Compreensão Listas

Prof. Alberto Costa Neto

Programação em Python

CC BY

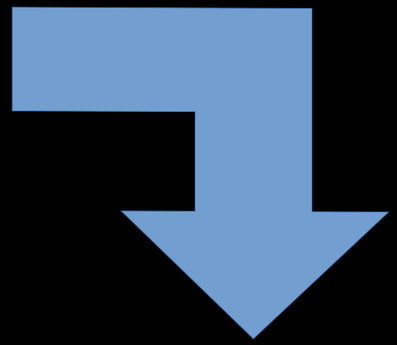
Compreensão de Listas: Qual é a utilidade?

- Às vezes queremos gerar uma lista resultante da aplicação de uma expressão ou função sobre um certo conjunto de valores contidos em uma lista.
- Por exemplo, dada uma lista de preços (números reais), aplicar um desconto (20%) e gerar uma lista com os preços finais

```
>>> precos = [100.0, 200.0, 500.0, 800.0]
>>> precos_desc = []
>>> for v in precos:
...     precos_desc.append(v*0.8)
...
>>> print(precos_desc)
```

Compreensão de Listas: Qual é a utilidade?

```
>>> precos = [100.0, 200.0, 500.0, 800.0]
>>> precos_desc = []
>>> for v in precos:
...     precos_desc.append(v*0.8)
...
>>> print(precos_desc)
```



Com compreensão de Listas fica muito mais simples

```
>>> precos = [100.0, 200.0, 500.0, 800.0]
>>> print( [v * 0.8 for v in precos] )
```

Mais alguns exemplos

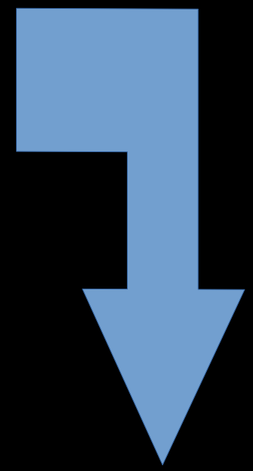
```
>>> # Lista com o quadrado de cada valor da lista
>>> print( [x ** 2 for x in range(11)] )
[0,1,4,9,16,25,36,49,64,81,100]
>>>
>>> # Lista com o sucessor do triplo de cada valor da lista
>>> print( [3 * x + 1 for x in range(11)] )
[1,4,7,10,13,16,19,22,25,28,31]
>>>
```

Aplicando um Filtro

```
>>> # Lista com valores ímpares que sejam sucessores
>>> # do triplo de cada valor de 0 a 10
>>> print( [3 * x + 1 for x in range(11) if x%2 == 0] )
[1, 7, 13, 19, 25, 31]
>>>
```

Expressando conjuntos com Compreensão de Listas

$A = \{x^2 \mid x \in N \wedge x \text{ é ímpar} \wedge x < 100\}$

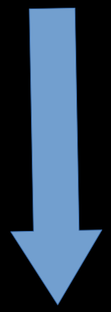


Com compreensão de Listas ficaria

```
A = [ x ** 2 for v in range(100) if x % 2 == 1 ]
```

Migrando de um laço for para Compreensão de Listas

```
res = []
for v in lista:
    if <EXPR2>:
        res.append(<EXPR1>)
```



Com compreensão de Listas ficaria

```
res = [ <EXPR1> for v in lista if <EXPR2> ]
```

Migrando de um laço for para Compreensão de Listas

```
res = []
for v in lista:
    if <EXPR2>:
        res.append(<EXPR1>)
```

Com compreensão de Listas ficaria

```
res = [ <EXPR1> for v in lista if <EXPR2> ]
```

```
res = []
for v in [1,2,3,4,5]:
    if v > 2:
        res.append(v*2)
```

```
res = [v*2 for v in [1,2,3,4,5] if v > 2]
```

Exemplo de substituição do for

- Dados 2 inteiros (X e Y) na mesma linha, imprimir os múltiplos de 5 de X a Y separados pelo caractere |

```
valores = input().split()
X = int(valores[0])
Y = int(valores[1])
# gera uma lista com os múltiplos de 5 de X a Y
mult_de_5 = [n for n in range(X, Y+1) if n%5 == 0]
# converte os valores da lista de int para string
mult_de_5_str = [str(v) for v in mult_de_5]
# imprime os valores da lista separados por |
print('|'.join(mult_de_5_str))
```

Usar ou não usar, eis a questão!

+ Compacto
+ Difícil de entender
- Flexível que um laço for
- Nem todas as linguagens de programação suportam

http://sorisomail.com/img/1288793725561.jpg