VARIABLE EQUALITY

We can think of variable equality in two fundamental ways:

	Memory Address	Object State (data)
	`is`	`==`
	identity operator	equality operator
	`var_1 is var_2`	`var_1 == var_2`
Negation	`is not`	`!=`
	`var_1 is not var_2`	`var_1 != var_2`
	`not(var_1 is var_2)`	`not(var_1 == var_2)`

Examples

```
a = 10
b = a
```

`a is b` ✓

`a == b` ✓

```
a = 'hello'
b = 'hello'
```

`a is b` ✓ but as we'll see later, don't count on it!

`a == b` ✓

```
a = [1, 2, 3]
b = [1, 2, 3]
```

`a is b` ✗

`a == b` ✓

```
a = 10
b = 10.0
```

`a is b` ✗

`a == b` ✓

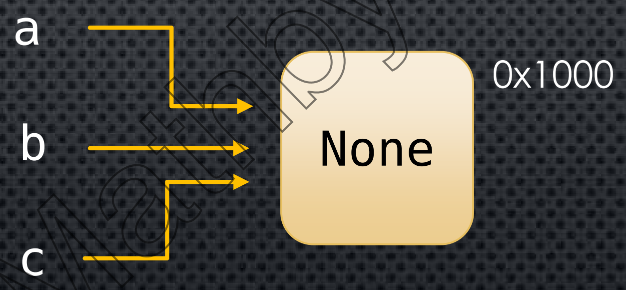
The None object

The None object can be assigned to variables to indicate that they are not set (in the way we would expect them to be), *i.e. an "empty" value (or null pointer)*

But the None object is a real object that is managed by the Python memory manager

Furthermore, the memory manager will always use a shared reference when assigning a variable to None

```
a = None
b = None
c = None
```



So we can test if a variable is "not set" or "empty" by comparing it's memory address to the memory address of None using the `is` operator

```
a is None
```
✔

```
x = 10
```

```
x is None
```
✘

```
x is not None
```
✔