Itaú Unibanco

Itaú

Programa de formação

ITAú analytics

Módulo I – Fundamentos Computacionais
Sessão 1 - Aula 3 – Visão Geral do Python

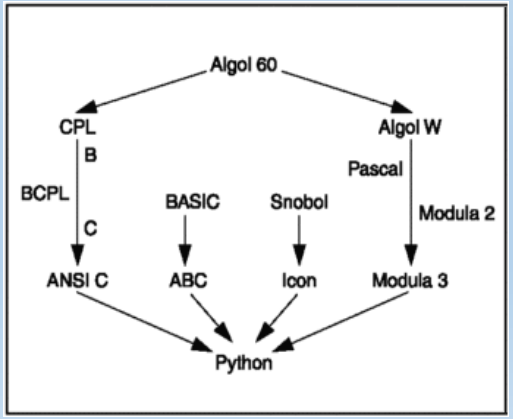
Prof. Dr. Luiz Alberto Vieira Dias
Prof. Dr. Lineu Mialaret

Características do Python

- O que a linguagem de programação Python tem em comum com o alfabeto?

Características do Python

- A linguagem de programação Python foi concebida originalmente por Guido van Rossum no final dos anos 1980 no Instituto de Pesquisa Nacional para Matemática e Ciência da Computação (CWI), na Holanda como sucessora da linguagem ABC capaz de tratar exceções e prover interface com o sistema operacional Amoeba
- Desde então tem resultado numa proeminente linguagem utilizada na indústria e na educação
  - A 2ª versão, Python 2, foi liberada em 2000
  - A 3ª versão, Python 3, liberada em 2008

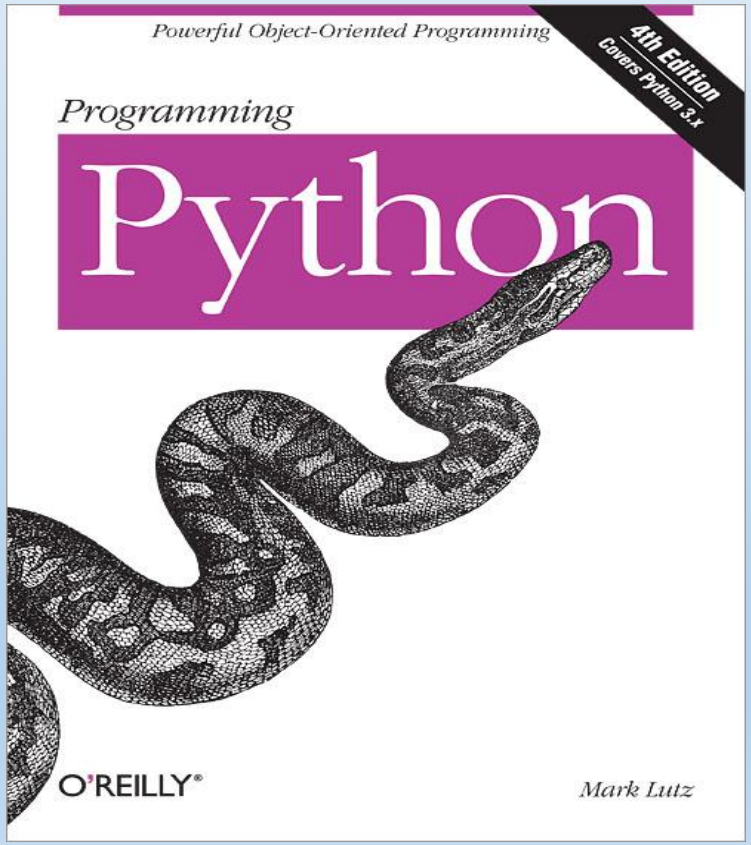


Características do Python

- O nome da linguagem veio do fato dos membros do grupo de Guido van Rossum darem aos softwares desenvolvidos nomes de programas da TV: no caso da linguagem foi o programa *"Monty Python's Flying Circus"*

Características do Python

- A associação com a serpente Píton ocorreu por sugestão da editora O'Reilly, famosa por sempre colocar retratos de animais silvestres nas capas de seus livros



Características do Python

- Há um fundamento filosófico por trás da linguagem Python
- Semelhança com o Manifesto Ágil?

