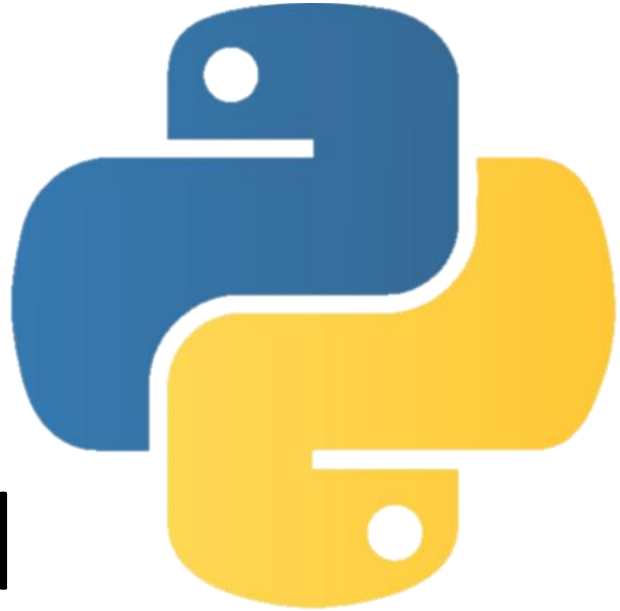
Curso Python 2024 Teoría-1: Memoria y mutabilidad



Hoy:

- ✓ Repaso clase anterior: git y venv
- ✓ Variables y referencias a la memoria
- ✓ Mutabilidad y casos
- ✓ Igualdad de variables

Entorno de trabajo: Herramientas

Entorno de trabajo: Git y Github

Comandos básicos*:

- git status
- git add <file_name>
- git commit –m “<message>”
- git push

Entorno de trabajo: Git y Github

Comandos básicos remoto*:

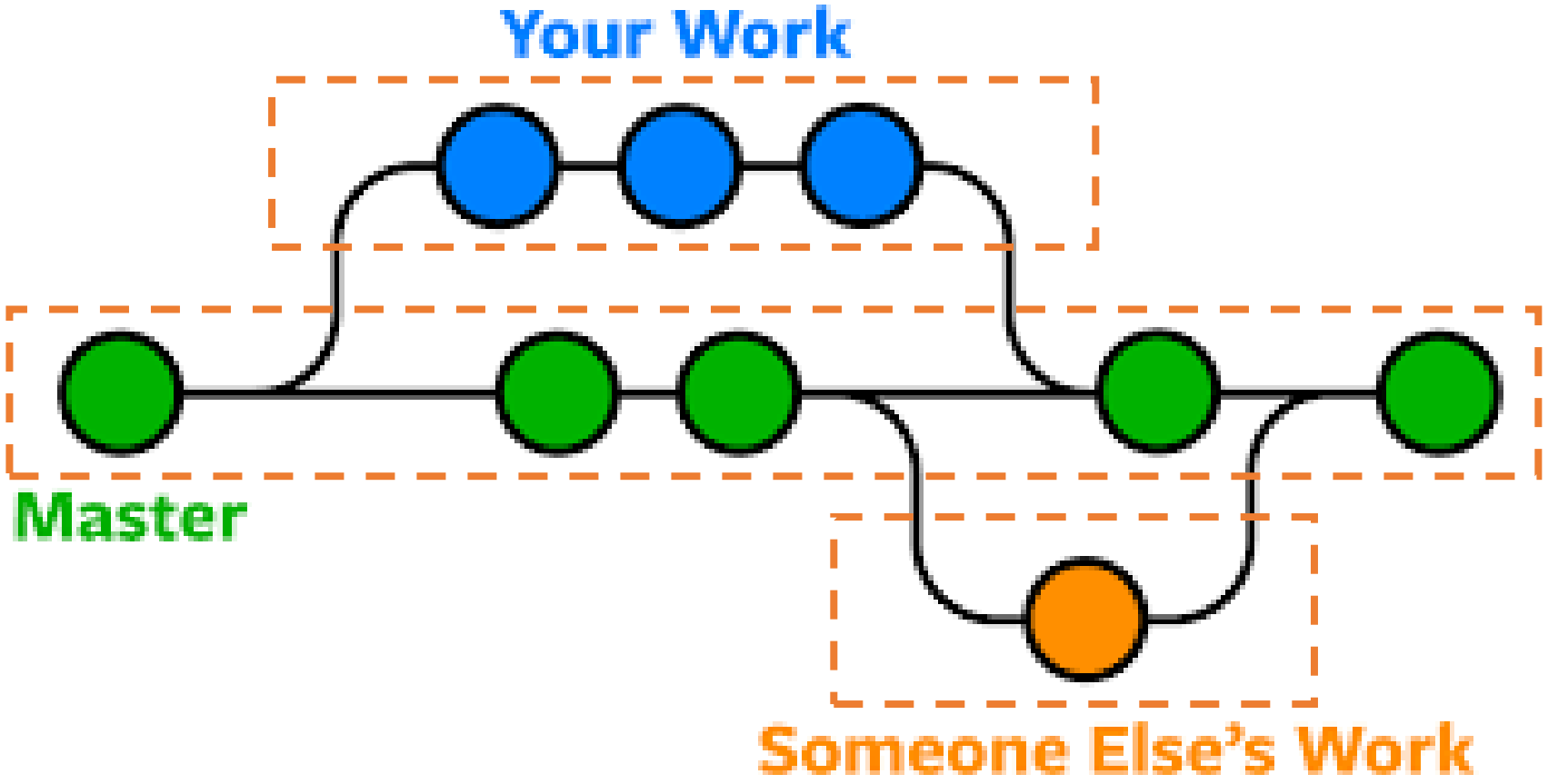
- git pull
- git branch <branch_name>
- git checkout <branch_name>

