
Python
I
1 2 3 4 5 6 7 8 9

Básico

Instalação

Terminal

Tipos I

Estruturas de Controle

Tipos II

Funções

Scripts

Leituras Recomendadas

Python

“Python é uma linguagem de programação interpretada, interativa e orientada a objetos, que combina um notável poder com uma sintaxe muito clara.”

Quem Usa Python?

- Google
- Nasa
- Industrial Light & Magic
- Itautec
- INdT
- Você?

Para Quê Se Usa Python?

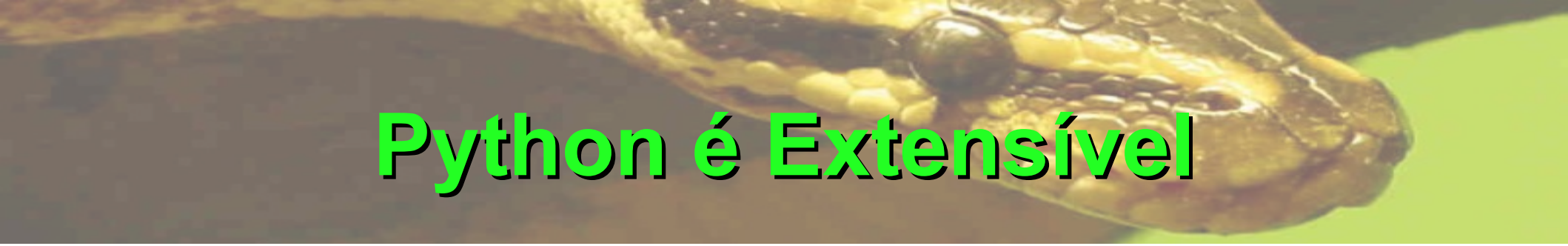
- Desenvolvimento Web e Internet
- Acesso a Bases de Dados
- Desktop GUIs
- Computação Numérica e Científica
- Programação em Rede
- Jogos
- Gráficos 3D
- Modelagem de Hardware
- Educação

Características

- Interpretada
- Portável
- Extensível (C....)
- Uso extensivo de introspecção
- Livre
- Multi-tarefa
- Baixa performance
- Multiparadigma: Procedural, Orientada a objetos, Funcional (em breve Lógica)
- Case-sensitive

Python é Portável

- Python roda em:
  - Windows
  - Linux/Unix
  - MacOS X
  - OS/2
  - Amiga
  - Handhels Palm
  - E nas máquinas virtuais .NET (IronPython) e Java (Jython)



Python é Extensível

- C! C! C!
- Bindings
- Bibliotecas em C disponíveis em Python:
  - PyGame (SDL)
  - PyGTK
  - PyOpenGL
  - muitos eteceteras...

ByteCode Python

- Python compila automaticamente o código fonte para bytecode
- A extensão do arquivo “compilado” é *.pyc
- A máquina virtual de Python pode ser embutida num executável junto com os bytecodes de sua aplicação

```
app.py + python2.4.dll = app.exe
```

Sem Comandos Declarativos

- Todos os comandos em Python fazem alguma coisa, não há comandos declarativos (int, float, etc.) ou comandos para “coisas óbvias” (new):
  - Java
    ```
    Objeto obj = new Objeto()
    ```
  - Python
    ```
    obj = Objeto()
    ```

Código Python

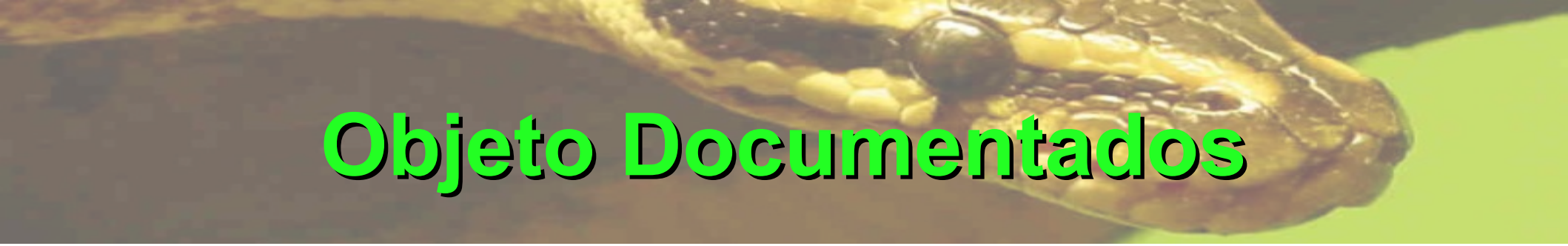
- C

```
int
soma(int a, int b)
{
    int c = a + b;
    return c;
}


print("%d",
      soma(1, 3));
```

- Python

```
def soma(a, b):
    c = a + b
    return c

print "%d" % soma(1, 3)
```

Blocos delimitados por **identação**!