http://python-history-pt-br.blogspot.com/2009/04/filosofia-do-python.html

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Características do Python

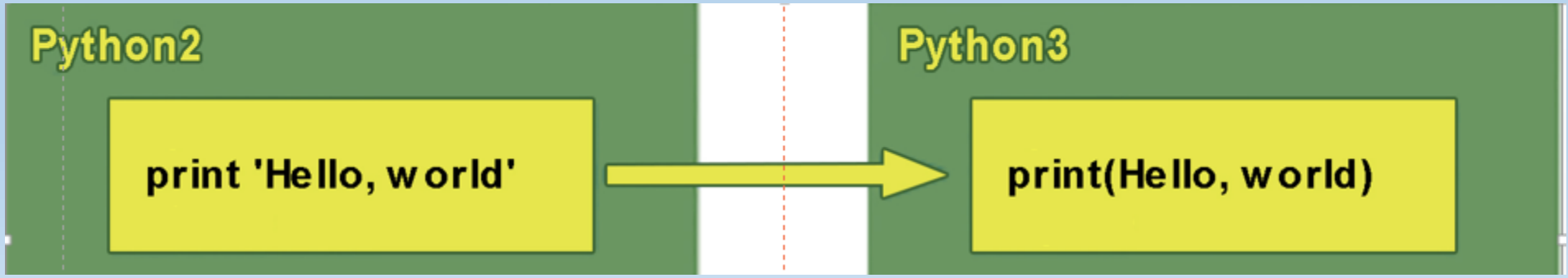
- Há um fundamento filosófico por trás da linguagem Python
- Semelhança com o Manifesto Ágil?

http://python-history-pt-br.blogspot.com/2009/04/filosofia-do-python.html

Bonito é melhor que feio.
Explícito é melhor que implícito.
Simples é melhor que complexo.
Complexo é melhor que complicado.
Plano é melhor que aninhado.
Esparso é melhor que denso.
Legibilidade conta.
Casos especiais não são especiais o bastante para se quebrar as regras.
Embora a simplicidade supere o purismo.
Erros nunca deveriam passar silenciosamente.
A menos que explicitamente silenciados.
Ao encarar a ambiguidade, recuse a tentação de adivinhar.
Deveria haver uma – e preferencialmente apenas uma – maneira óbvia de se fazer isto.
Embora aquela maneira possa não ser óbvia à primeira vista se você não for holandês.
Agora é melhor que nunca.
Embora nunca, seja muitas vezes melhor que pra já.
Se a implementação é difícil de explicar, é uma má ideia.
Se a implementação é fácil de explicar, pode ser uma boa ideia.
Namespaces são uma idéia estupenda – vamos fazer mais deles!

Características do Python

- Há significativas diferenças e incompatibilidades entre Python 2 e Python 3
  - print ‘ ‘ e print ()
  - True e False
  - …



Características do Python

- É uma linguagem de alto nível e legibilidade (sintaxe elegante)

```
#include <iostream>
int main() {
    std::cout << "Hello, world!";
}
```

C++

```
using System;
namespace HelloWorld {
    class Hello {
        static void Main() {
            Console.WriteLine( "Hello, world!" );
            Console.ReadKey();
        }
    }
}
```

C#

```
#import <Foundation/Foundation.h>
int main ( int argc, const char * argv[] ) {
    NSAutoreleasePool *pool = [[NSAutoreleasePool alloc] init];
    NSLog (@"Hello, world!");
    [pool drain];
    return 0;
}
```

Objective-C

```
class Hello {
    public static void main( String[] args ) {
        System.out.println( "Hello, world!" );
    }
}
```

Java

```
>>> print ("Hello, world")
```

Python

Características do Python

- É uma linguagem de alto nível e legibilidade (sintaxe elegante)
  - Determinar se um elemento está presente em uma lista

```
>>> 3 in [1, 2, 3, 4, 5]
True
```

  - Verificar se uma substring está em uma string:

```
>>> 'sub' in 'string'
False
```