Entorno de trabajo: venv

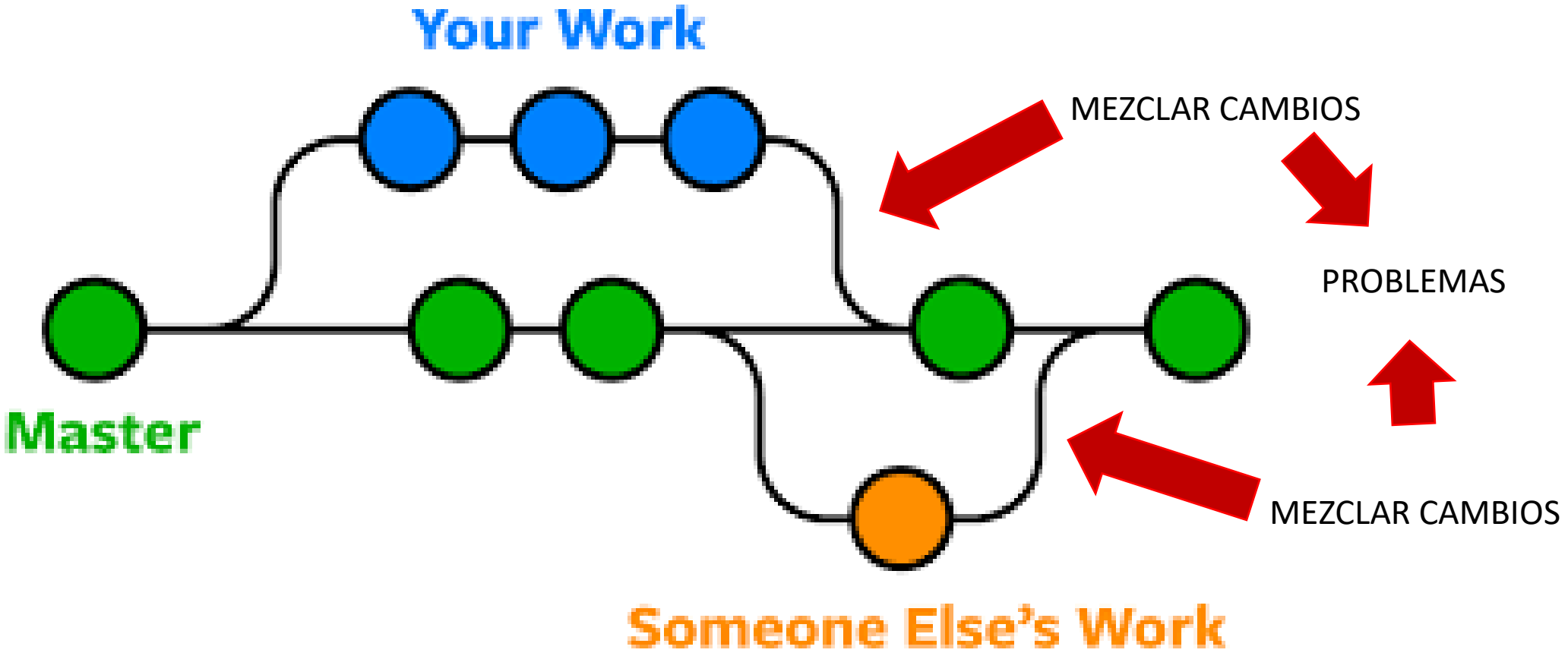
Comandos*:

- python –m venv .venv
- .venv\Scripts\activate (Windows)
- source .venv/bin/activate (Linux y Mac)

Entorno de trabajo: Git



Entorno de trabajo: Git



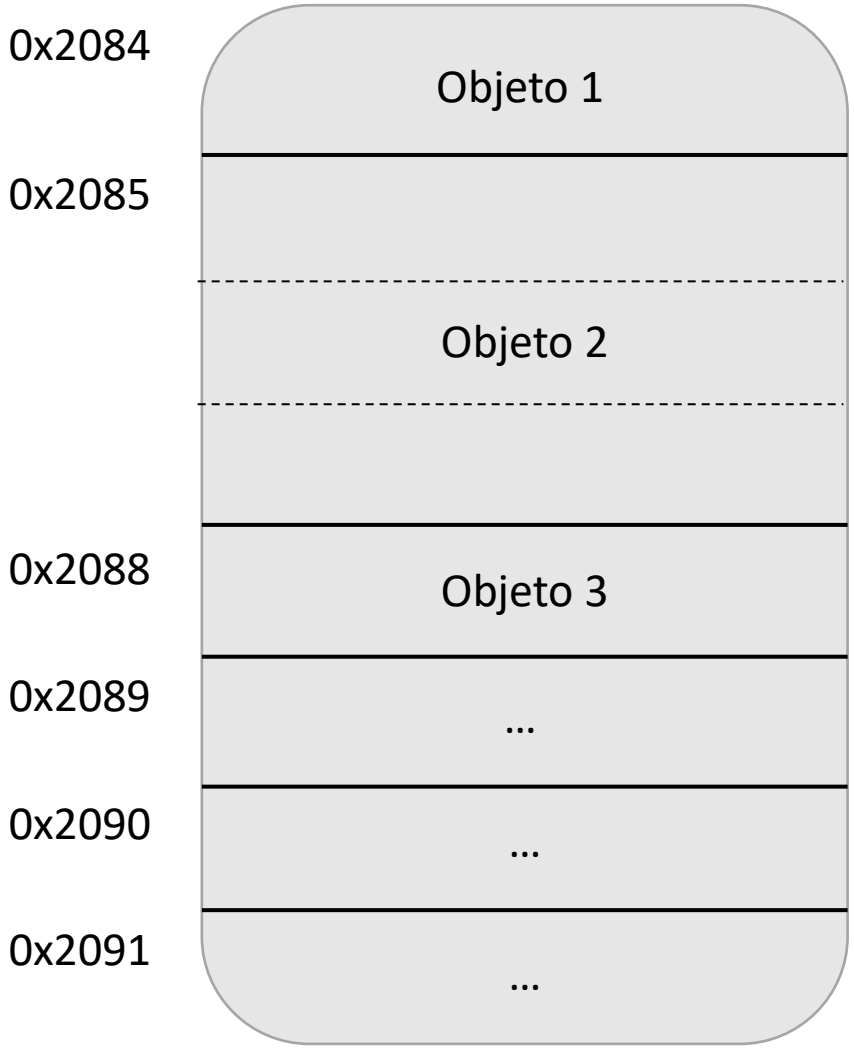
Entorno de trabajo: Git y Github

Definir usuario local:

- git config --global user.name "Your Name"
- git config --global user.email "you@example.com"

