Variables, Memory and Garbage Collection in Python

Andres Rojas (SWE)

Holberton

Version 1.0

Let’s recap

Every programming language uses variables to represent values that help developers keep track of information, create mathematical operations or display information to users.

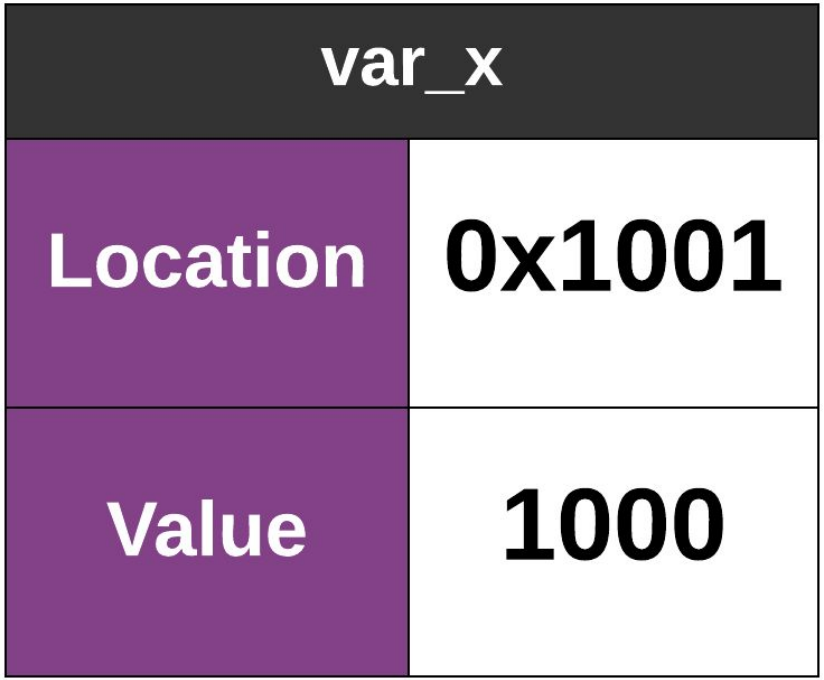
It does not matter if it is a High-level programming language like python, or a low-level programming language like C, all of them use variables which in the end, are addresses to a memory space that stores a value.

But wait, it is different in both languages!

In C, a variable owns a space in memory, let's imagine

int var_x=1000

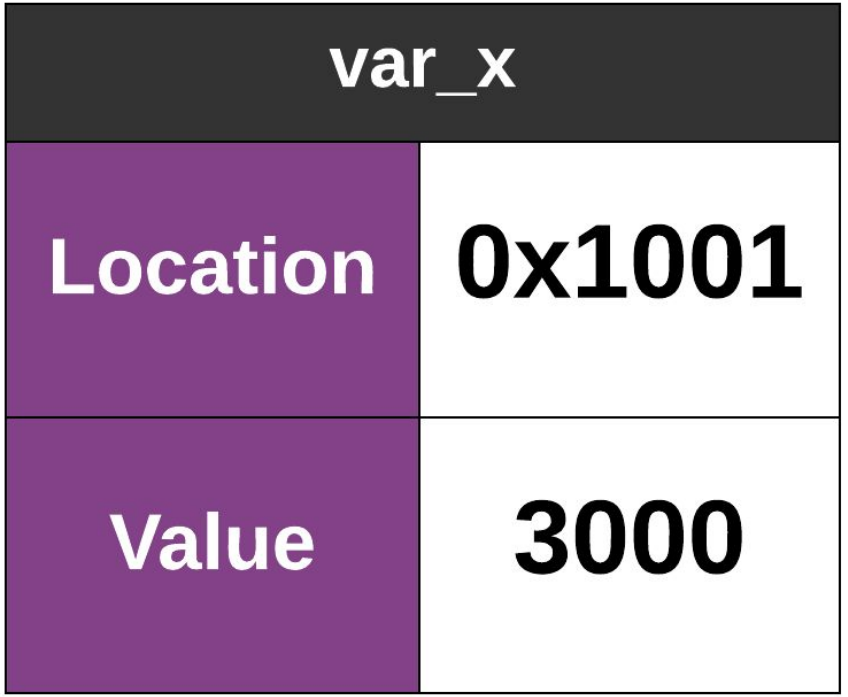
Is stored in memory like this:



When we change the value of a variable

Let's suppose we update the value **var_x = 3000**

C simply updates the value inside the variable



When we change the value of a variable

Now let's imagine **int var_y = var_x**

var_x	
Location	0x1001
Value	2000

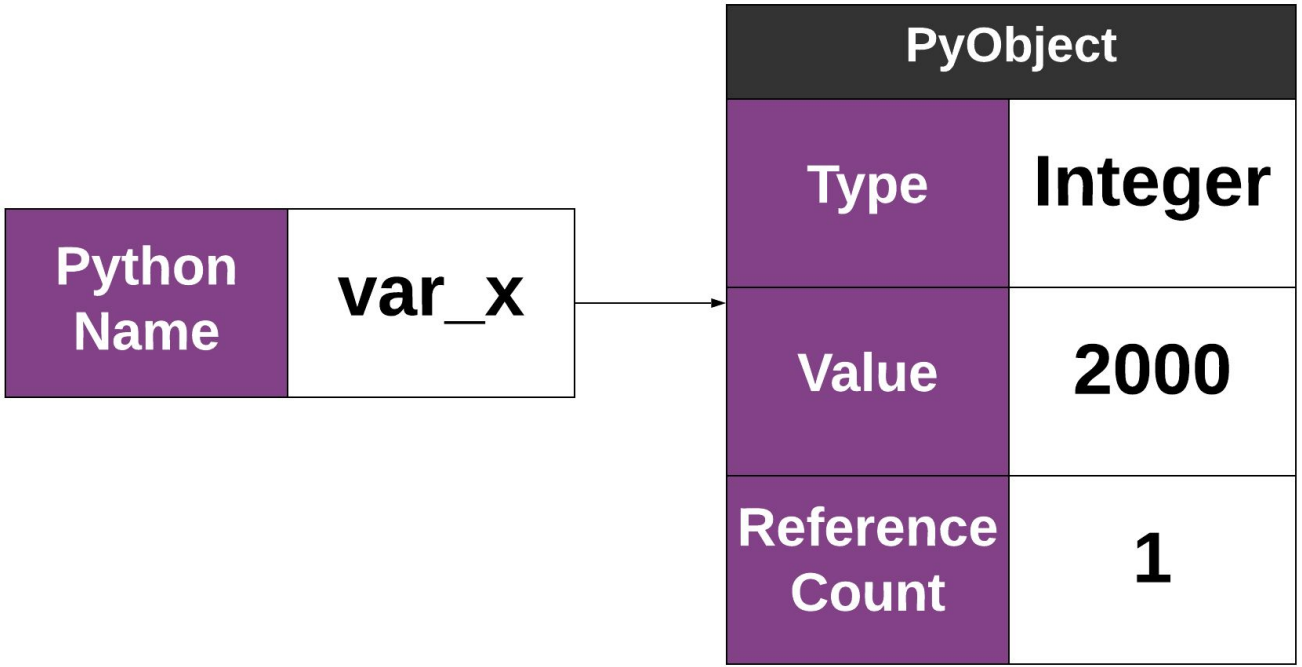
var_y	
Location	0x2001
Value	2000


SEEMS LEGIT TO ME
memegenerator.net

In Python, it is a little bit different

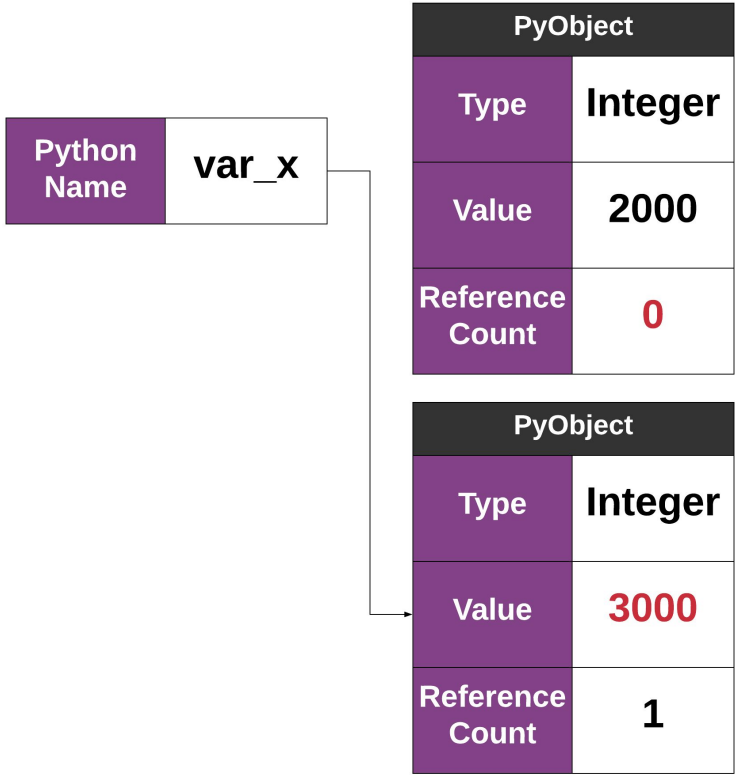
Creation of a variable is similar, but think of names, not variables.