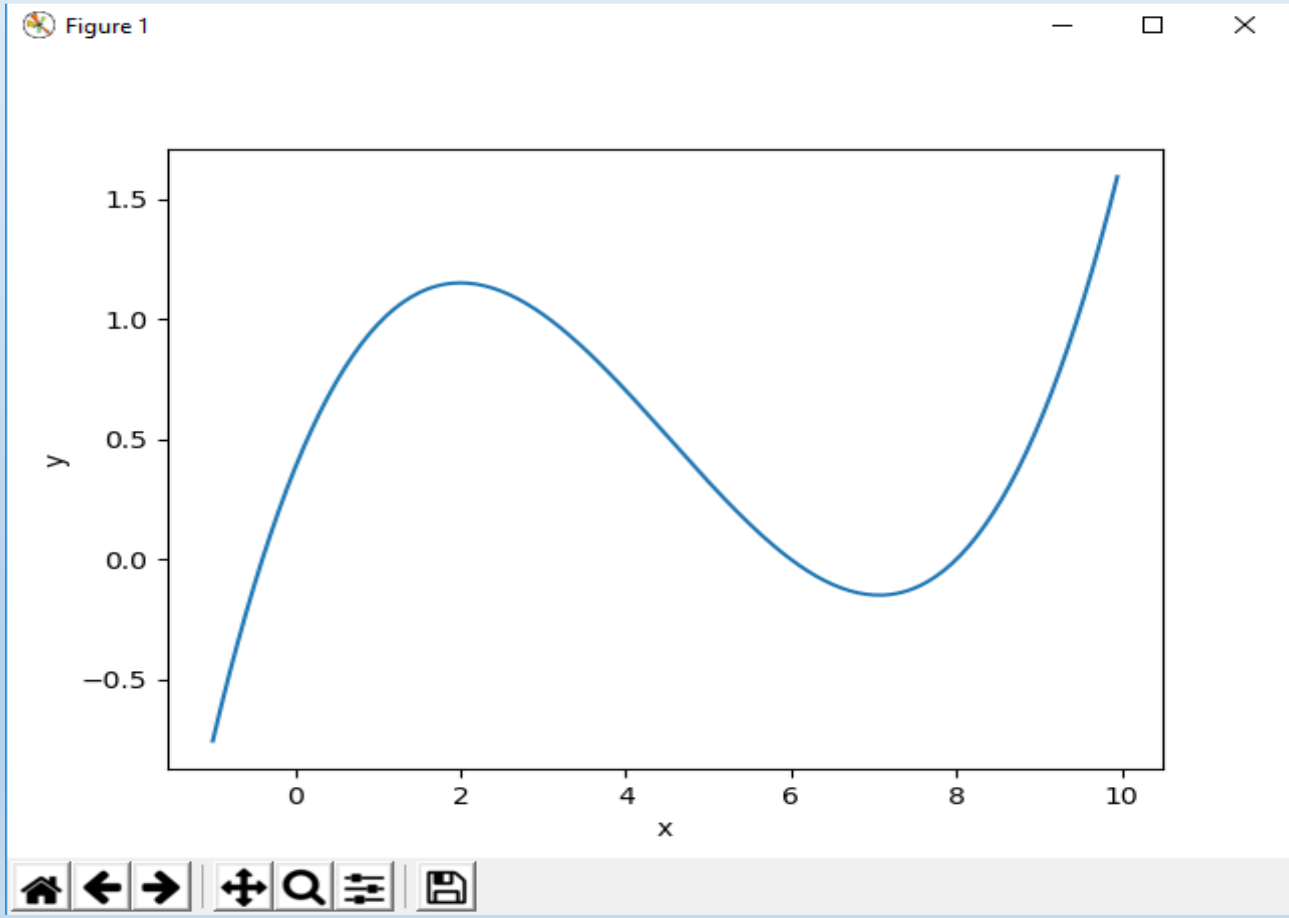
  - Booleanos e operadores lógicos:

```
>>> a = True
>>> not a
False
>>> 'sub' not in ['string', 'hello', 'world']
True
>>> a or True and not 1==2
True
```