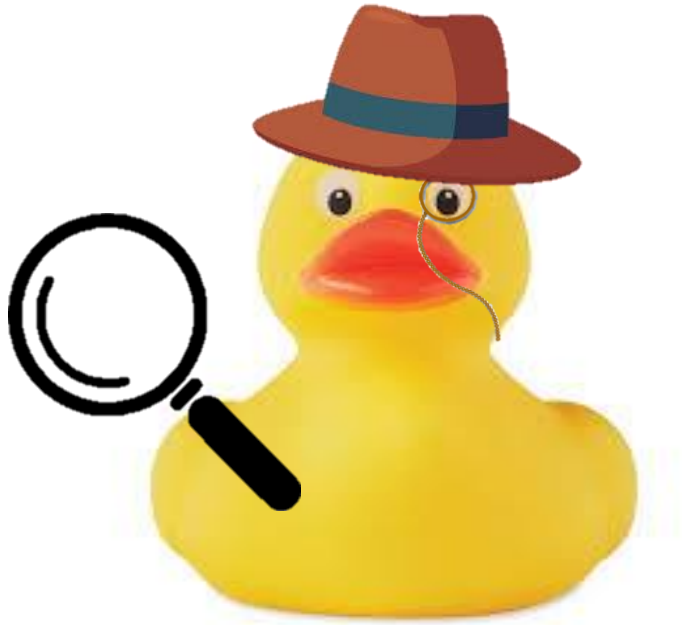
Memoria

- La memoria es un conjunto de cajones donde podemos pedir y guardar datos
- Habrá datos que ocupen más de un hueco
- Con saber donde empieza cada objeto nos basta