Objeto Documentados

- Não existem tipos primitivos como “int” de Java – tudo em Python é um objeto, incluindo funções e métodos.
- Diferentemente de C++ e Java os comentários de documentação ficam dentro dos métodos e classes, fazendo parte do objeto que documentam.

```
def metodo():
    '''String de documentação.'''

    return “blah”
```


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas

Sessão de Instalação

- Instalação pra Windows, porque qualquer Linux já deve vir com Python
- Baixe o arquivo .msi em

http://www.python.org

Você Não Precisa de uma IDE!

- Informação importantíssima:

Linguagens de programação não são gêmeas siamesas de IDEs!

Uma coisa é uma coisa,
outra coisa é outra coisa.

- Python facilita muito a vida de quem quer usar apenas um editor de texto (com syntax highlighting, é claro!)


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas
1 2 3 4 5 6 7 8 9

Terminal Python

- Python possui um terminal (ou console) que pode ser usado para emitir comandos individualmente, testar a sintaxe, obter ajuda sobre comandos, etc.

- Abra o Terminal e ....

```
>>> print “Hello World”
Hello World
>>> a = “Hello World”
>>> a
'Hello World'
```

Comandos Úteis

- **dir**: mostra os métodos do objeto indicado

```
>>> a = “teste”
>>> dir(a)
[........., 'strip', 'swapcase',
'title', 'translate', 'upper',
'zfill']
>>> a.strip
<built-in method strip of str
object at 0xb789b9c0>
```

- **help**: mostra a string de ajuda do objeto indicado

```
>>> help(a)
```


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas
1 2 3 4 5 6 7 8 9

Tipagem

- **Dinâmica**

```
>>> a = 10
>>> type(a)
<type 'int'>
>>> a = “uma string”
>>> type(a)
<type 'str'>
```

- **Forte**
  - Não existem “casts”, não é permitido:

```
>>> a = (str) 13
```

  - É necessária uma conversão:

```
>>> a = str(13)
```

Tipos Numéricos

- Tipos

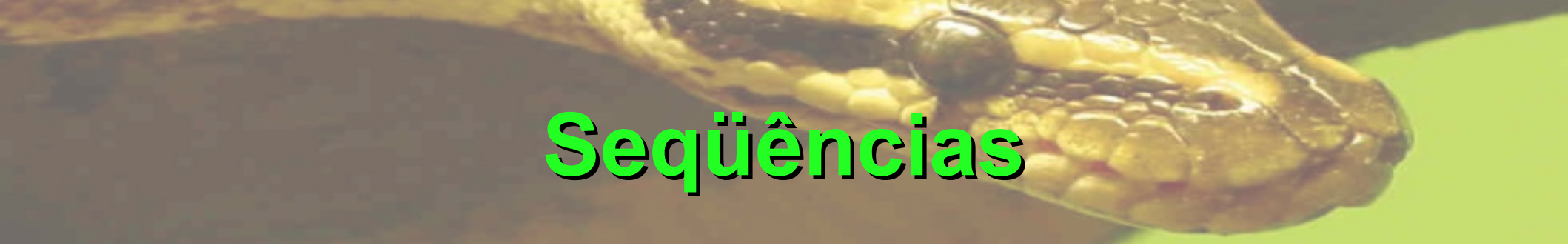
int	Inteiro
long	Inteiro longo de tamanho ilimitado
float	Ponto flutuante
complex	Número complexo
bool	Booleano (True or False)

- O tipo é definido durante a atribuição do valor:

```
>>> a = 1.0
>>> type(a)
<type 'float'>
```

- Ou por um construtor

```
>>> a = float(1)
```



Seqüências

- **str** – Seqüência de caracteres (imutável)

```
>>> a = 'blah'
>>> type(a)
<type 'str'>
>>> a = u'sou uma string unicode: blá'
>>> type(a)
<type 'unicode'>
```

- **tuple** (imutável)

```
>>> a = (1, 2, 'blah')
```

- **list** (mutável)

```
>>> a = []
>>> a = [1, 2, 3, (1, 2), 'boo']
```

Conjuntos & Mapeamentos

- **set** – Conjunto; parecido com lista, mas sem itens repetidos

```
>>> a = set([1, 1, 2, 4, 5, 5])
>>> print a
set([1, 2, 4, 5])
```