Características do Python

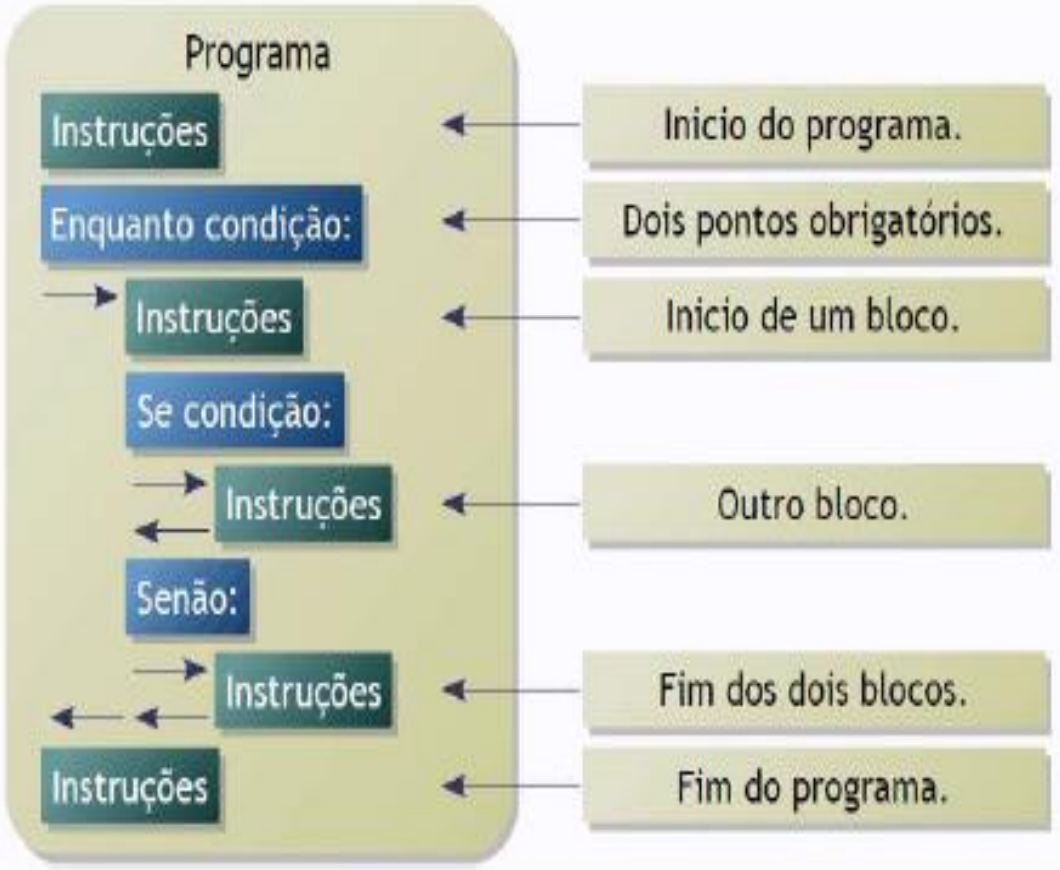
- É uma linguagem de alto nível e legibilidade (sintaxe elegante)
  - Plotar um gráfico

```
import numpy as np
import matplotlib as mpl
mpl.use('TkAgg')
import matplotlib.pyplot as plt

x = np.arange(-1, 10, 0.05)
y = 0.02*(x-8)*(x-6)*(x+0.4)
plt.plot(x, y)
plt.ylabel('y')
plt.xlabel('x')
plt.show()
```