Let's create a new variable **var_x = 2000**



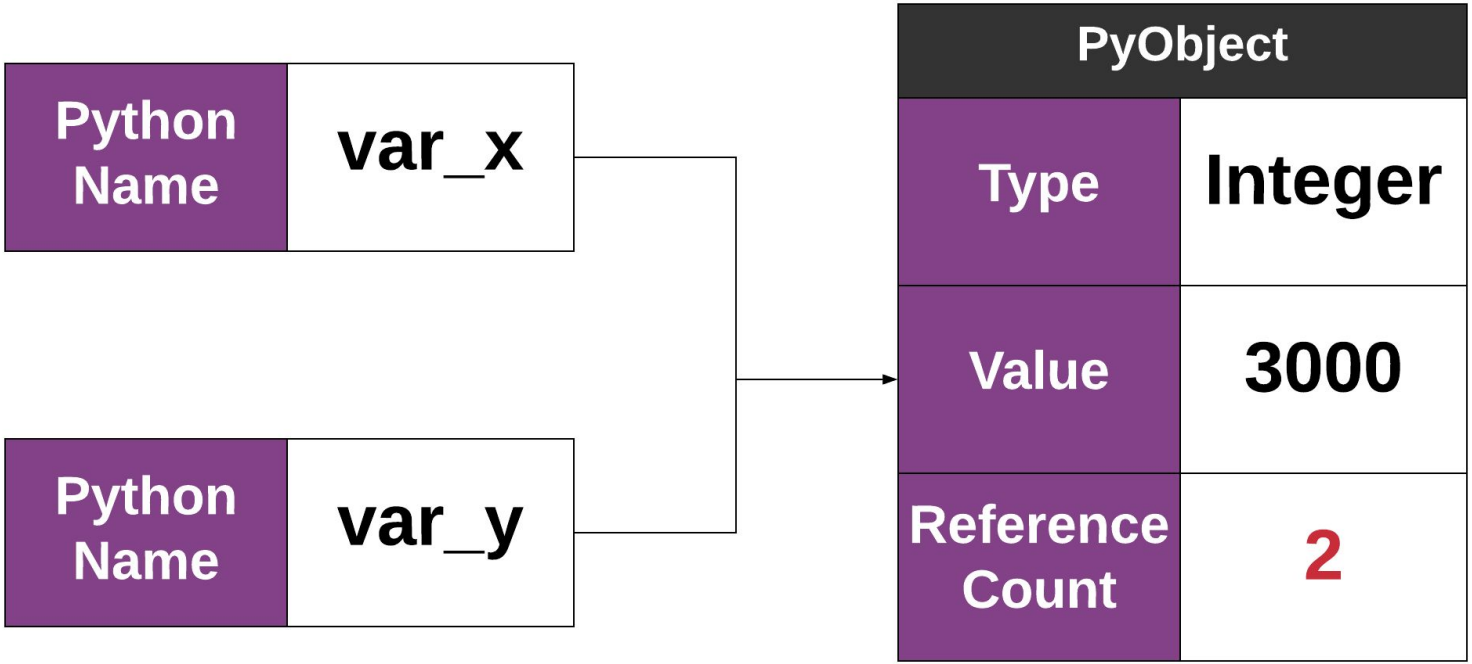
When we update the value

It makes a simple update: **var_x = 3000**



When you do var_y = var_x...