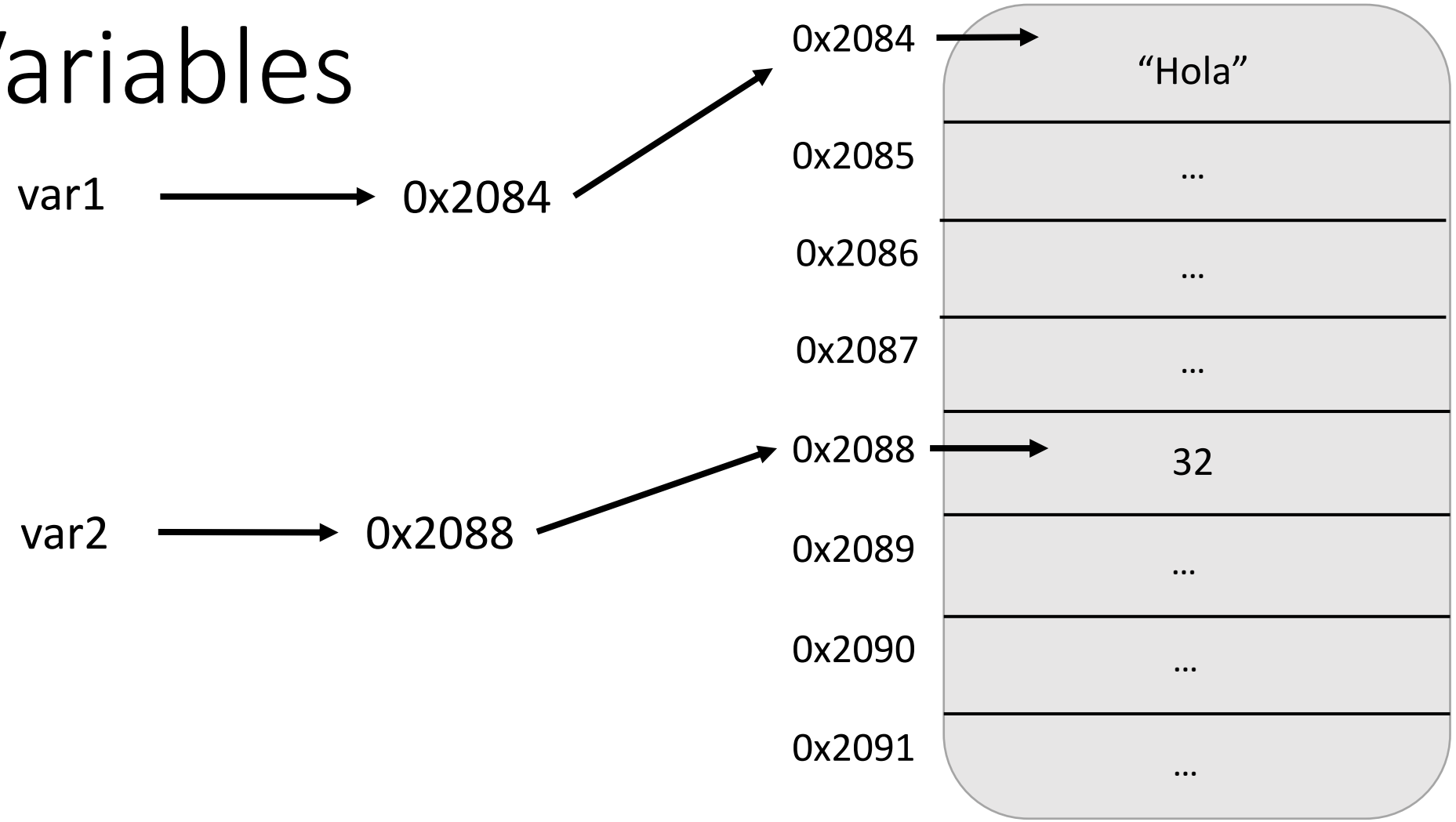
Variables y referencias

Las variables almacenan los objetos que tenemos en memoria…



… en la práctica es cierto pero hay más detrás de estas

Variables

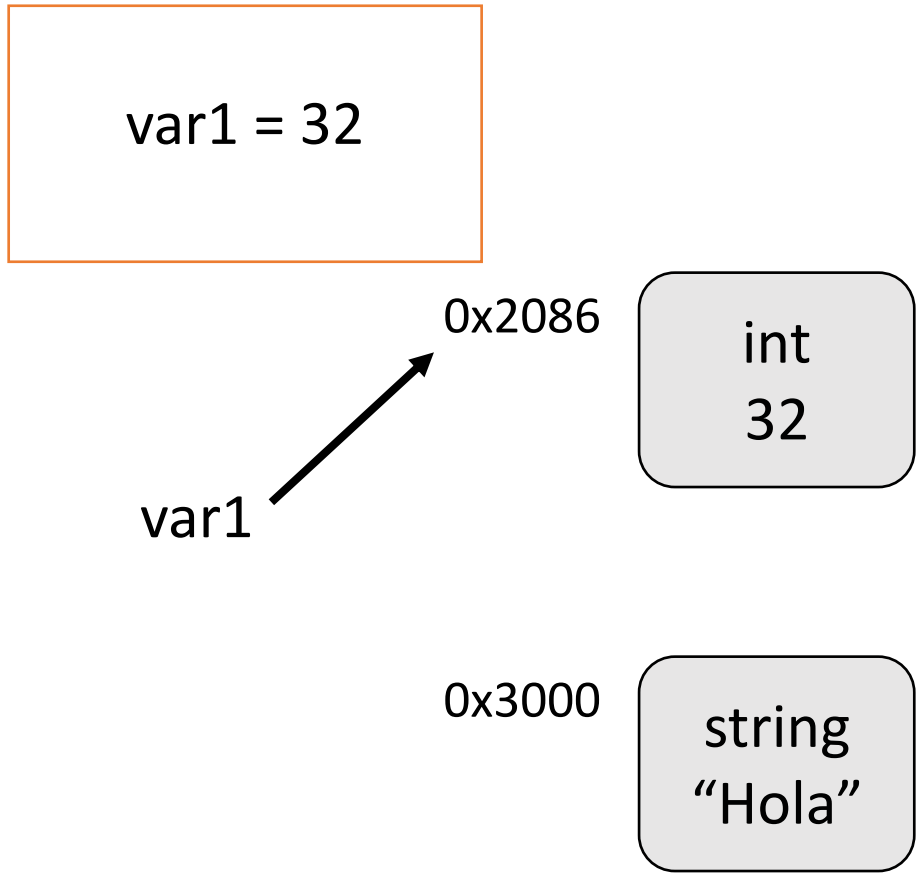

var1
0x2084
var2
0x2088
0x2084
0x2085
0x2086
0x2087
0x2088
0x2089
0x2090
0x2091
“Hola”
…
…
…
32
…
…
…

Variables

- Son “nombres” de las direcciones de memoria
- Una variable no es un objeto realmente
- Una variable es una referencia a una dirección que contiene un objeto

