

85ª EDIÇÃO

SEQ UFRJ

20 a 24 de agosto



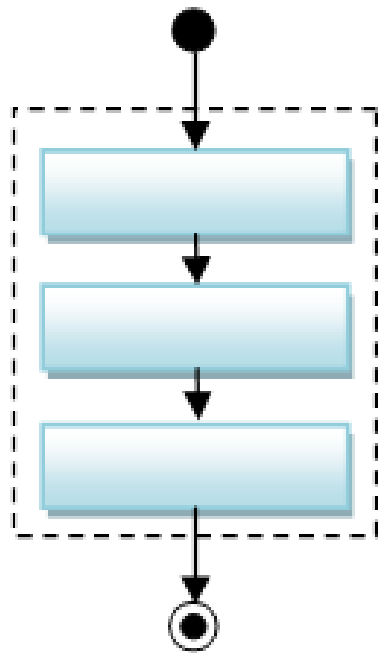
Introdução à programação para ciência e engenharia em *Python*

Iuri Soter Viana Segtovich

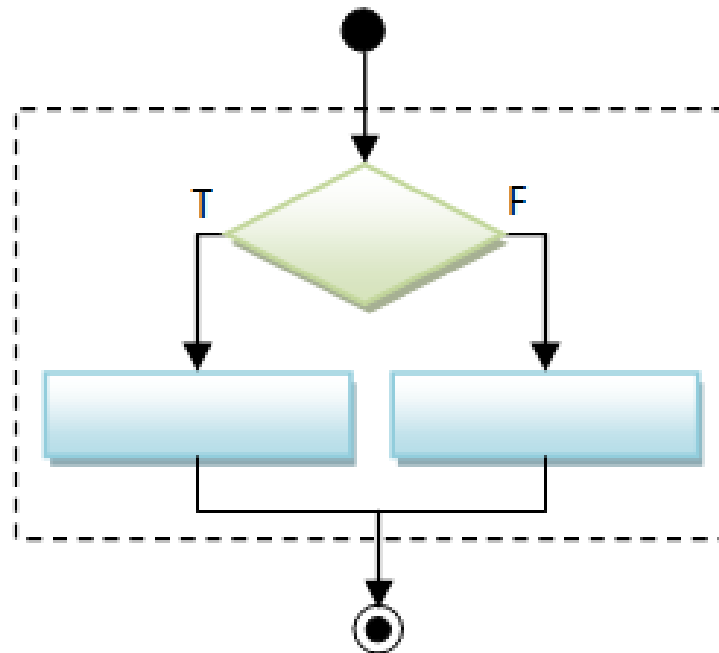
RESUMO

Lógica e sintaxe

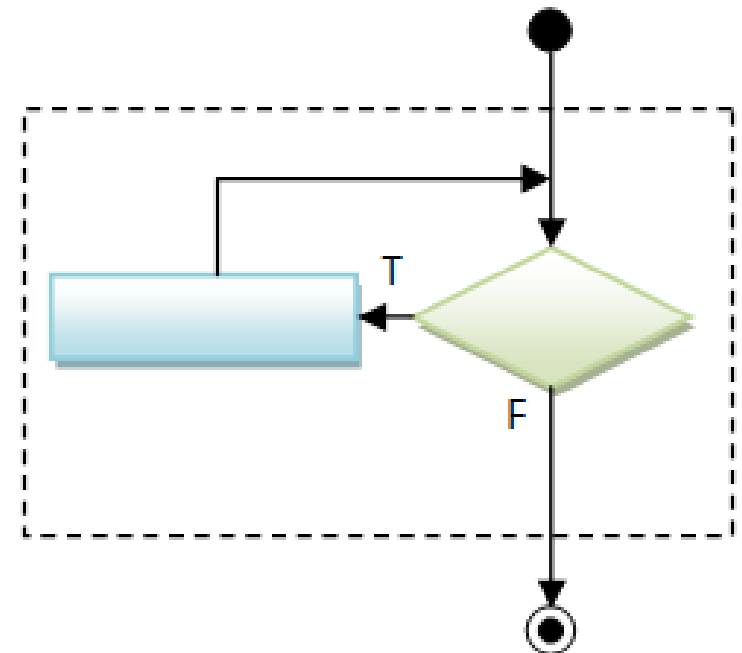
Lógica



Sequential



Conditional (Decision)



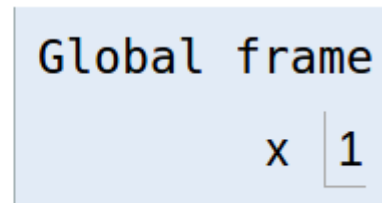
Loop (Iteration)

Vínculos: nomes e objetos

- `x=1` vincula temporariamente o nome `x` ao

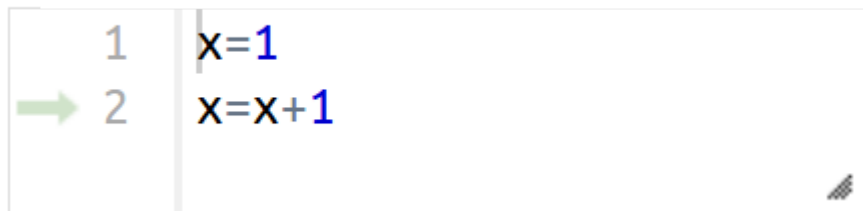


```
1 x=1
2 x=x+1
```

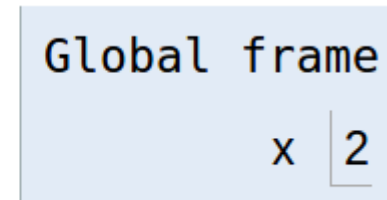


```
Global frame
x 1
```

- `x=x+1` faz o cálculo de `x+1`, a direita do `=`, e revincula o nome `x`, à esquerda do `=`, ao resultado desse cálculo



```
1 x=1
2 x=x+1
```



```
Global frame
x 2
```