It makes a simple update:



To get the variable address

Use the function **id(*[variable name]*)**, it will return the variable address in base 10, and can be easily converted into hexadecimal using **hex()**

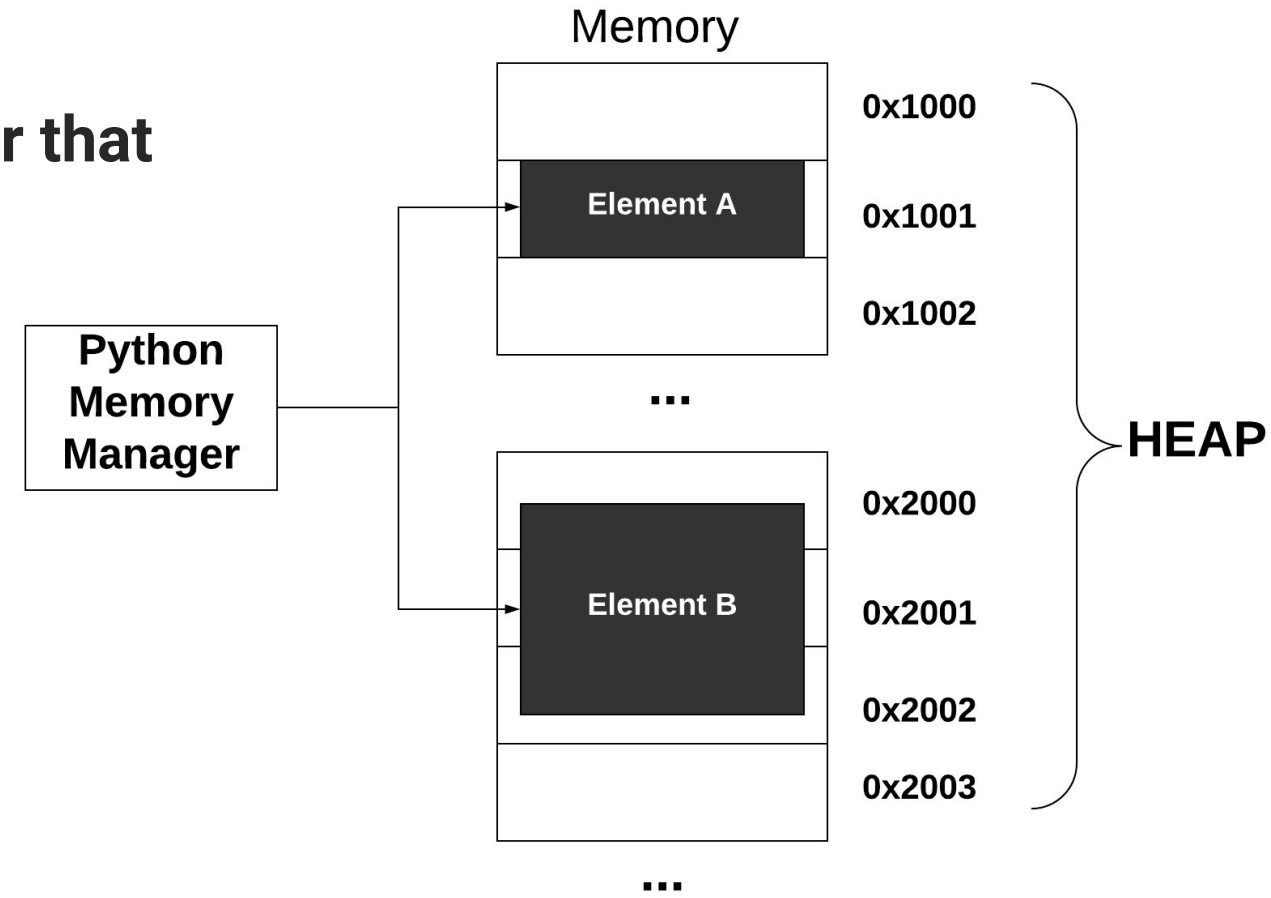
Counting cards… I mean, references

In the previous example, we say that the reference Count, was increased every time we assign a new name to the same PyObject, this is used as a control method to keep the Memory Clean

Fun with Python's sys.getrefcount()

Who is in charge or that process?