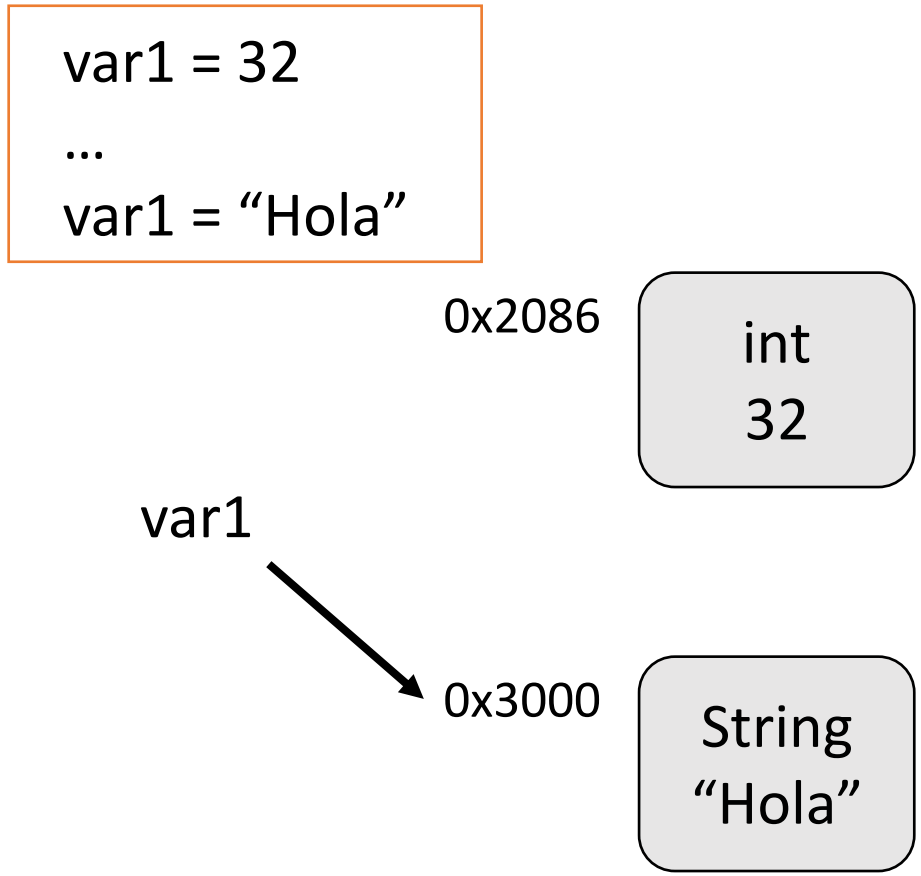
Variables: Python

- Las variables no tienen un tipo asignado
- Python es un lenguaje dinámico (no estático)
- Si cambiamos el tipo de variable referenciado cambiamos su referencia a otro sitio en memoria

