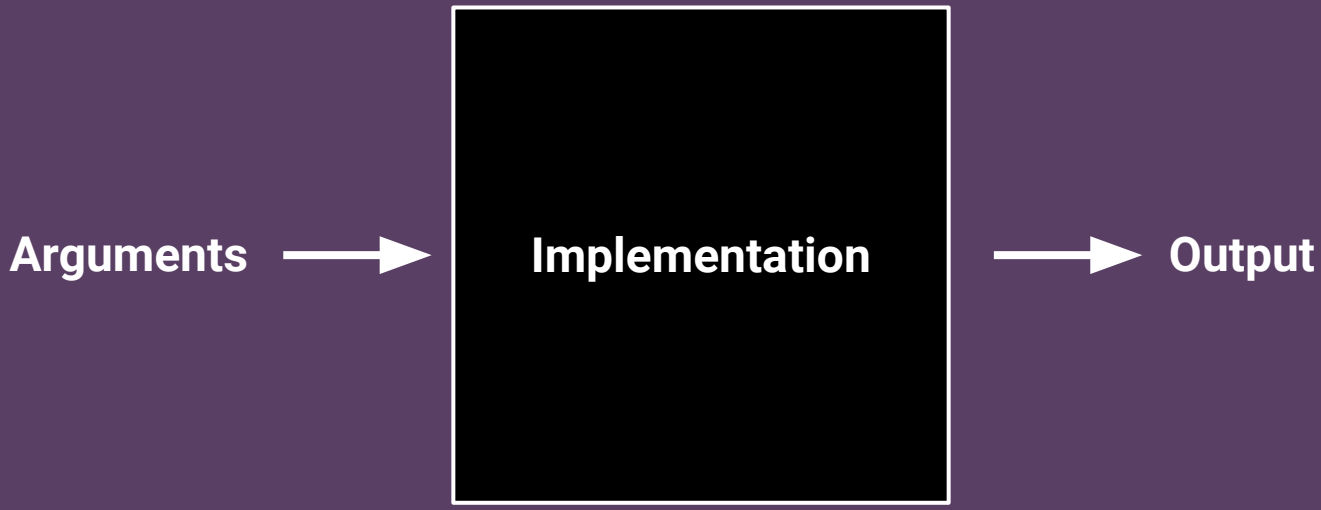
What you write:

```
a = function(5, 2.3)
b = function(6, 5.2)
c = function(9, 3.7)
```

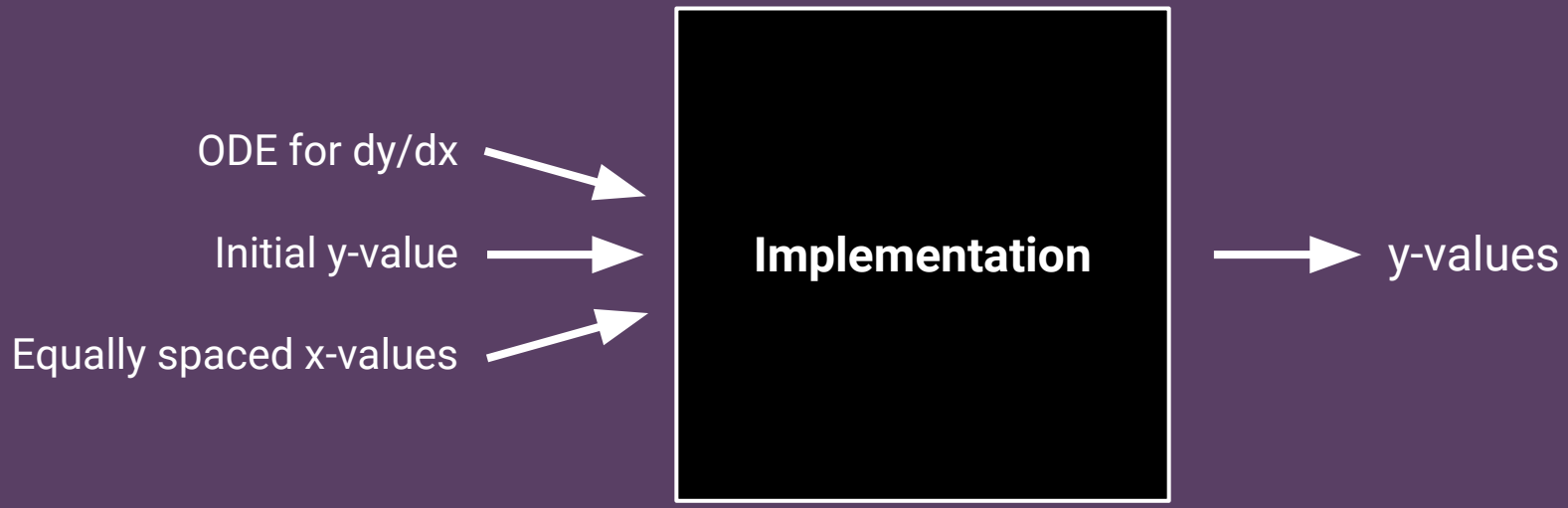
What the computer does:

```
arg1 = 5
arg2 = 2.3
(computer runs all code in
implementation, using arg1 and arg2)
a = output of implementation

arg1 = 6
arg2 = 5.2
(computer runs all code in
implementation, using arg1 and arg2)
b = output of implementation

arg1 = 9
...
```

# How Functions Work



# Today: Writing an Euler method function



# Documenting Functions

Include descriptions of your function's **purpose**, **arguments**, and **output**

```
# weighted_avg computes a weighted average of values
# val1: first value
# val2: second value
# weights: proportions, should be a list of two values summing to 1
# output: the weighted average of vals, using weights
def weighted_avg(val1, val2, weights):
    avg = val1 * weights[0] + val2 * weights[1]
    return avg
```

# Documentation as a Coding Strategy

Documenting functions can also help you figure out how to implement them

```
# weighted_avg computes a weighted average of values
# val1: first value
# val2: second value
# weights: proportions, should be a list of two values summing to 1
# output: the weighted average of vals, using weights
def weighted_avg(val1, val2, weights):
    Implementation TBD
    return avg
```