Características do Python

- Indentação como delimitação de blocos

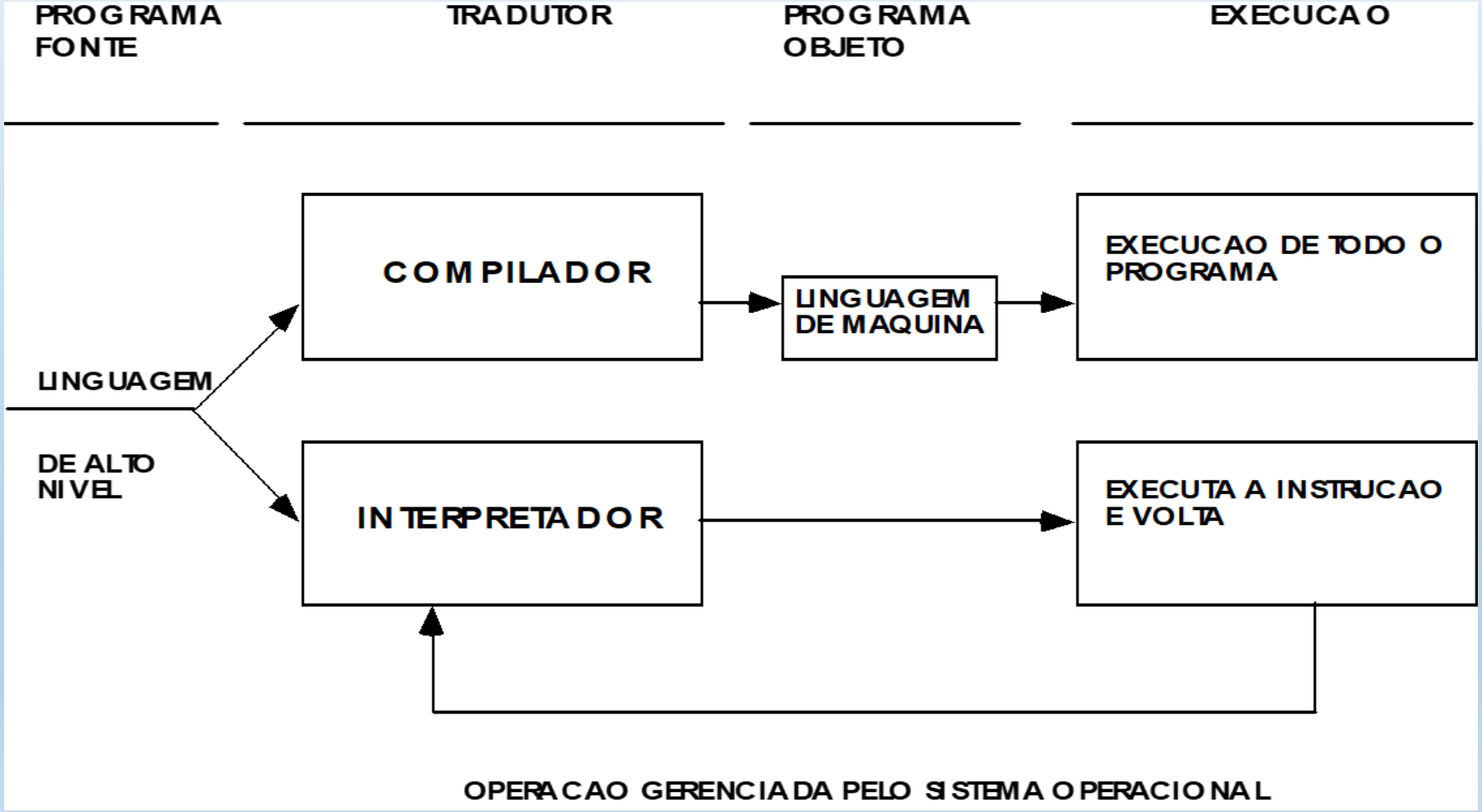



```
1  print("Imprimindo tabuada do 8")
2  for i in range(1,11):
3      j = i * 8
4      print(" 8 x ",i," = ",j)
5  print("Simples assim")
```

Início do bloco

Características do Python

- É uma linguagem interpretada



Características do Python

- Por ser interpretada, é uma linguagem lenta...

```
import os
from time import time
print ("Primeira implementação")
start_time = time( )
xvec = range(2000000)
yvec = range(2000000)
zvec = [0.5*(x+y) for x,y in zip(xvec,yvec)]
end_time = time( )
elapsed_time = end_time - start_time
print (elapsed_time)
```

Características do Python

- Por ser interpretada, é uma linguagem lenta...

```
import os
from numpy import *
from time import time
print ("Segunda implementação")
start_time = time( )
xvec = arange(2000000)
yvec = arange(2000000)
zvec = (xvec+yvec)*0.5
end_time = time( )
elapsed_time = end_time - start_time
print (elapsed_time)
```

Características do Python

- Por ser interpretada, é uma linguagem lenta...