condicionais

- O conteúdo de um bloco condicional "if" é executado apenas se a condição for satisfeita, e o conteúdo do bloco "else" adjunto é executado apenas se a condição não for satisfeita.

```
1 mf=4
2 if mf>=5:
3     print('aprovado')
4 else:
→ 5     print('reprovado')
```

aprovado

Iterações

- O conteúdo de um bloco iterativo "for" é executado um vez para cada valor do contador i na sequência.

```
→ 1 for i in (0,1,2):  
   2     print(i)
```

```
0  
1  
2
```

- O conteúdo de um bloco iterativo while é executado várias vezes até que determinada condição seja satisfeita

```
1 i=0  
→ 2 while(i<3):  
3     print(i)  
4     i=i+1
```

```
0  
1  
2
```

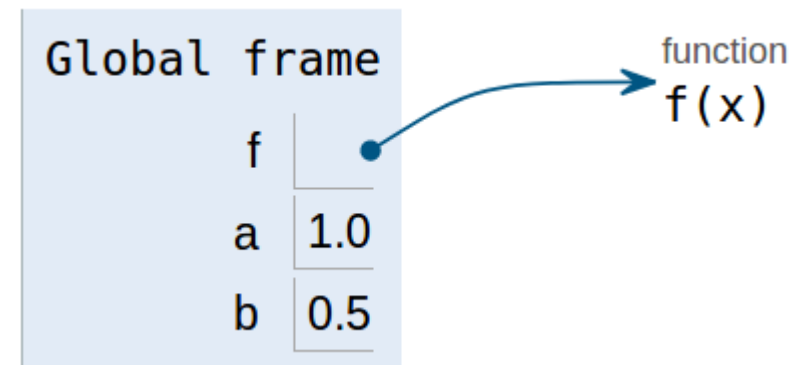
funções

- O conteúdo de um bloco função é executado sempre que a função é chamada, usando os argumentos fornecidos a cada vez.

```
1  def f(x):  
2      y=1/x  
3      return y  
4  a=f(1)  
→ 5  a=f(2)
```

f	
x	1
y	1.0
Return value	1.0

f	
x	2
y	0.5
Return value	0.5



Sintaxe python

```
print("oi, tudo bem? ", "tudo certinho?")
```

- Função print
- Parenteses
- Um ou mais strings separados por vírgula

- resposta=input()

- Função input
- Parentese vazios
- Resposta vem como string

Sintaxe python

- tipos - expressão literal - função de conversão
- Inteiro – 1; -1; 10_000 – int(argumento)
- Float – 1.; 1.5e-3; 2.4E8 – float(argumento)
- Complexo – 3j+1; 1.6j+2; 1j - complex(argumento)
- string – "um"; 'dois'; ""três""; ""quatro"" – str(argumento)
- lógico – True; False – bool(argumento)
- Tupla – 1,2,3; (1,2,3); (1,) – tuple(argumento)
- Lista – [1,2,3]; [1] - list(argumento)
- Conjunto – {1,2,3}; {0} - set(argumento)
- Dicionário – {'um':1,'dois':2,'tres':3}; {} - dict(key=value,key=value, ...); dict(lista de tuplas)

operadores

- Cálculos; calculos consigo mesmo
 - $x=a+b$; $x+=a$
 - $x=a-b$; $x-=a$
 - $x=a*b$; $x*=a$
 - $x=a/b$; $x/=a$
 - $x=a**b$; $x**=a$
- Divisão de inteiros
 - $i=a\%b$
 - $i=a//b$

operadores

- Comparações entre numeros
- $x==y$
- $\text{abs}(x-y)\leq 1\text{e-}14$
- $!=$
- $\text{abs}(x-y)>1\text{e-}14$
- $>$
- $<$
- $>=$
- $<=$

- Tabela lógica
- And
- Or
- not

A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Index and slicing

- Strings, Tuplas e listas
- Indexing com colchetes, e slicing com colchetes e dois pontos.
- `T[0]`; `t[1]`; `t[2]` ...
- `T[-1]`; `t[-2]`; `t[-3]` ...
- Slice
- `T[0:3]`; `t[0:]`, `t[:3]`; `t[::2]`; `t[-1:-11]`

```
Index from rear:  -6  -5  -4  -3  -2  -1
Index from front:  0   1   2   3   4   5
+---+---+---+---+---+---+
| a | b | c | d | e | f |
+---+---+---+---+---+---+
Slice from front:  :   1   2   3   4   5   :
Slice from rear:   :  -5  -4  -3  -2  -1   :
```

Operadores/métodos

- Strings
- Capitalize
- Upper lower

Operadores/metodos

- tuplas

Operadores/metodos

- Modificação de Listas
- l.append(elemento)
- l.extend(outralista)
- l.pop(indice)
- l.sort()

Operadores/metodos

- conjunto

Operadores/metodos

- diccionario

condicional

Iteração

Função

modulos

- Import math as m
- m.sin
- m.cos
- m.log
- m.exp
- m.sqrt
- ...