Introducción a Python.

Curso de Python.

Mariano Fernández Abadía

Curso 2022/23

Rama de Estudiantes IEEE UPCT

Introducción.

Python.

Python (/pai·thn/) es un lenguaje de programación cuyas características lo han convertido en la última década en unos de los lenguajes de programación más populares en múltiples aplicaciones (ciencia, ingeniería, análisis de datos, servidores, etc.).

Python.

Python (/pai·thn/) es un lenguaje de programación cuyas características lo han convertido en la última década en unos de los lenguajes de programación más populares en múltiples aplicaciones (ciencia, ingeniería, análisis de datos, servidores, etc.).
Su diseño está centrado en la legibilidad del código. Se puede definir como un lenguaje:

Python.

Python (/pai·thn/) es un lenguaje de programación cuyas características lo han convertido en la última década en unos de los lenguajes de programación más populares en múltiples aplicaciones (ciencia, ingeniería, análisis de datos, servidores, etc.).
Su diseño está centrado en la legibilidad del código. Se puede definir como un lenguaje:

- Multiplataforma. Como decía el viejo lema de Java: “Escríbelo una vez, ejecútalo en cualquier lugar”.

Python.

Python (/pai·thn/) es un lenguaje de programación cuyas características lo han convertido en la última década en unos de los lenguajes de programación más populares en múltiples aplicaciones (ciencia, ingeniería, análisis de datos, servidores, etc.).
Su diseño está centrado en la legibilidad del código. Se puede definir como un lenguaje:

- Multiplataforma. Como decía el viejo lema de Java: “Escríbelo una vez, ejecútalo en cualquier lugar”.
- Multiparadigma. Permite usar cómodamente varias formas de entender la programación: POO, programación estructurada, imperativa, funcional, etc.

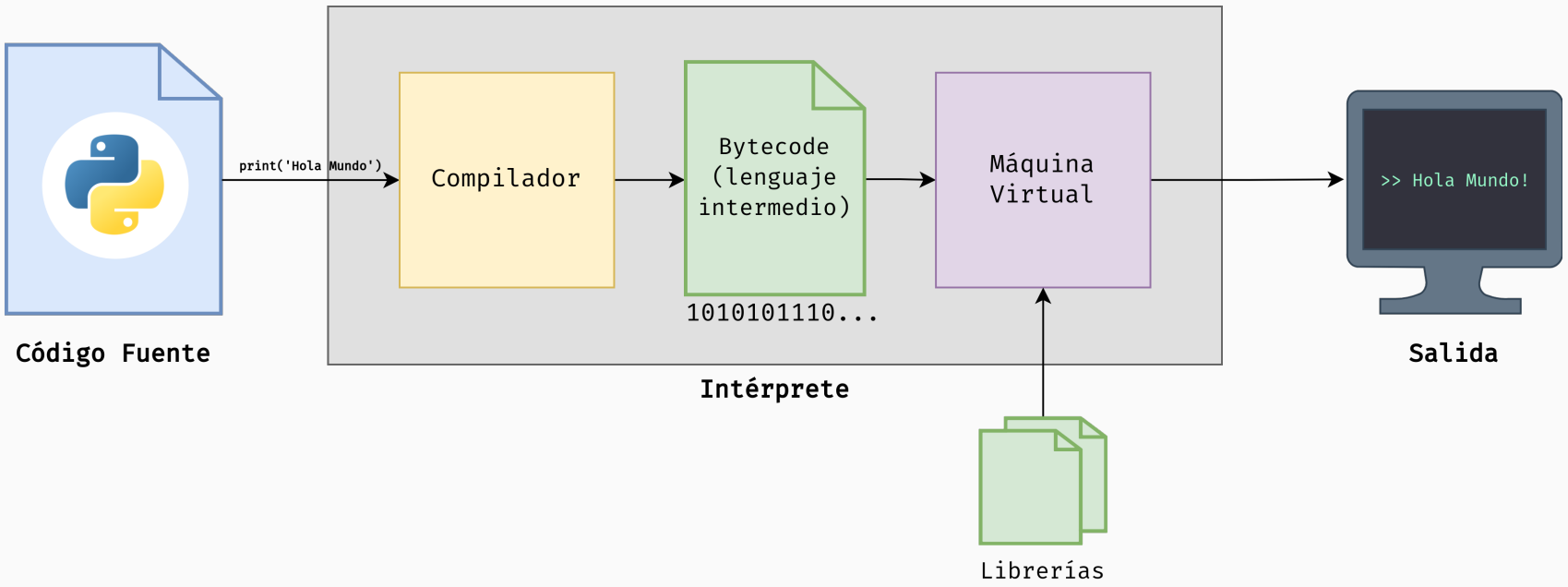
Python.