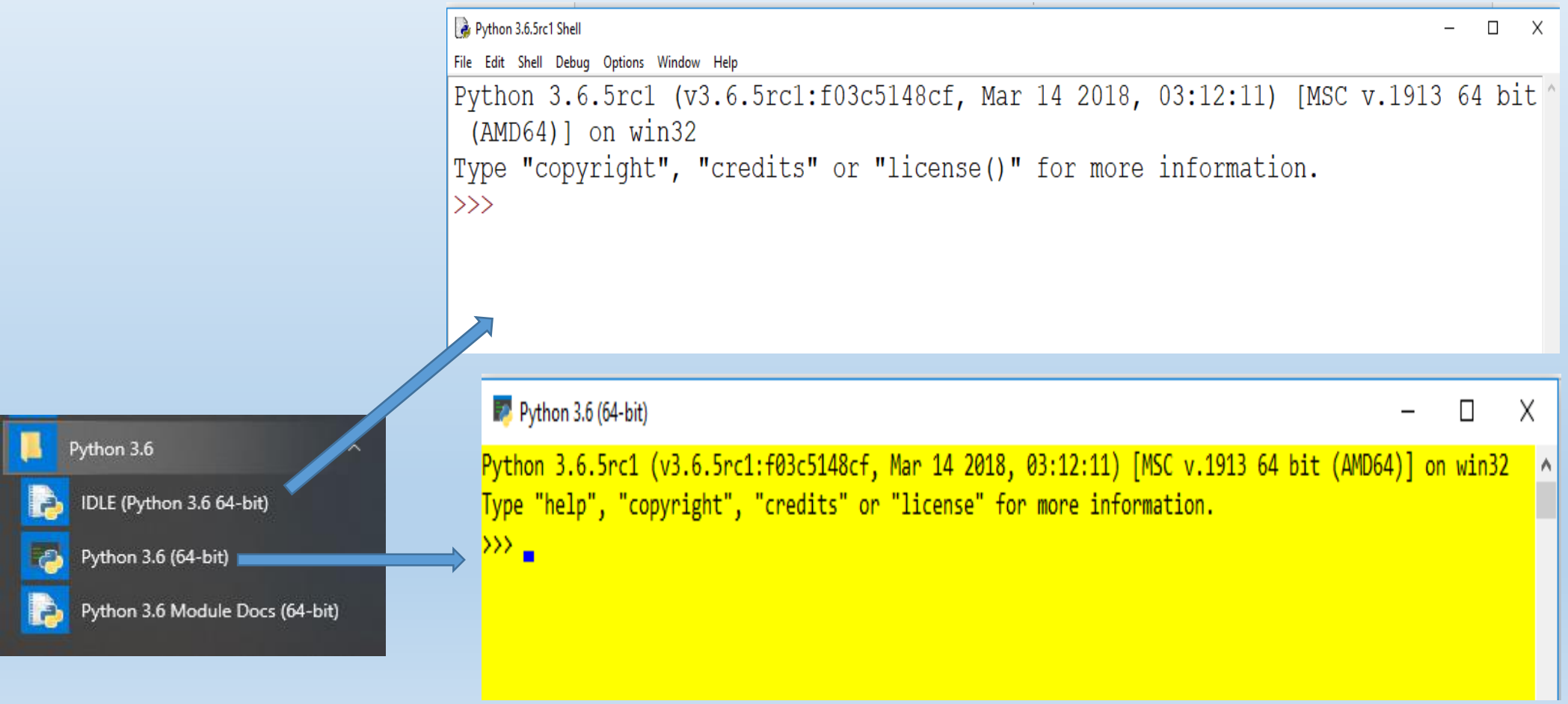
```
import os
from time import time
print ("Primeira implementação")
start_time = time( )
xvec = range(2000000)
yvec = range(2000000)
zvec = [0.5*(x+y) for x,y in zip(xvec,yvec)]
end_time = time( )
elapsed_time = end_time - start_time
print (elapsed_time)
```

```
import os
from numpy import *
from time import time
print ("Segunda implementação")
start_time = time( )
xvec = arange(2000000)
yvec = arange(2000000)
zvec = (xvec+yvec)*0.5
end_time = time( )
elapsed_time = end_time - start_time
print (elapsed_time)
```

```
D:\Arq-Python>python Perform1.py
Primeira implementação
0.47469544410705566

D:\Arq-Python>python Perform2.py
Segunda implementação
0.019945621490478516

D:\Arq-Python>python Perform1.py
Primeira implementação
0.46774911880493164

D:\Arq-Python>python Perform2.py
Segunda implementação
0.017912864685058594
```