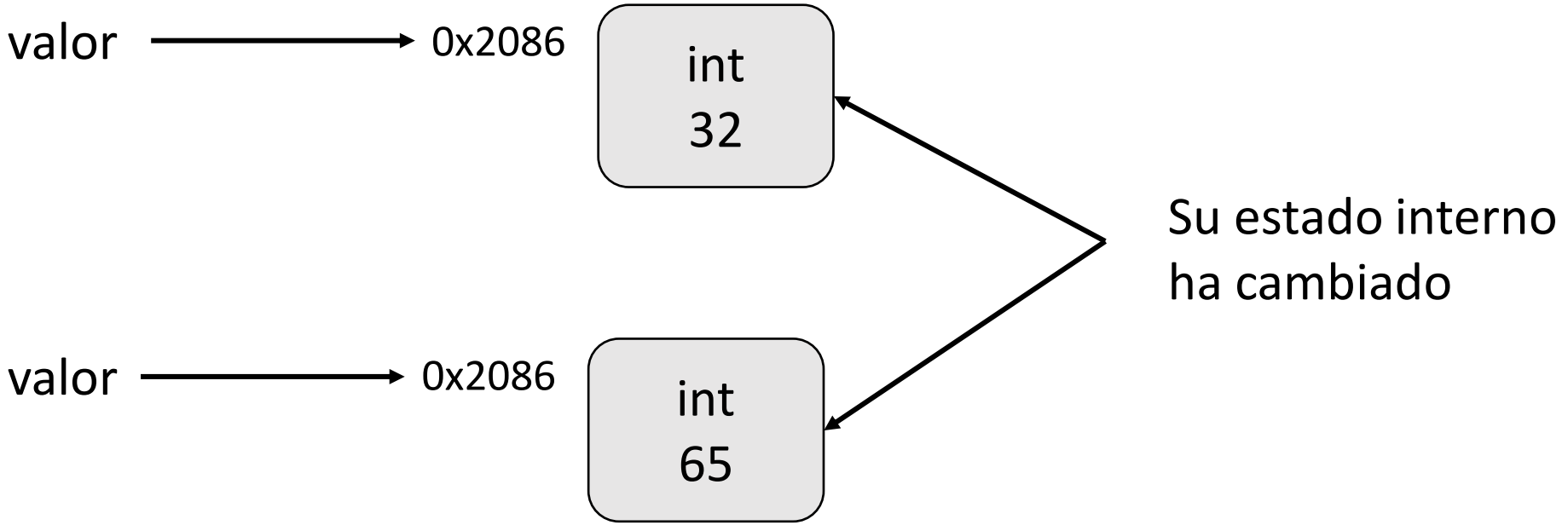


Variables: Python

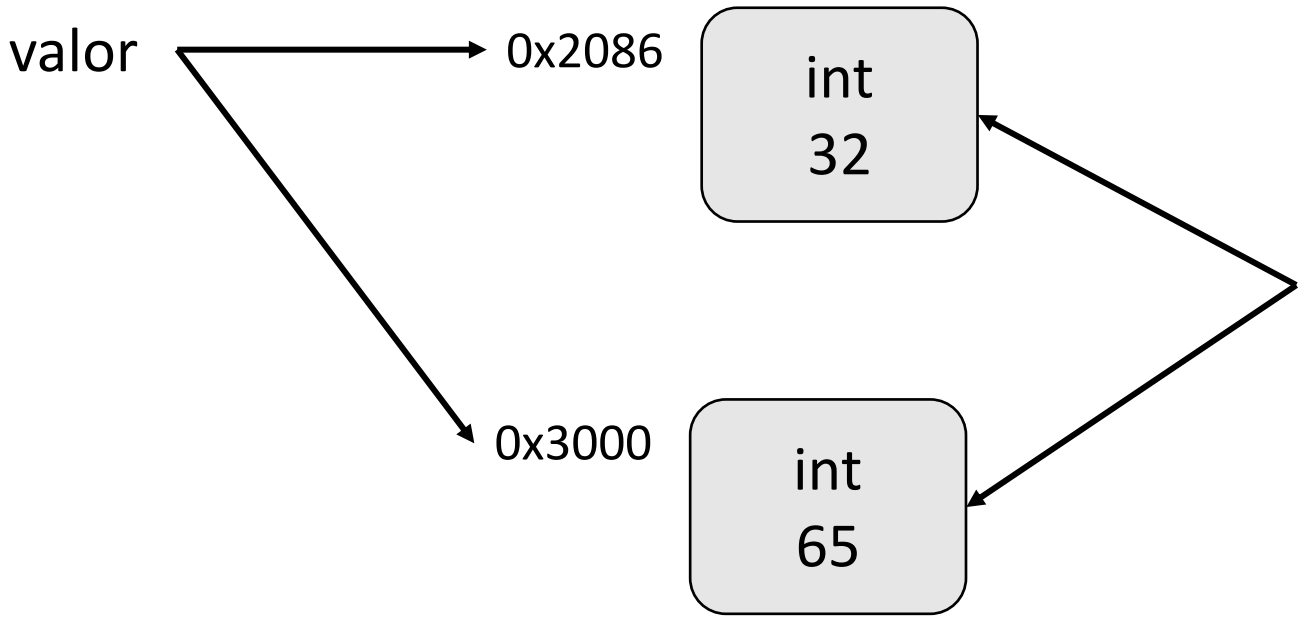
- Las variables no tienen un tipo asignado
- Python es un lenguaje dinámico (no estático)
- Si cambiamos el tipo de variable referenciado cambiamos su referencia a otro sitio en memoria



Mutabilidad: Objetos mutables



Mutabilidad: Objetos inmutables



Su estado interno
NO ha cambiado

↓

Ha cambiado
su referencia
de la memoria

Mutabilidad: Ejemplos Python

MUTABLES

- Tipos numéricos
- Strings
- Tuplas
- Clases definidas por usuarios

INMUTABLES

- Listas
- Sets
- Diccionarios
- Clases definidas por usuarios

Mutabilidad:Tuplas vs Listas

- En listas podemos remplazar, añadir y eliminar, en tuplas NO
- Es más seguro trabajar con tipos inmutables