Python (/pai·thn/) es un lenguaje de programación cuyas características lo han convertido en la última década en unos de los lenguajes de programación más populares en múltiples aplicaciones (ciencia, ingeniería, análisis de datos, servidores, etc.).
Su diseño está centrado en la legibilidad del código. Se puede definir como un lenguaje:

- Multiplataforma. Como decía el viejo lema de Java: “Escríbelo una vez, ejecútalo en cualquier lugar”.
- Multiparadigma. Permite usar cómodamente varias formas de entender la programación: POO, programación estructurada, imperativa, funcional, etc.
- Iterpretado. El código se ejecuta línea a línea a través de un intérprete.

Intérprete.

Intérprete.


print('Hola Mundo')
Compilador
Bytecode
(lenguaje
intermedio)
1010101110...
Máquina
Virtual
>> Hola Mundo!
Código Fuente
Intérprete
Salida
Librerías

Máquina Virtual.

Ejemplo de compilación código máquina (x86-64 gcc).

Código en C.

```c
int suma(int a, int b)
{
    return a+b;
}
```

Código en ensamblador.

```
suma(int, int):
    push    rbp
    mov     rbp, rsp
    mov     DWORD PTR [rbp-4], edi
    mov     DWORD PTR [rbp-8], esi
    mov     edx, DWORD PTR [rbp-4]
    mov     eax, DWORD PTR [rbp-8]
    add     eax, edx
    pop     rbp
    ret
```

Máquina Virtual.

De forma análoga, la máquina virtual de Python ejecuta su propio lenguaje “máquina”: el *bytecode*.

Código en Python.

```
def suma(a, b):
    return a+b;
```

Código en bytecode.

```
2   0 LOAD_FAST     0 (a)
    2 LOAD_FAST     1 (b)
    4 BINARY_ADD
    6 RETURN_VALUE
```

Cada instrucción en la máquina virtual se corresponden con una serie de operaciones bien optimizadas en la máquina real.

Representación de Datos en Python.

Objetos.

Todo en Python son objetos. Las clases, funciones e incluso tipos de datos simples como enteros; números en coma flotante y cadenas de texto, son objetos en Python.

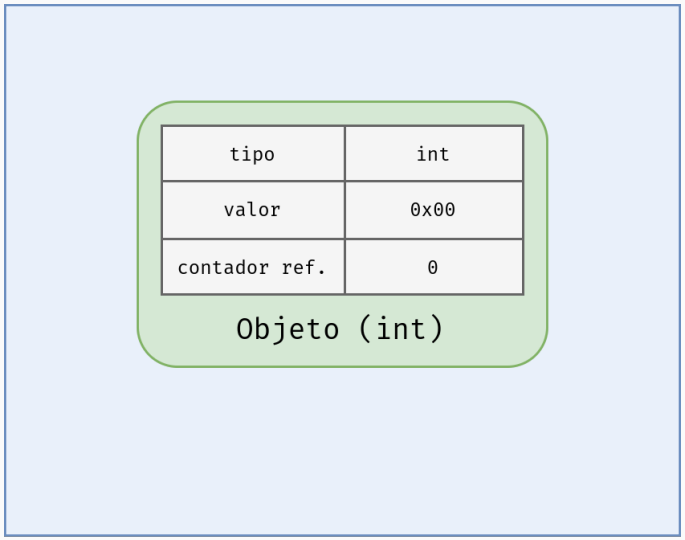
Se llama **objeto** a una estrutura de datos con unos atributos (como variables) y métodos (funciones) definidos. Llamamos **clase** a la "plantilla" que define los atributos y métodos de un objeto. Esto se tratará más adelante con más profundidad.