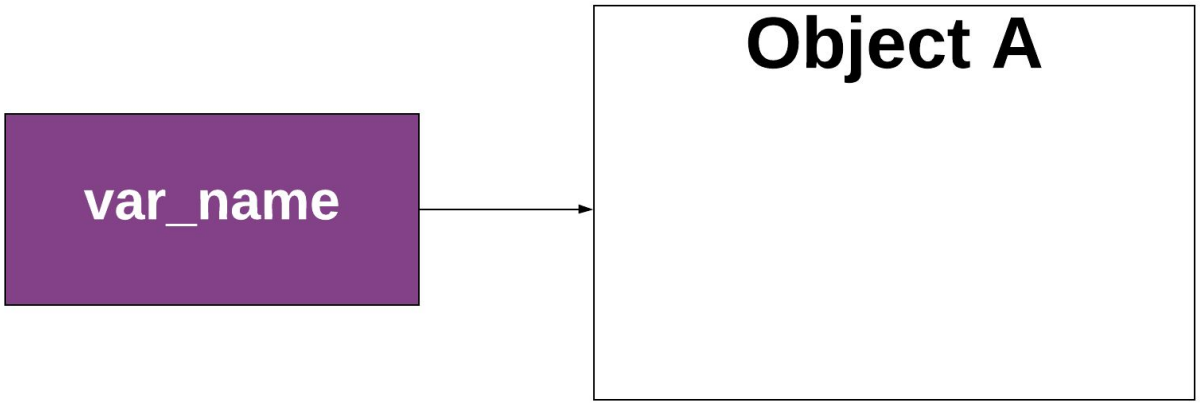
It has the Basic Heap layout, but it is managed by the [Python Memory Manager]



Garbage Collection

Circular References

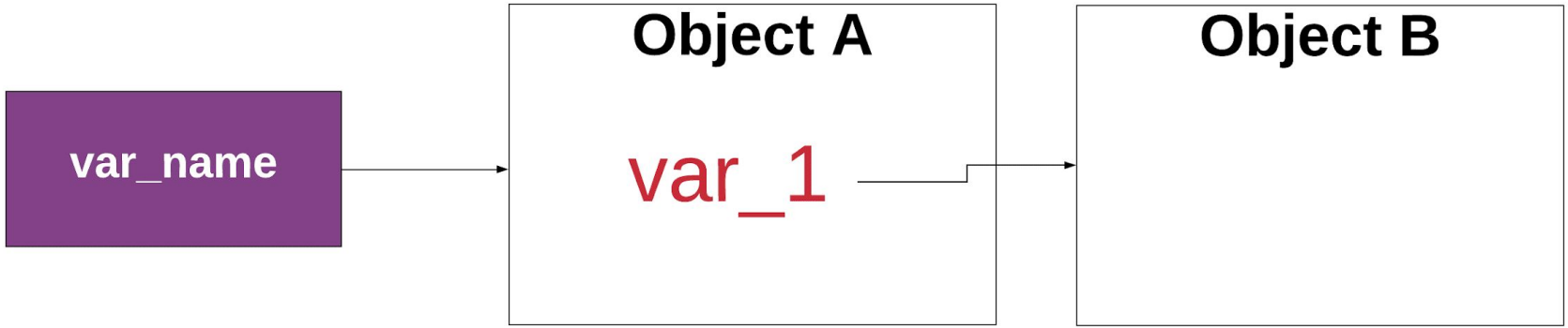
Let’s imagine a variable that points to an object