- **dict** – tabela hash ou dicionário; composto de pares (chave, valor)

```
>>> a = {'blah':7, 12:(640, 480)}
>>> print a['blah']
8
>>> print a[12]
(640, 480)
```

Comparações

- **is** – identidade
- **is not** – negação da identidade

```
>>> a = []
>>> b = []
>>> b is a
False
>>> b is not a
True
>>> c = a
>>> c is a
True
>>> c is not b
True
```

Básico
Instalação
Terminal
Tipos I

Estruturas de Controle

Tipos II
Funções

if

- C

```
if (a == 1) {
   printf("op1\n");
} else if (a == 2) {
   printf("op2\n");
} else {
   printf("outra\n");
}
```

- Python

```
if a == 1:
   print "op1"
elif a == 2:
   print "op2"
else:
   print "outra"
```

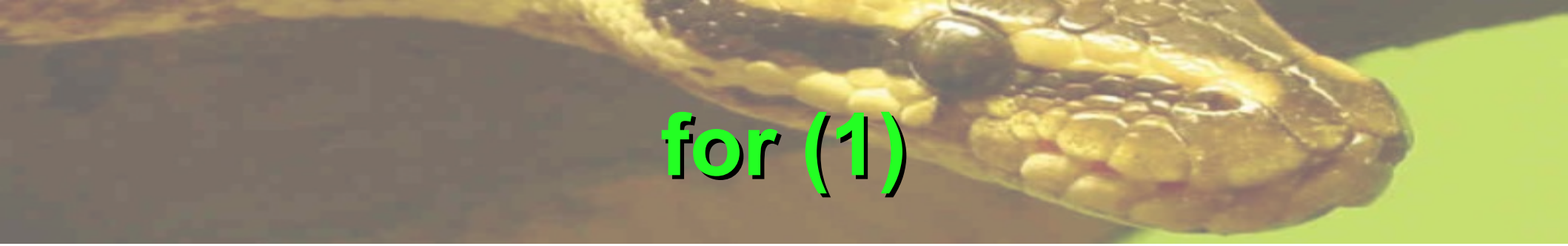
while

- C

```c
while (a < 1) {
    a++;
    printf("a = %d", a);
}
```

- Python

```
while a < 1:
    a += 1;
    print "a = %d" % a
```



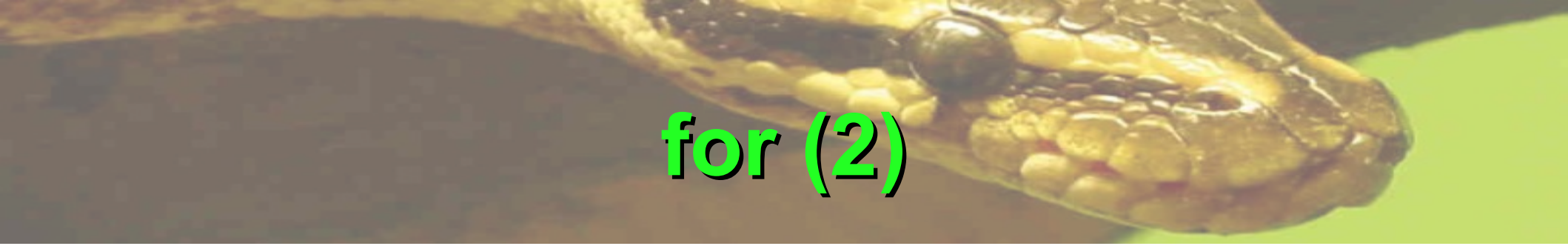
for (1)

- C

```
for (a=0; a < 10; a++) {
   printf("num:%d\n", a);
}
```

- Python

```
for a in range(10):
   print "num:%d" % a
```



for (2)

- C

```
int i;
char *str[] =
      {“item1”, “item2”,
       “item3”, “item4”};

for (i = 0; i < 4; i++) {
   printf(“%s\n”, str[i]);
}
```

- Python

```
str = [“item1”, “item2”,
       “item3”, “item4”]

for s in str:
   print s
```

for em Python é similar ao **foreach** de PHP


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas
1 2 3 4 5 6 7 8 9

Operadores Lógicos

- Lógicos: **a and b, a or b, not a**
- Lógicos bitwise (bit a bit):
  - and: **a & b**
  - or: **a | b**

```
>>> 1 | 2
3
```

  - xor: **a ^ b**
  - shifting: **a >> b, a << b**

```
>>> a = 2
>>> a << 3
16
```

Operações Sobre Seqüências (1)