Aviso.

Todo lo que veremos a continuación es cierto en la implementación estándar de Python de escrita en C (CPython) y en la mayoría de implementaciones. Sin embargo, no tiene porqué ser así en todas.

Representación de los objetos.

Todos los elementos que aparecen en Python derivan de la clase ***object*** y contienen los atributos “tipo” (la clase de dato que es), “valor” donde se guarda la información en bruto del objeto y un “contador de referencia” que cuenta el número de variables que hacen referencia al objeto. Por ejemplo un objeto de tipo *int* (número entero) se puede representar del siguiente modo en la memoria:

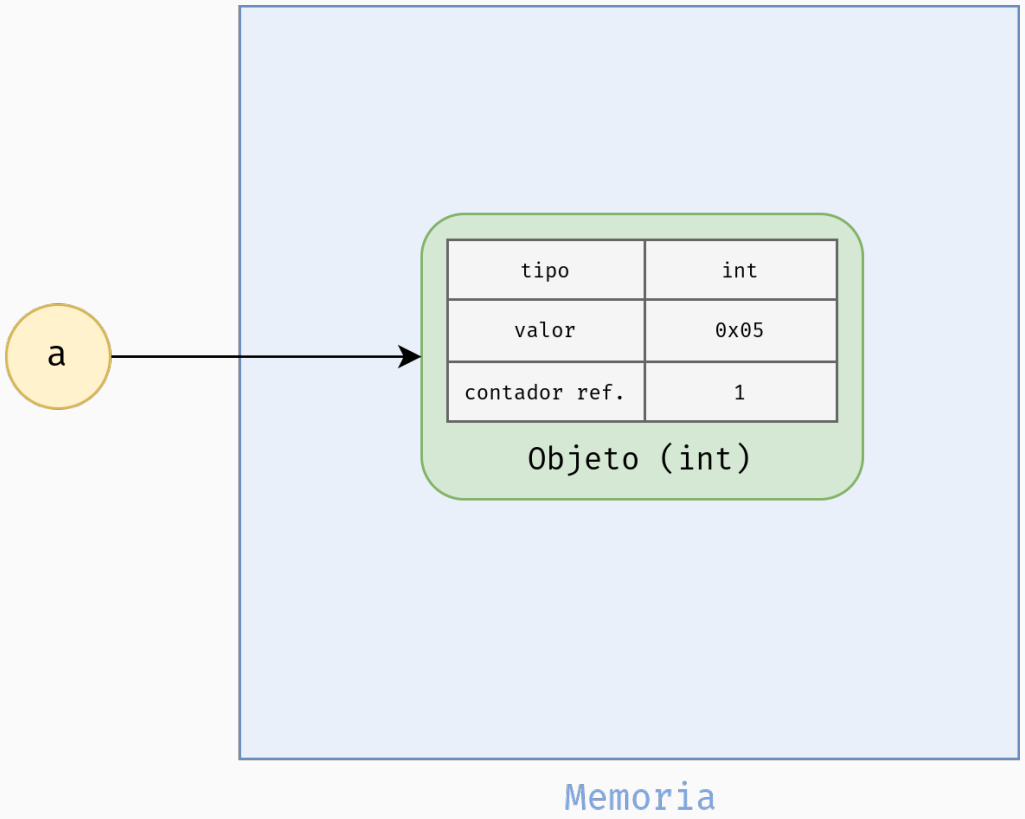


Asignación.

