# Python

## Funções e parâmetros

Prof. Dr. Dieval Guizelini

Analista e Desenvolvedor de Sistemas

Mestre em Bioinformática  e Doutor em Ciências-Bioquímica

dieval at ufpr.br / dievalg at gmail.com

# Declaração de rotina / função

- Uma função, geralmente, é formada por 3 elementos: nome, parâmetros e “corpo”.
  Exemplo:
  def ola( nome ):
  print(‘oi ‘, nome)

```
>>> ola( 'fulano de tal' )
Olá  fulano de tal
```

- Uma rotina é um bloco de instruções nomeado que não retorna nenhuma informação, enquanto que função e um bloco de instruções nomeado que retorna um valor.
  No caso acima, temos uma rotina e abaixo uma função:

def quadrado( n ):
return( n*n )

```
>>> def quadrado( n ):
        return( n * n )

>>> quadrado(3)
9
```

# Passagem de parâmetros

- Por valor: quando o conteúdo é copiado do programa chamador para a função, e qualquer alteração realizada na função não modifica o conteúdo da variável original. Exemplo:

```
>>> def maisUm( n ):
        n += 1
        print( n )


>>> x = 1
>>> maisUm( x )
2
>>> x
1
```

- Por referência: quando o endereço da variável  é passado para a função, de forma que não ocorre a cópia do conteúdo e toda modificação realizada durante a execução da função modifica o conteúdo correspondente ao programa chamador.

# Parâmetros opcionais e variados

- def func(a, b=1, *args, **kargs):
  print 'a = %s\nb = %s\nargs = %s\nkwargs = %s' % (a,b,args,kwargs)

```
>>> def func(a, b=1, *args, **kargs):
        print('a = %s\nb = %s\nargs = %s\nkwargs = %s' % (a,b,args,kargs))


>>> func( 1 )
a = 1
b = 1
args = ()
kwargs = {}
>>> func( 1, 2, 'casa' )
a = 1
b = 2
args = ('casa',)
kwargs = {}
>>> func( 1, 2, 'e', parl='a' )
a = 1
b = 2
args = ('e',)
kwargs = {'parl': 'a'}
```

# Variável local x global

- Nesse exemplo, x foi definida e atribuído o valor 1 fora de qualquer função, portanto ele é global.
- n é o nome do parâmetro e só existe no contexto da função variavelGlobal, ela é classificada como escopo local.

```
>>> def variavelGlobal( n ):
        print(x,n)


>>> variavelGlobal( 2)
1 2
>>> x
1
```

```
>>> x = 5
>>> def varGlobal2( n ):
        x = 3
        print(n,x)


>>> varGlobal2(1)
1 3
>>> x
5
```

# Mais global...

- Quando precisamos “inspecionar/exportar” uma variável local a uma função...

- Exemplo (ocorre erro):

```
def add(value1, value2):
    result = value1 + value2

add(2, 4)
print(result)
```

# Mais global...

- Exemplo (ocorre erro):

```
def add(value1, value2):
    result = value1 + value2

add(2, 4)
print(result)
```

- Solução:

```
def add(value1, value2):
    global result
    result = value1 + value2

add(2, 4)
print(result)
>>> 6
```

# Funções de bibliotecas (built-in)

- A biblioteca de funções precisa ser “importada” para o seu programa. Exemplo:

```
import math
print( math.sqrt(9) )
```

# Lambda: é uma expressão que pode substituir uma função

- *Lambda* frequentemente é chamado de “funções anônimas” que aceitam argumentos (inclusive opcionais) e que só suportam uma expressão.

- Ao executar *lambda*, Python retorna uma função ao invés de atribuí-la à um nome como acontece com *def,* por isso são anônimas.

- O conceito e o nome são emprestados de Lisp.

- Pense no lambda com “funções de uma linha só”.

# Sintaxe

- **lambda** parâmetro: faz_algo_com(parâmetro)

- Exemplo 1:
  soma = **lambda** x, y: x + y
  print( soma(2,3) )

# Exemplo

```
def length_each(words):
        return [len(w) for w in words]

# utilizando a função length_each
palavras = ['teste', 'sol', 'carro', 'ilha', 'lua']


print( length_each(palavras) )
>>> [5, 3, 5, 4, 3]

length_each_lambda = lambda words: [len(w) for w in words]

print(length_each_lambda (palavras)  )

>>> [5, 3, 5, 4, 3]
```

# Exemplo: Um “sequence assembly” com lambdas

```
cm = lambda d,ds: max([m(d,e) for e in ds if e != d])

m = lambda d,e: max([(s(d,e,o),o,e,d) for o in range(1-len(e),len(d))])

s = lambda d,e,o: sum([1 for p in range(max(0-o,0), min([len(e)-o, len(e), len(d)-o])) if e[p] ==
d[p+o]])

con = lambda x,o,s,c : c[0:max(0,o)] + s +  c[len(s)+o:]

a = lambda s, o : con(*cm(s, o)) if len(o) == 1 else a(con(*cm(s, o)), [ y for y in o if y != cm(s,
o)[2]])

ah = lambda d : a(d[0],d[1:])
```

# Executando...

```
reads = ['TCCCAGTGAACCCA', 'TTCCGTGCGGTTAAG', 'GTCCCAGTGAACCCACAA', 'TGAACCCACAAAACG',
'ACCCACAAAACGTGA', 'GAACCCACAAAACGTGA', 'TCCGTGCGGTTAAGC', 'TGAACCCACAAA',
'CCGTGCGGTTAAGCGTGA', 'TGACAGTCCCAGTGAA', 'AACCCACAAAACGTGA', 'AGTGAACCCACAAAACGT',
'GTTAAGCGTGA', 'CCGTGCGGTTAAGCGTGA', 'AGCGTGACAGT', 'TGCGGTTAAGCG', 'ACAAAACGTGATG',
'ACAGTCCCAGTGAACC', 'TAAGCGTGACAGTCCCA', 'TCGAATTCCGT', 'TTCTCGAATTCCGTGCG', 'ACAAAACGTG',
'CCACAAAACGTG', 'TGCGGTTAAG', 'GAACCCACAAAACGTGA', 'TCTCGAATTCC', 'ATTCCGTGCGGTTAA',
'ACCCACAAAAC', 'CGTGCGGTTAAGCGTGA', 'CCAGTGAACCCACAA', 'TGCGGTTAAGCGTG', 'CCCACAAAACG',
'TCTCGAATTC', 'AATTCCGTGCGGTT', 'ACAGTCCCAGTGA', 'GTCCCAGTGAACCCA', 'TGAACCCACAAA',
'CCCACAAAACGTG', 'TCCCAGTGAACCCACA', 'CTCGAATTCCGTGCG']


print(ah(reads))
# TGCGGACAAAACGTGTGAACGTGAGGTTAAGCGTGACAGTCCCAGTGAACCCACAAAACG
```

# Referências

- Princípios funcionais
  https://wiki.python.org.br/PrincipiosFuncionais
- Programação funcional
  https://wiki.python.org.br/ProgramacaoFuncional

- Exemplo do montador:
  https://pythonforbiologists.com/a-terrible-genome-assembler-in-six-lines/

- Mais sobre parâmetros
  https://realpython.com/python-pass-by-reference/