- **in** – Verifica se um objeto está contido numa seqüência

```
>>> a = 'blahblah'
>>> 'ah' in a
True
>>> 'x' not in a
True
```

- **len** – Tamanho da seqüência

```
>>> len('teste')
5
```

- **s + t** – Concatenação

```
>>> (1, 2, 34) + (3, 4, 6)
(1, 2, 34, 3, 4, 6)
```

Operações Sobre Seqüências (2)

- seq**[n]** – Item n da seqüência

```
>>> a = (1, 3, 5, 7, 8)
>>> a[3]
7
```

- seq**[:n]** – Itens 0 a n, excluindo n

```
>>> a[:3]
(1, 3, 5)
```

- seq**[n:]** – Itens n até o último, incluindo n

```
>>> a[3:]
(7, 8)
```

- seq**[i:f]** – Itens i até f, incluindo i e excluindo f

```
>>> a[2:4]
(5, 7)
```

Operações Sobre Seqüências (3)

- seq**[i:f:j]** – Itens i até f, incluindo i e excluindo f e pulando j itens

```
>>> a = range(10)
>>> a
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> a[0::2]
[0, 2, 4, 6, 8]
```

- **min, max** – Menor e maior item da seqüência, respectivamente

```
>>> min('aknsdmz')
'a'
>>> max('aknsdmz')
'z'
```

Iterando Sobre Itens de Uma Seqüência

- Experimentem isso:

```
>>> seq = range(30)[0::3]
>>> print seq
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
>>> for i in seq:
...     print i
...
```

Operações Sobre Seqüências Mutáveis (Listas)

- **count** – quantas vezes um item aparece na lista

```
>>> a = [1, 43, 87, 7, 4, 87, 5, 7, 98]
>>> a.count(7)
2
```

- **append, remove** – adiciona e remove um item da lista, respectivamente

```
>>> a.append(99)
>>> a.remove(5)
>>> print a
[1, 43, 87, 7, 4, 87, 7, 98, 99]
```

- **pop** – remove e retorna o último item da lista

```
>>> a.pop()
```

Operações Sobre Strings

- **strip** – remove caracteres brancos das extremidades

```
>>> a = '   blah   '
>>> a.strip()
'blah'
```

- **find, rfind** – retorna posição da string, a partir do início e do fim, respectivamente

```
>>> a = 'o rato roeu a roupa do rei'
>>> a.find('ro')
7
>>> a.rfind('ro')
14
```

Operações Sobre Mapeamentos

- **keys** – retorna uma lista com as chaves do mapeamento

```
>>> d = {1:'a', 2:'b', 'tres':'c'}
>>> d.keys()
[1, 2, 'tres']
```

- Exemplo

```
>>> for key in d.keys():
...     print '%s - %s' % (key, d[key])
...
```


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas
1 2 3 4 5 6 7 8 9

Funções

- Exemplo 1:

```
>>> def coisifica (arg1, arg2='Default'):
...     print arg1
...     return 'coisa' + arg2
```

- Exemplo 2:

```
>>> def foo (arg1=7, arg2=1):
...     print 'arg1=%d, arg2=%d'%(arg1, arg2)
...
>>> foo()
arg1=7, arg2=1
>>> foo(6)
arg1=6, arg2=1
>>> foo(arg2=19)
arg1=7, arg2=19
```


Básico
Instalação
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras Recomendadas
1 2 3 4 5 6 7 8 9

O Famigerado Fatorial

```
#!/usr/bin/env python
#-*- coding:utf-8 -*-

def fact(num):
    if num == 1 or num == 0:
        return 1
    elif num > 1:
        return num * fact(num - 1)
    else:
        return None


def main():
    print "Fatorial de 0: %d" % fact(0)
    print "Fatorial de 1: %d" % fact(1)
    print "Fatorial de 6: %d" % fact(6)


if __name__ == '__main__':
    main()
```

Executando Scripts

- No Linux:

```
python script.py
```

ou

```
chmod +x script.py
./script.py
```

- No Windows:
  - se python estiver devidamente instalado, basta clicar duas vezes
  - ou ainda criar um atalho


...
Terminal
Tipos I
Estruturas de Controle
Tipos II
Funções
Scripts
Leituras
Recomendadas
1 2 3 4 5 6 7 8 9

Leituras Recomendadas

- **Tutorial Python**
  - http://www.pythonbrasil.com.br/moin.cgi/DocumentacaoPython?action=AttachFile&do=get&target=python24.pdf
- **Dive Into Python**
  - http://www.diveintopython.org/toc/index.html
- **Site: PythonBrasil**
  - http://www.pythonbrasil.com.br

CinLUG

http://www.cinlug-br.org