Cuando asignamos cualquier cosa a una variable lo que hacemos es crear una referencia.

Ejemplo

```
a = 5
```

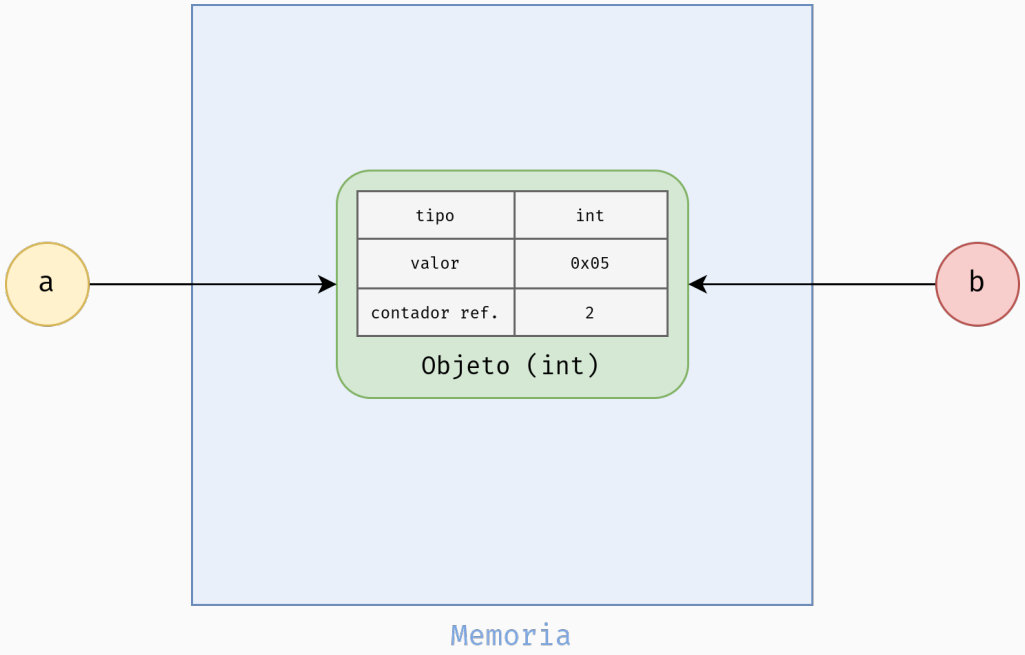


Asignación.

Si asignamos una variable a otra, ambas apuntan al mismo sitio.