The reference count of **Object A** is **1**

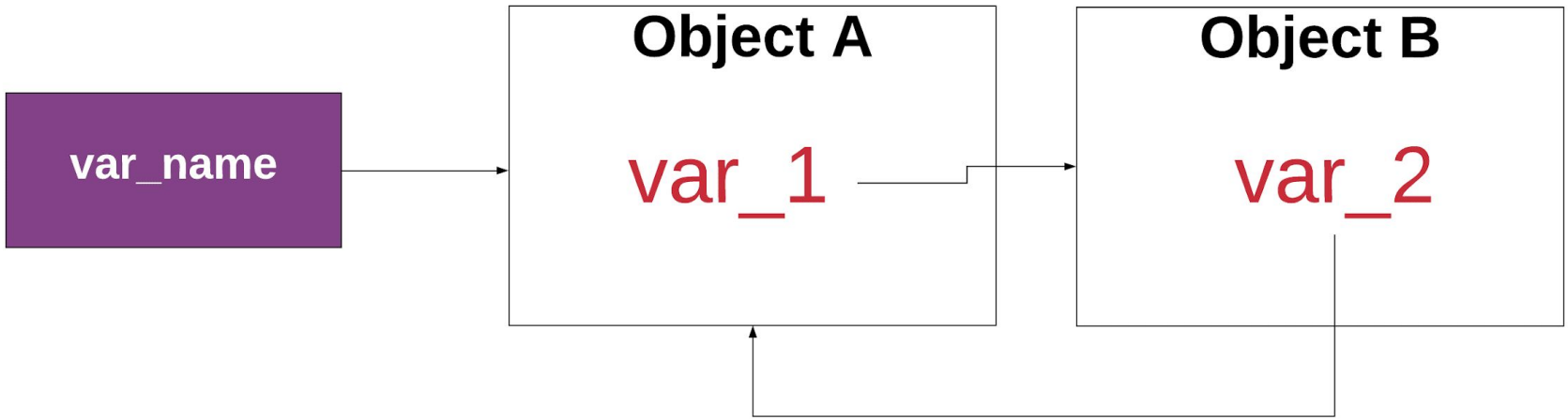
Circular References

Now let's imagine that **Object A**, has a variable that points to **Object B**

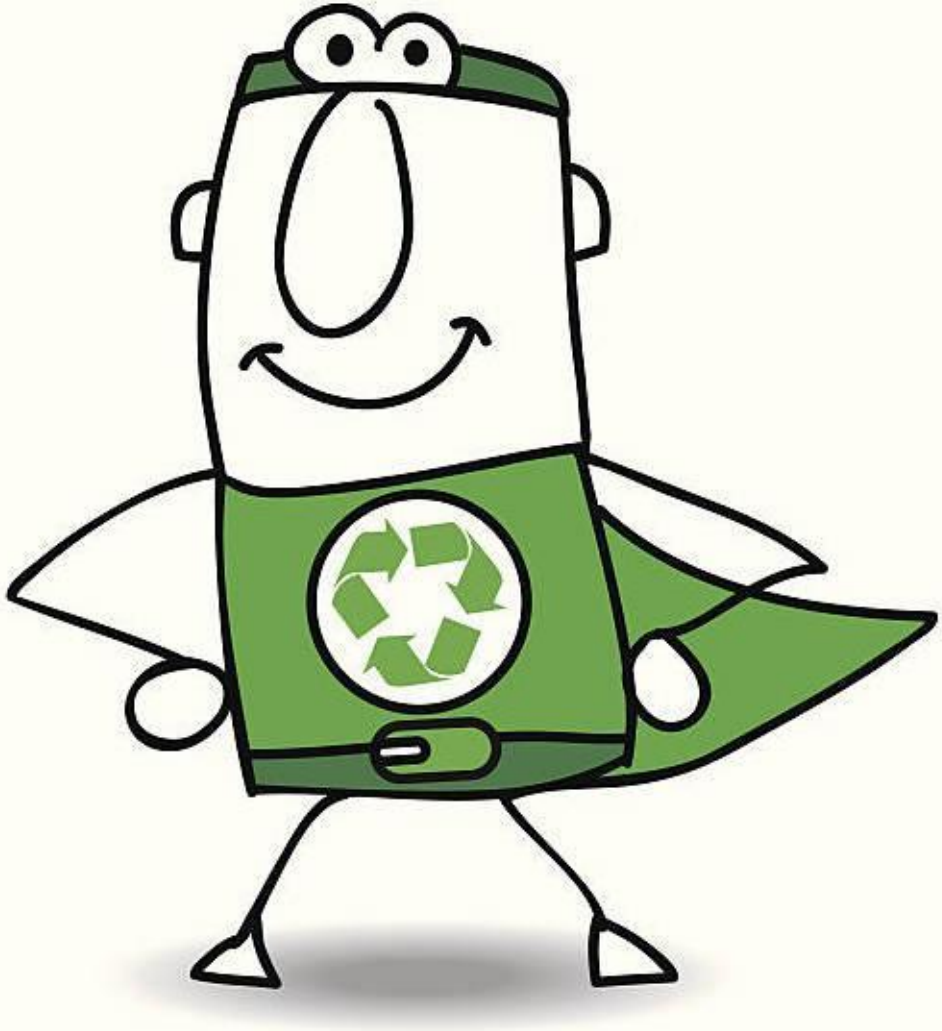


Circular References

To make it more complex, let mahe **Object B** point to **Object A**



- Can be controlled using the gc module
- It is turned on by default
- You can turn it off (under your own risk)
- Runs periodically
- You can use it Manually
- It might fail for Python < 3.4



Andrés Rojas (SWE)

@andreserojasi

Holberton

Date