

# Live de Python #17

Containers nativos

# Roteiro

- Desempacotamento
- Iteração
- Listas
- Tuplas
- Conjuntos
- Conjuntos congelados
- Dicionários
- `from collections import *`

Se der tempo:

- Algumas funções de iteráveis
  - `enumerate()`
  - `reversed()`
  - `iter()`
  - `sorted()`

# Desempacotamento

```
>>> x = ( 1, 2, 3, 4, 5 )
```

```
>>> *a, b, c = x  
# (1, 2, 3) 4 5
```

```
>>> a, *b, c = x  
# 1 (2, 3, 4) 5
```

```
>>> a, b, *c = x  
# 1, 2, (4, 5)
```

```
>>> *a, b, *c = x  
# (1, 2) 3 (4, 5)
```

# Desempacotamento

```
>>> x = ( 1, 2, 3, 4, 5 )
```

```
>>> *a, b, c = x  
# (1, 2, 3) 4 5
```

```
>>> a, *b, c = x  
# 1 (2, 3, 4) 5
```

```
>>> a, b, *c = x  
# 1, 2, (4, 5)
```

```
>>> a, b, *c = x  
# 1, 2, (4, 5)
```



# For

```
>>> for e in <iter>:  
    do  
else:  
    do
```

```
>>> for e in "grupy":  
    print(e)
```

```
>>> for e in [1,2,3]:  
    print(e)
```

```
>>> for i,e in enumerate([1,2,3]):  
    print(i,e)
```

```
# 0 1  
# 1 2  
# 2 3
```


# Listas

**Sim, elas aceitam tudo**

# Elas aceitam todos os tipos de objeto - Listas

```
>>> a = [1, 1. + ij, "eduardo", [1,2,3], (1,2)]
```

```
>>> a[0] # 1  
>>> a[1] # 1. +j  
>>> a[2] # "eduardo"  
>>> a[3] # [1,2,3]
```



```
>>> a[3][0] # 1  
>>> a[1][1] # 2  
>>> a[4][0] # 1  
>>> a[2][-1] # "o"
```

# Slice - Listas

```
>>> n = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> n[0]      # 0  
>>> n[6:]     # [6, 7, 8, 9]  
>>> n[:-6]    # [0, 1, 2, 3]  
>>> n[::2]    # [0, 2, 4, 6, 8]
```



**[De onde : até onde: de quanto em quanto]**

**[2 : 10 : 3]**



```
>>> n[::2] # [0, 1, 2, 3]  
>>> n[::2] # [0, 2, 4, 6, 8]
```

## Slice - Listas

```
>>> matriz = [ [0, 1, 2], [3, 4, 5], [7, 8, 9] ]
```

```
>>> matriz[0]                # [0, 1, 2]
```

```
>>> matriz[0][1:]           # [1, 2]
```

```
>>> matriz3d = [ [ [0, 0, 0] ], [ [0, 0, 0] ] ]
```

```
>>> matriz3d[0][0][0]       # 0
```

# Métodos - Listas

```
>>> x = [1, 2, 3]
```

```
>>> x.append(4)  
# [1,2,3,4]
```

```
>>> x.remove(2)  
# [1, 3]
```

```
>>> x.insert(4)  
# [4,1,2,3]
```

```
>>> x.pop()  
# [1, 2]
```

```
>>> x.count(2)  
# 1
```

```
>>> x.reverse()  
# [3, 2, 1]
```

# Métodos - Listas

**Fila**

```
>>> x = [1, 2, 3]
```

```
>>> x.append(4)  
# [1,2,3,4]
```

```
>>> x.insert(4)  
# [4,1,2,3]
```

```
>>> x.count(2)  
# 1
```

**Lista**

```
>>> x.remove(2)  
# [1, 3]
```

```
>>> x.pop()  
# [1, 2]
```

```
>>> x.reverse()  
# [3, 2, 1]
```

**Pilha**

# Tuplas

**Elas não são só listas imutáveis**

# Métodos - Tuplas

```
>>> x = (1, 2, 3)
```

```
>>> x.count(2)  
# 1
```

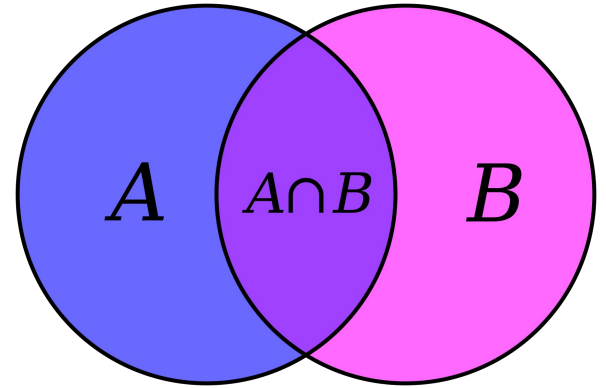
```
>>> x.index(2)  
# 1
```

```
>>> x = 1  
>>> y = 2  
>>> x,y = y,x
```