Ejemplo

```
b = a
```

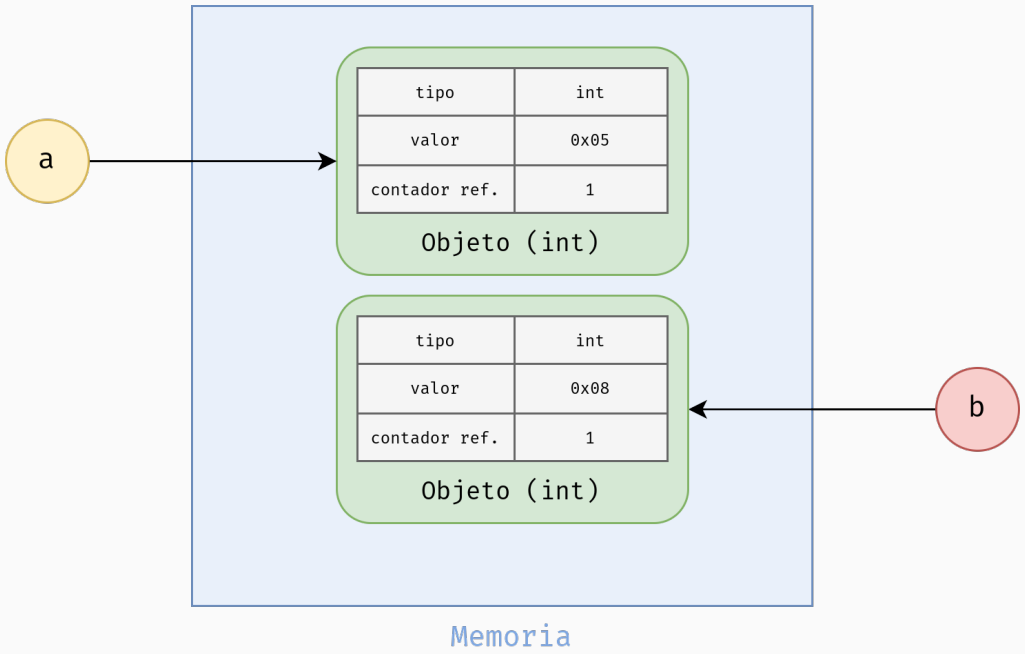


Asignación.

Cuando operamos una variable no modificamos el valor sino que hacemos referencia a uno nuevo.

Ejemplo

```
b = b + 5 # es como asignar a 'b' el valor 8
```

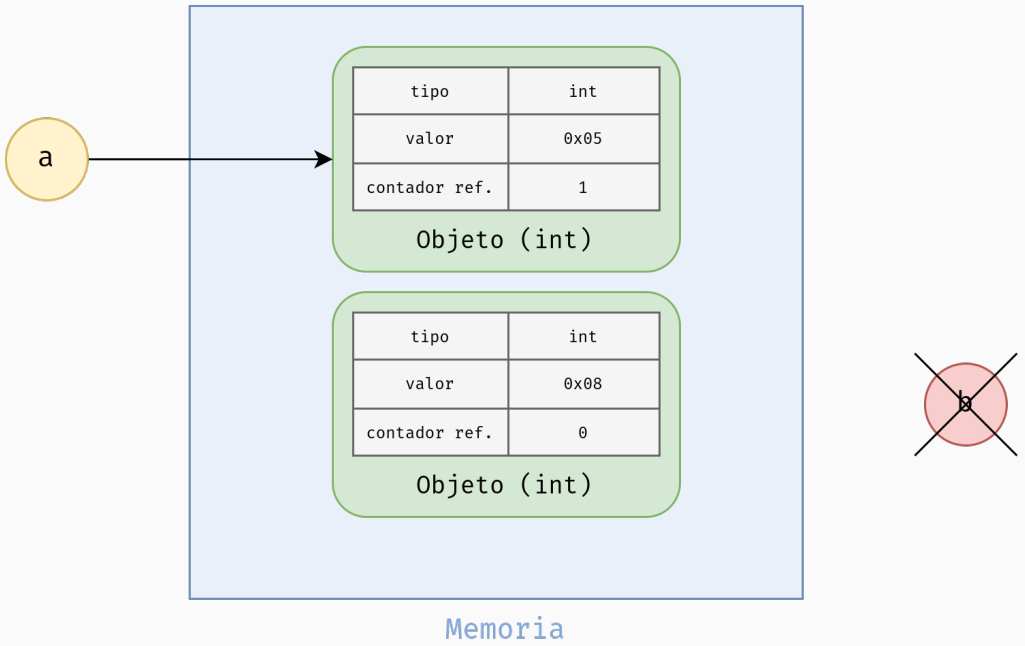


Eliminar variables.

Si una variable se elimina, solo dejamos de referenciar un objeto.