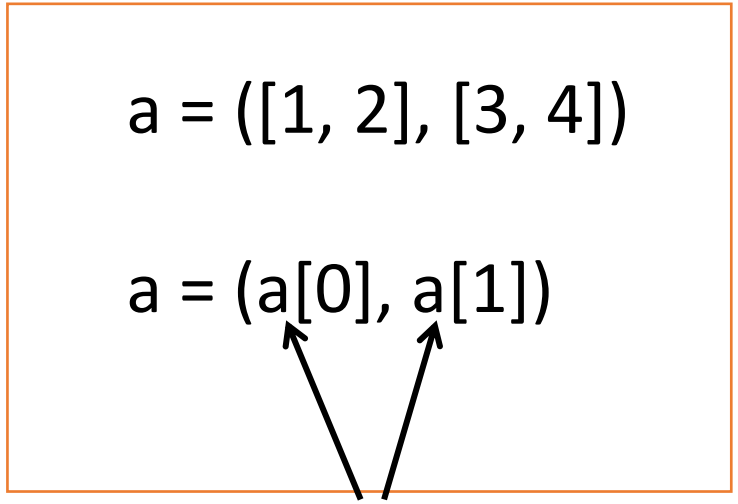
```
lista1 = [1, 2]
tupla1 = (3, 4)

lista1.append(3)
lista1[0] = 100

# Con tupla dará error
# tupla1[0] = 100
```

Mutabilidad: ¡Cuidado!

- Es posible que un objeto inmutable contenga elementos mutables
- Ejemplo: Tupla que contiene listas
- Podemos modificar objetos mutables que se encuentren dentro

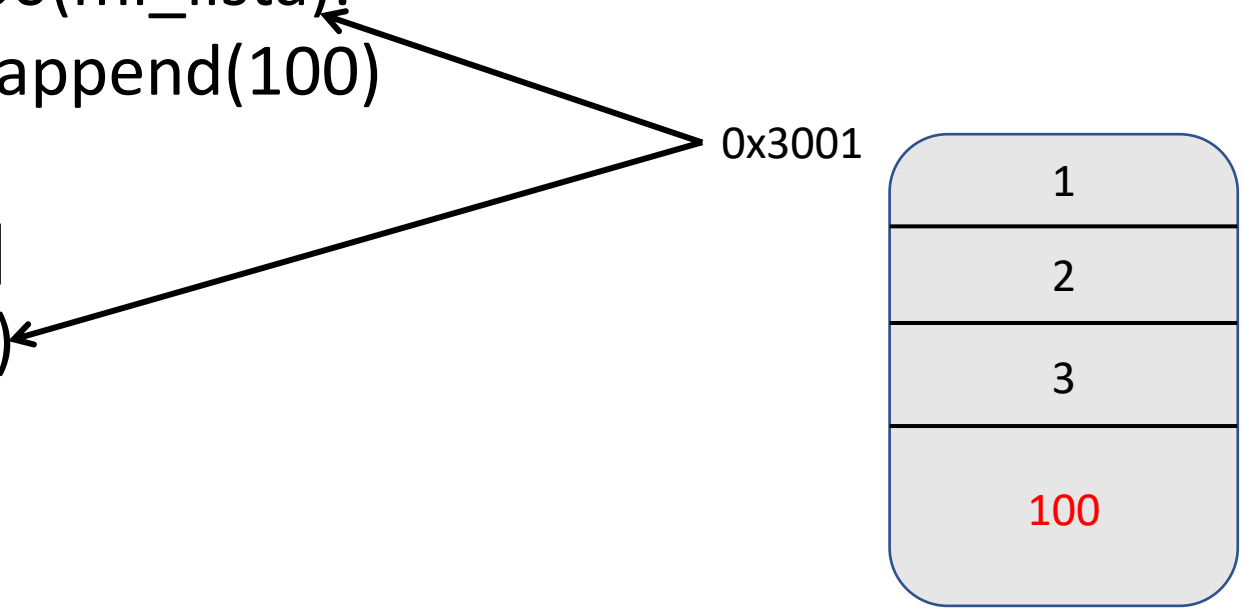


La tupla contiene referencias a objetos mutables

Mutabilidad: Funciones

```
def add_100(mi_lista):
    mi_lista.append(100)

a = [1, 2, 3]
add_100(a)
```



Mutabilidad: Shared references

- Cuando asignamos a una variable otra se comparte la referencia !!!
- En caso de inmutables además se comparte la referencia al realizar asignación con el objeto mismo !!!