

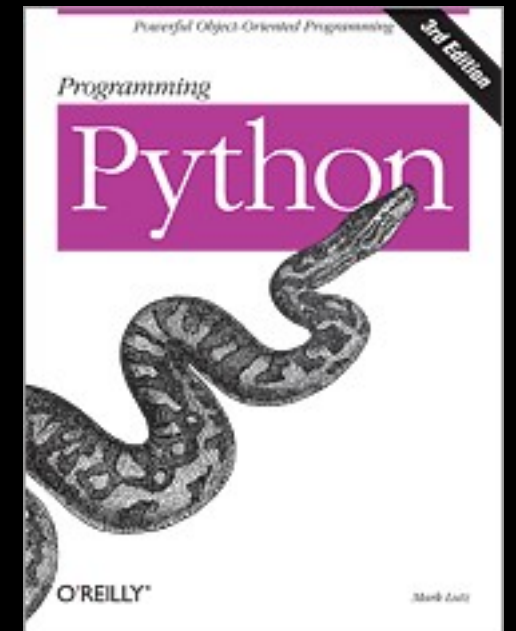
A Linguagem Python: Uma visão geral

Prof. Alberto Costa Neto
Programação em Python

Python é a linguagem do interpretador Python e daqueles que são capazes de conversar com ele.

Um indivíduo que “fala” Python é denominado Pythonista.

Quase todos os Pythonistas conhecidos usam software desenvolvido por Guido van Rossum.



Monty Python ou **The Pythons** ^{[1][2]} é um grupo de comédia britânico, que foram os criadores e intérpretes da série cômica *Monty Python's Flying Circus*, um programa de televisão britânico que foi ao ar pela primeira vez em 5 de outubro de 1969. Como série televisiva, consistiu de 45 episódios divididos em 4 temporadas. Entretanto o fenômeno Python não se limitou a apenas isso, espalhando-se por shows, filmes, programas de rádio e diversos jogos de computador e livros, além de lançar seus seis integrantes ao estrelato.

Iniciantes: Syntax Errors

Precisamos aprender a **linguagem Python** para comunicar nossas instruções ao Python.

No início cometeremos muitos erros, como crianças pequenas aprendendo a falar.



Iniciantes: Syntax Errors

Diferentemente de um bebê, quando você comete erros, o computador não vai lhe achar “lindo”.

Ele irá dizer “**syntax error**” - dado que ele **sabe** a linguagem e você está aprendendo.

Parece que o Python é cruel e sem sentimentos.



Iniciantes: Syntax Errors

Você deve lembrar que *vocês* são inteligentes e *podem* aprender

- o computador é simples e rápido, mas não pode aprender
- então é mais fácil para você aprender Python do que programar o computador para entender Português.



Comunicando-se com o Python

O que diremos?
E como diremos?



Vamos falar com o Python...

```
albertocostaneto — Python — 80x24
Last login: Sat May 20 21:29:33 on console
[MacBook-Pro-de-Alberto-3:~ albertocostaneto$ python
Python 2.7.9 (v2.7.9:648dcafa7e5f, Dec 10 2014, 10:10:46)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
[>>> print 'hello world'
hello world
>>>
```

```
alberto@pc-acn: ~
alberto@pc-acn: ~ 80x24
alberto@pc-acn:~$ python3
Python 3.4.3 (default, Nov 17 2016, 01:08:31)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello world')
hello world
>>>
```



```
alberto@pc-acn:~$ python3
```

```
Python 3.4.3 (default, Nov 17 2016, 01:08:31)
```

```
[GCC 4.8.4] on linux
```

```
Type "help", "copyright", "credits" or "license" for more  
information.
```

```
>>>
```



Próximo
Comando?

```
alberto@pc-acn:~$ python3
```

```
Python 3.4.3 (default, Nov 17 2016, 01:08:31)
```

```
[GCC 4.8.4] on linux
```

```
Type "help", "copyright", "credits" or "license" for more  
information.
```

```
>>> x = 1
```

```
>>> print(x)
```

```
1
```

```
>>> x = x + 1
```

```
>>> print(x)
```

```
2
```

```
>>> exit()
```

Este é um bom teste para certificar-se de que você tem o Python instalado corretamente. Note que `quit()` também encerra a seção de interação.