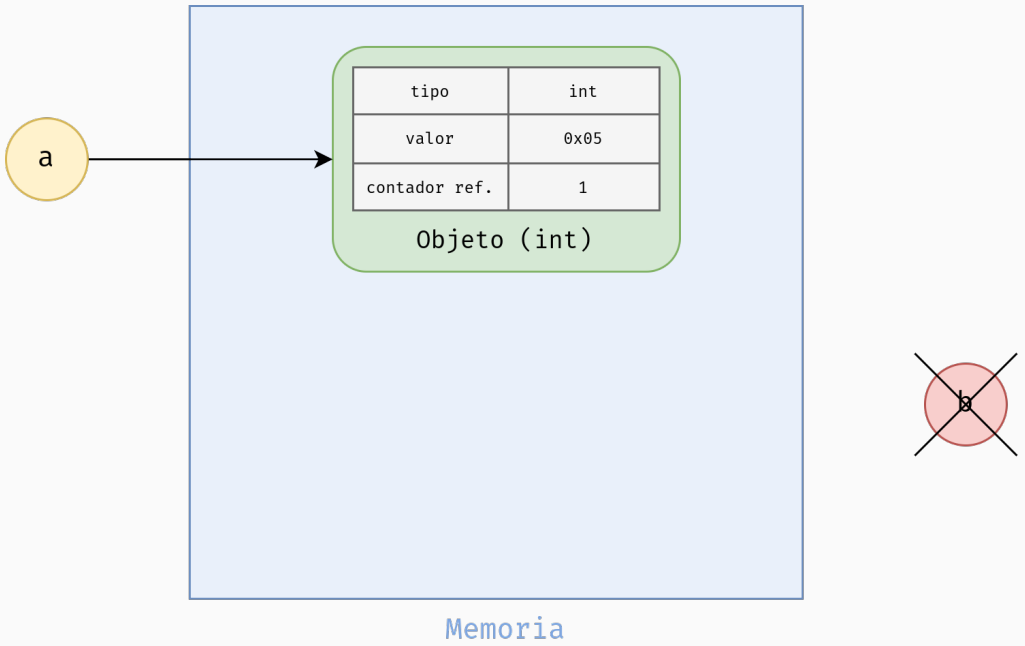
Ejemplo

```
del b
```



Colector de basura.

Cuando se deja de apuntar a un objeto, este se elimina automáticamente mediante el colector de basura (*garbage collector*).

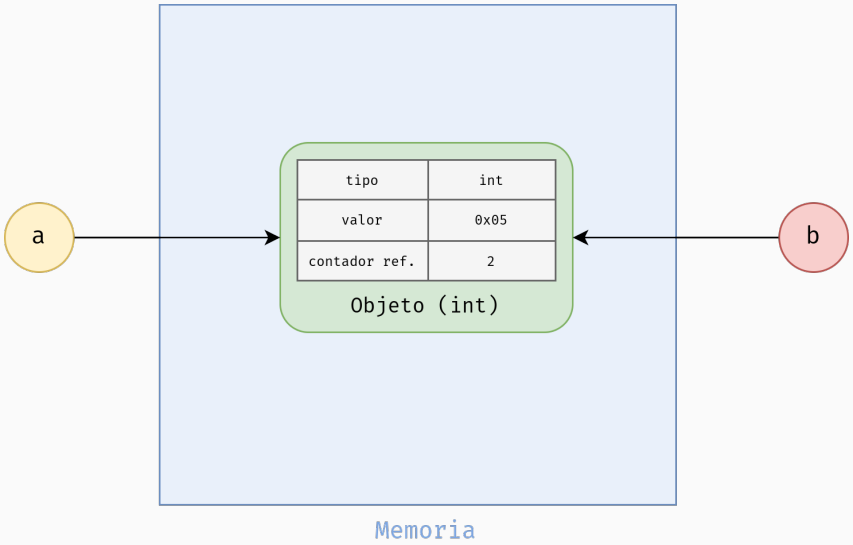


Referencias únicas.

Cuando asignamos un objeto ya existente a una variable, estamos haciendo referencia al objeto previamente definido. Esto significa, en nuestro ejemplo, que todas las variables de tipo entero de valor 5, hacen referencia al mismo objeto.

Ejemplo

```
a = 5
b = 5
```



¿Dudas?

Atribución

Esta presentación está hecha en LaTeX. La tema de la plantilla está disponible en:

`github.com/matze/mtheme`

Gracias a *Matthias Vogelgesang* por compartirlo.