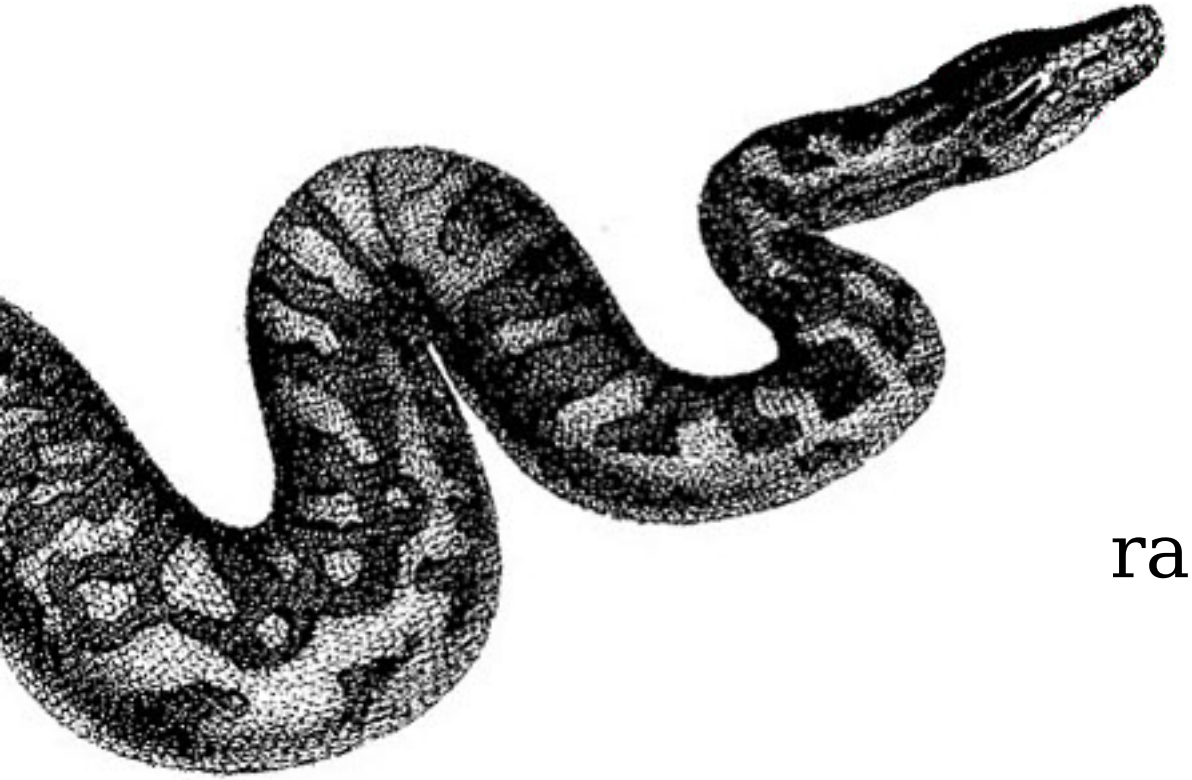
Python:
variáveis e referências



Luciano Ramalho
ramalho@python.pro.br

PythonPro

Variáveis

- Variáveis contém referências a objetos
  - variáveis **não** “contém” os objetos em si
- Variáveis não têm tipo
  - os objetos aos quais elas se referem têm tipo
- Uma variável não pode ser utilizada em uma expressão sem ter sido inicializada
  - não existe “criação automática” de variáveis

Atribuição

Forma simples

```
reais = euros * taxa
```

Outras formas

atribuição com operação

```
a+=10   # a=a+10
```

atribuição múltipla

```
x=y=z=0
```

atribuição posicional itens de sequências

```
a,b,c=lista
i,j=j,i  # swap
```

Atribuição

Exemplo

```
# Série de Fibonacci

a = b = 1
while True:
    print a
    a, b = b, a + b
```

PythonPro

Atribuição: princípios

Python trabalha com referências, portanto a atribuição não gera uma cópia do objeto

- Uma variável **não** é uma caixa que contém um valor (esqueça esta velha idéia!)
- Uma variável é uma etiqueta Post-it colada a um objeto (adote esta nova idéia!!!)

del: comando de desatribuição

- remove uma referência ao objeto
- não existindo mais referências, o objeto é varrido da memória

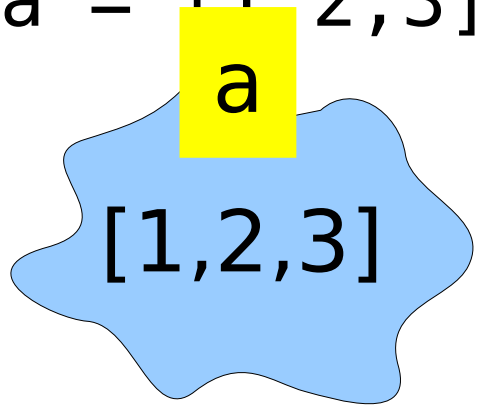
Variáveis

Podem ser entendidas como rótulos

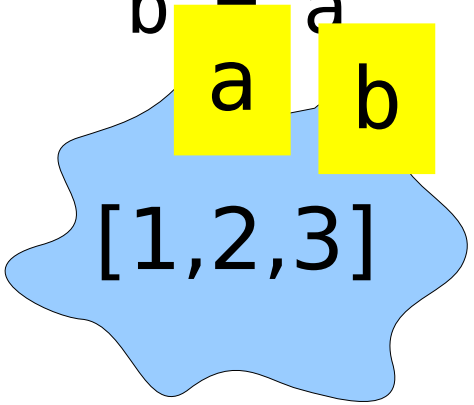
não são "caixas que contém valores"

Atribuir valor à variável equivale a colar um rótulo no valor

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




```
b = a
```



PythonPro

Apelidos e cópias

```
>>> a = [21, 52, 73]
>>> b = a
>>> c = a[:]
>>> b is a
True
>>> c is a
False
>>> b == a
True
>>> c == a
True
```

a e **b** são apelidos do mesmo objeto lista

c é uma referência a uma cópia da lista

```
>>> a, b, c
([21, 52, 73], [21, 52, 73], [21, 52, 73])
>>> b[1] = 999
>>> a, b, c
([21, 999, 73], [21, 999, 73], [21, 52, 73])
>>>
```