Elementos de Python

Vocabulário / Palavras – Variáveis e Palavras Reservadas

Estrutura de Sentenças – Padrões de sintaxe válidos

Estrutura de Estória – Construindo um programa com um propósito

Palavras Reservadas

Você não pode usar **palavras reservadas (keywords)** como nomes de variáveis / identificadores

| | | | | |
|--------|----------|---------|----------|--------|
| False | class | finally | is | return |
| None | continue | for | lambda | try |
| True | def | from | nonlocal | while |
| and | del | global | not | with |
| as | elif | if | or | yield |
| assert | else | import | pass | |
| break | except | in | raise | |

```
nome = input('Nome do arquivo:')
arquivo = open(nome, 'r')
texto = arquivo.read()
palavras = texto.split()

contadores = dict()
for palavra in palavras:
    contadores[palavra] = contadores.get(palavra,0) + 1
maior_contador = None
palavra_mais_frequente = None

for palavra,contador in contadores.items():
    if maior_contador is None or contador > maior_contador:
        palavra_mais_frequente = palavra
        maior_contador = contador
print(palavra_mais_frequente, maior_contador)
```

Uma “estória” curta
sobre como contar
palavras em um
arquivo com Python

```
python palavras.py
Nome do arquivo: words.txt
to 16
```

Sentenças ou Linhas

x = 2

x = x + 2

print(x)

Variável

Operador

Constante

Comando de Atribuição

Atribuição com
expressão

Função de impressão

Programando Parágrafos

Interativo versus Script

- **Interativo**
Você pode digitar diretamente uma linha por vez para o Python que ele responde
- **Script**
Você entra com uma sequência de comandos (linhas) em um arquivo usando um editor de texto e diz ao Python que execute os comandos deste arquivo

Scripts Python

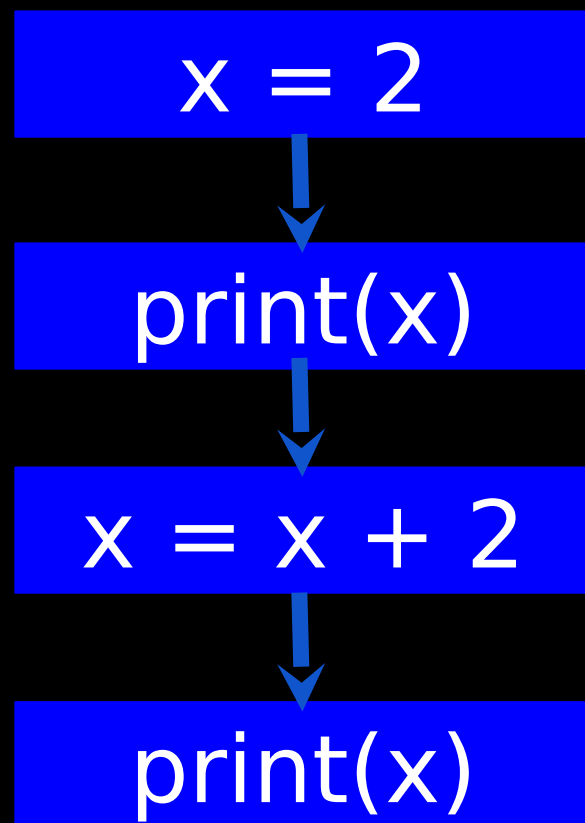
- **Python Interativo** é bom para experimentos e programas de 3 a 4 linhas
- Como a maioria dos programas é bem maior, nós os digitamos em um arquivo e pedimos Python rodar os comandos contidos no arquivo
- De certa forma estamos “dando um script para o Python”
- Como uma convenção, damos uma **extensão “.py”** a estes arquivos para indicar que contêm scripts Python

Escrevendo um Programa Simples

Passos de um Programa ou Fluxo de um Programa

- Como uma receita ou instruções de instalação, um programa é uma seqüência de passos a serem executados em ordem
- Alguns passos são condicionais – podem ser pulados
- Às vezes um passo ou grupo de passos precisa ser repetido
- Algumas vezes armazenamos um conjunto de passos a serem executados várias vezes em muitos pontos de um programa

Passos Seqüenciais



Programa:

`x = 2`

`print(x)`

`x = x + 2`

`print(x)`

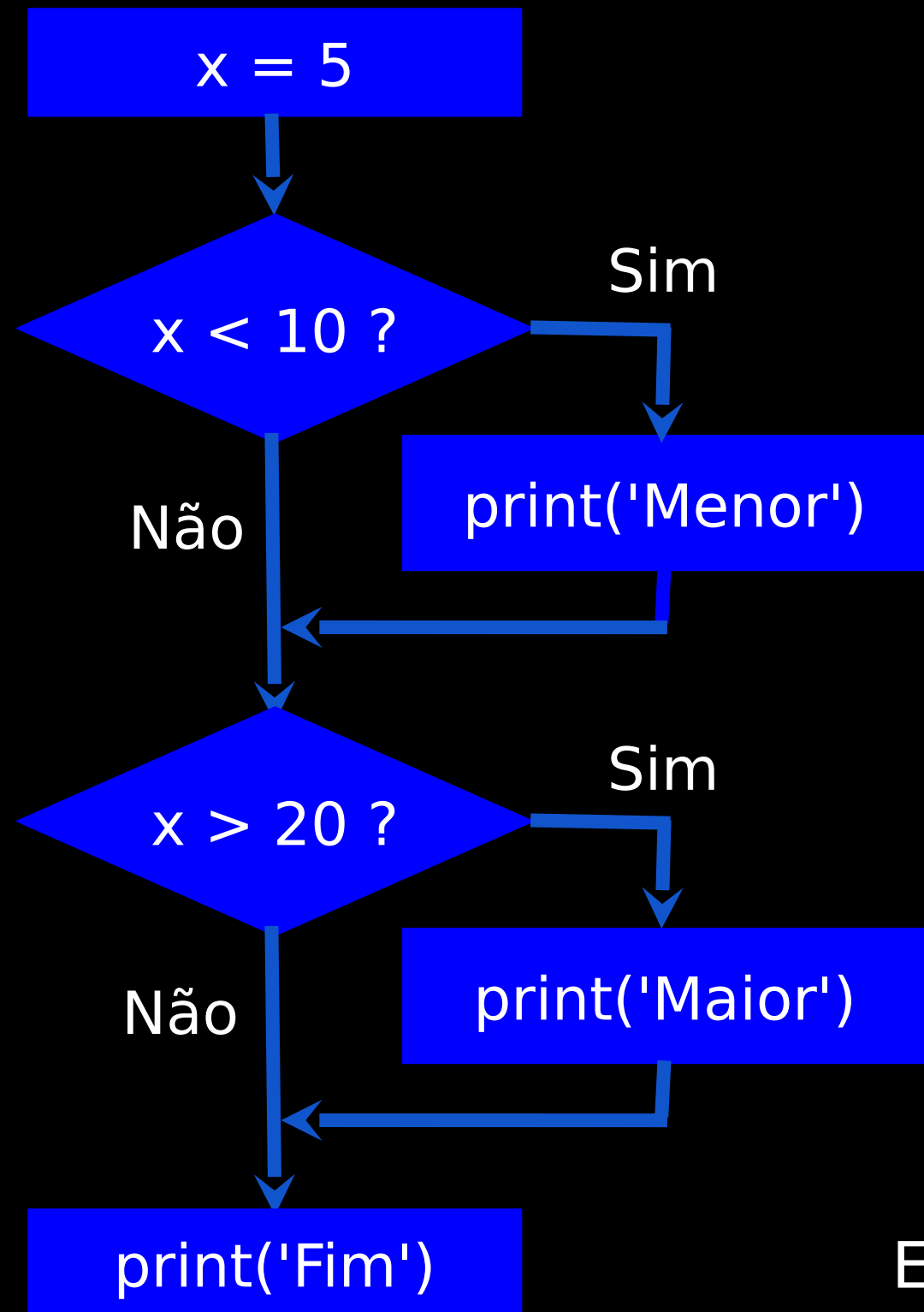
Saída:

2

4

Quando um programa está executando, ele segue o fluxo de um passo para o próximo. Como programadores, nós ajustamos os passos para o programa seguir

Passos Condicionais



Programa:

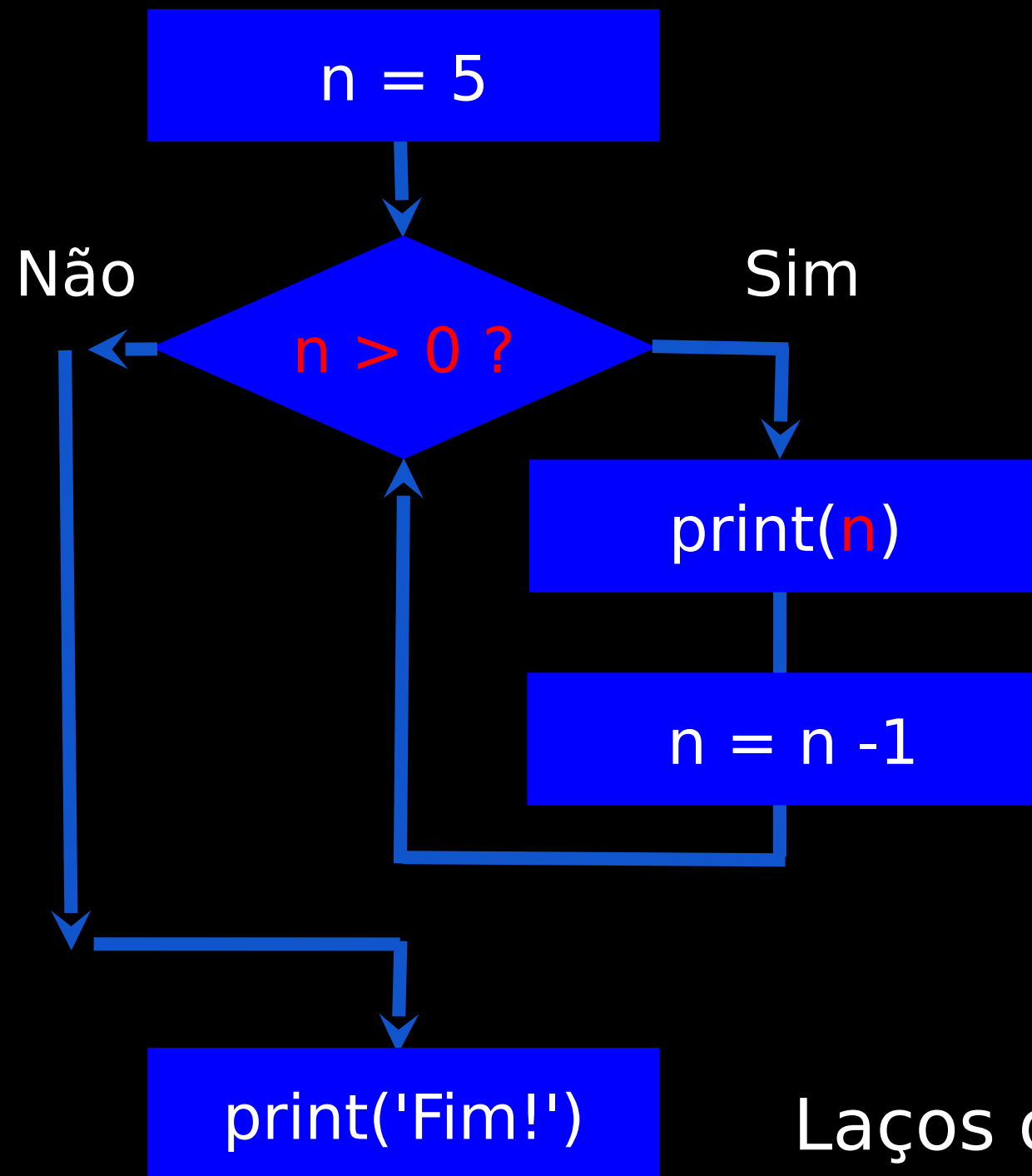
```
x = 5
if x < 10:
    print('Menor')
if x > 20:
    print('Maior')
print('Fim')
```

Saída:

Menor
Fim

Em alguns casos, há passos que só devem ser executados em determinadas condições

Passos Repetidos



Programa:

```
n = 5
while n > 0 :
    print(n)
    n = n - 1
print('Fim!')
```

Saída:

5
4
3
2
1
Fim!

Laços ou Loops (passos repetidos) têm variáveis de **variáveis de iteração** que mudam a cada passada do laço. Frequentemente estas **variáveis de iteração** recebem uma sequência de números.


```
nome = input('Nome do arquivo:')  
arquivo = open(nome, 'r')  
texto = arquivo.read()  
palavras = texto.split()
```

Sequencial

Repetido

Condicional

```
contadores = dict()  
for palavra in palavras:  
    contadores[palavra] = contadores.get(palavra,0) + 1  
maior_contador = None  
palavra_mais_frequente = None  
  
for palavra,contador in contadores.items():  
    if maior_contador is None or contador > maior_contador:  
        palavra_mais_frequente = palavra  
        maior_contador = contador  
  
print(palavra_mais_frequente, maior_contador)
```

Uma curta “estória” em Python sobre como contar palavras em um arquivo

Uma palavra usada para ler um dado do usuário

Uma sentença para atualizar um dos muitos contadores

Um parágrafo sobre como encontrar o maior item de uma lista

```
nome = input('Nome do arquivo:')
arquivo = open(nome, 'r')
texto = arquivo.read()
palavras = texto.split()

contadores = dict()
for palavra in palavras:
    contadores[palavra] = contadores.get(palavra, 0) + 1
maior_contador = None
palavra_mais_frequente = None

for palavra, contador in contadores.items():
    if maior_contador is None or contador > maior_contador:
        palavra_mais_frequente = palavra
        maior_contador = contador

print(palavra_mais_frequente, maior_contador)
```