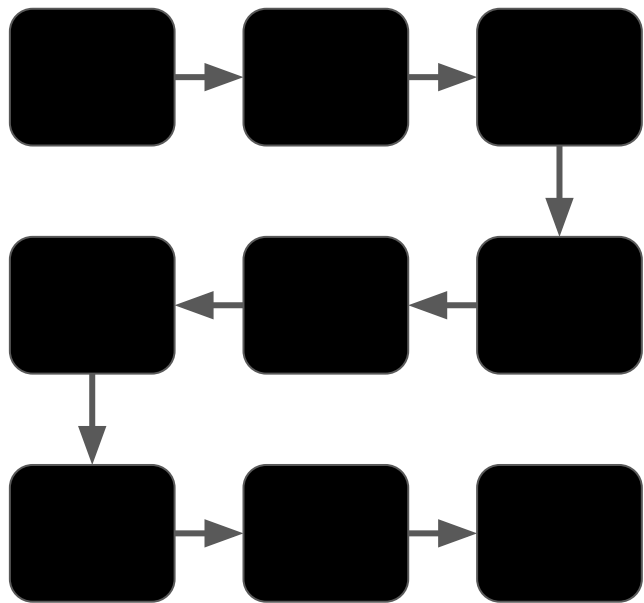
```
>>> print(x)  
# 2  
>>> print(y)  
# 1
```

# Conjuntos

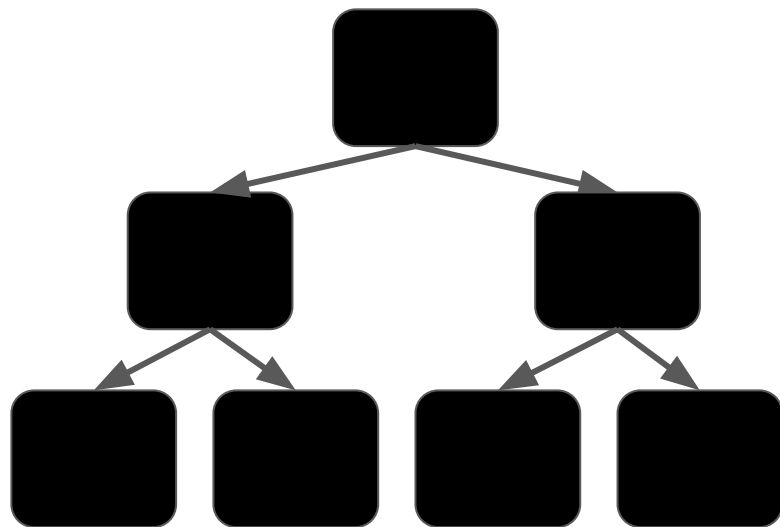
Valores “fixos”



# Que raios são hashable? - Conjuntos



**Litas e Tuplas**



**Conjuntos e dicionários**



# Métodos - Conjuntos

```
>>> x = {1, 2, 3}; y = {3, 4, 5}
```

```
>>> x.union(y)  
# {1, 2, 3, 4, 5}
```

```
>>> x.intersection(y)  
# {3}
```

```
>>> x.difference(y)  
# {1, 2}
```

```
>>> x.update(y)  
# {1, 2, 3, 4, 5}
```

```
>>> x.discard(1)  
# {2, 3}
```

```
>>> x.pop()  
# {2, 3}
```

# Métodos - Conjuntos

```
>>> x = {1, 2, 3}; y = {3, 4, 5}
```

Retorna um novo

```
>>> x.union(y)  
# {1, 2, 3, 4, 5}
```

```
>>> x.difference(y)  
# {1, 2}
```

Remove o 1º

```
>>> x.discard(1)  
# {2, 3}
```

```
>>> x.intersection(y)  
# {3}
```

Atualiza x

```
>>> x.update(y)  
# {1, 2, 3, 4, 5}
```

Remove o 1º

```
>>> x.pop()  
# {2, 3}
```

# Dicionários

**Programador: Humano responsável por transformar café em linhas de código**

# Como eles funcionam? - Dicionários

- A chave necessariamente deve ser hashable
- Os valores podem ser qualquer coisa
- Similar à um JSON

```
Pessoa = {  
    'nome': 'eduardo',  
    'cargo': 'programador',  
    'função': lambda f, *args: f(args),  
    'saldo': {'dia 5': 150, 'dia 10': [0, -100, -500]}  
}
```

# Como uso esse negócio? - Dicionários

```
>>> x = {'Maria': 50, 'Juana': 25}
```

```
>>> x['Maria']  
# 50
```

```
>>> x.keys()  
# ['Juana', 'maria']
```

```
>>> x['Maria'] = 10
```

```
>>> x.values()  
# [50, 25]
```

```
>>> x.popitem()  
# ('Juana', 25)
```

```
>>> x.setdefault('Carlos')  
# {'Juana': 25, 'maria': 100, 'Carlos':  
  None}
```