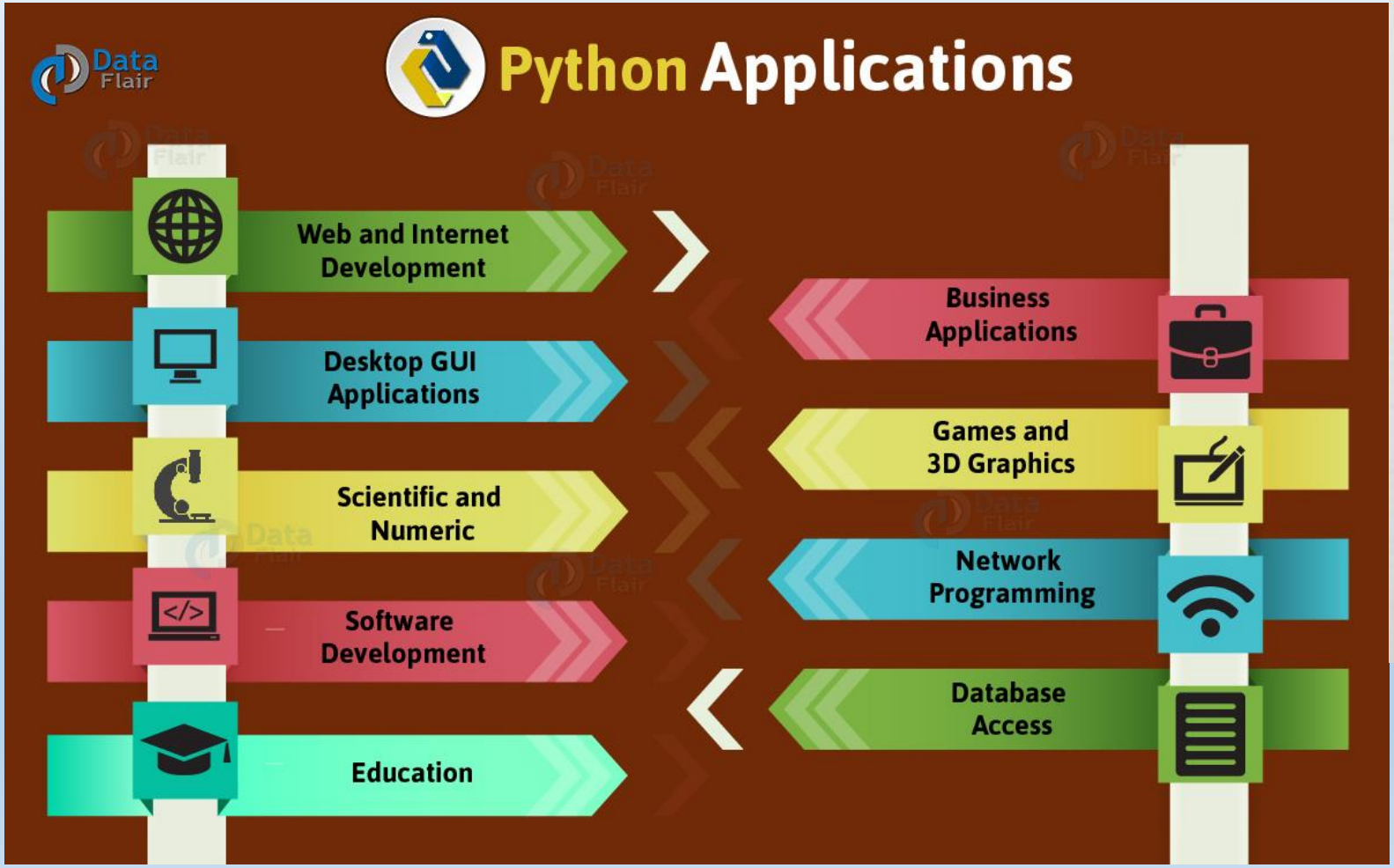
Características do Python

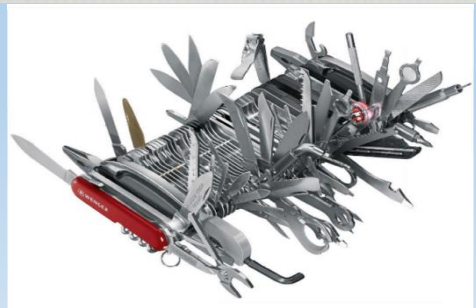
- Permite a execução de programas de forma interativa



Características do Python

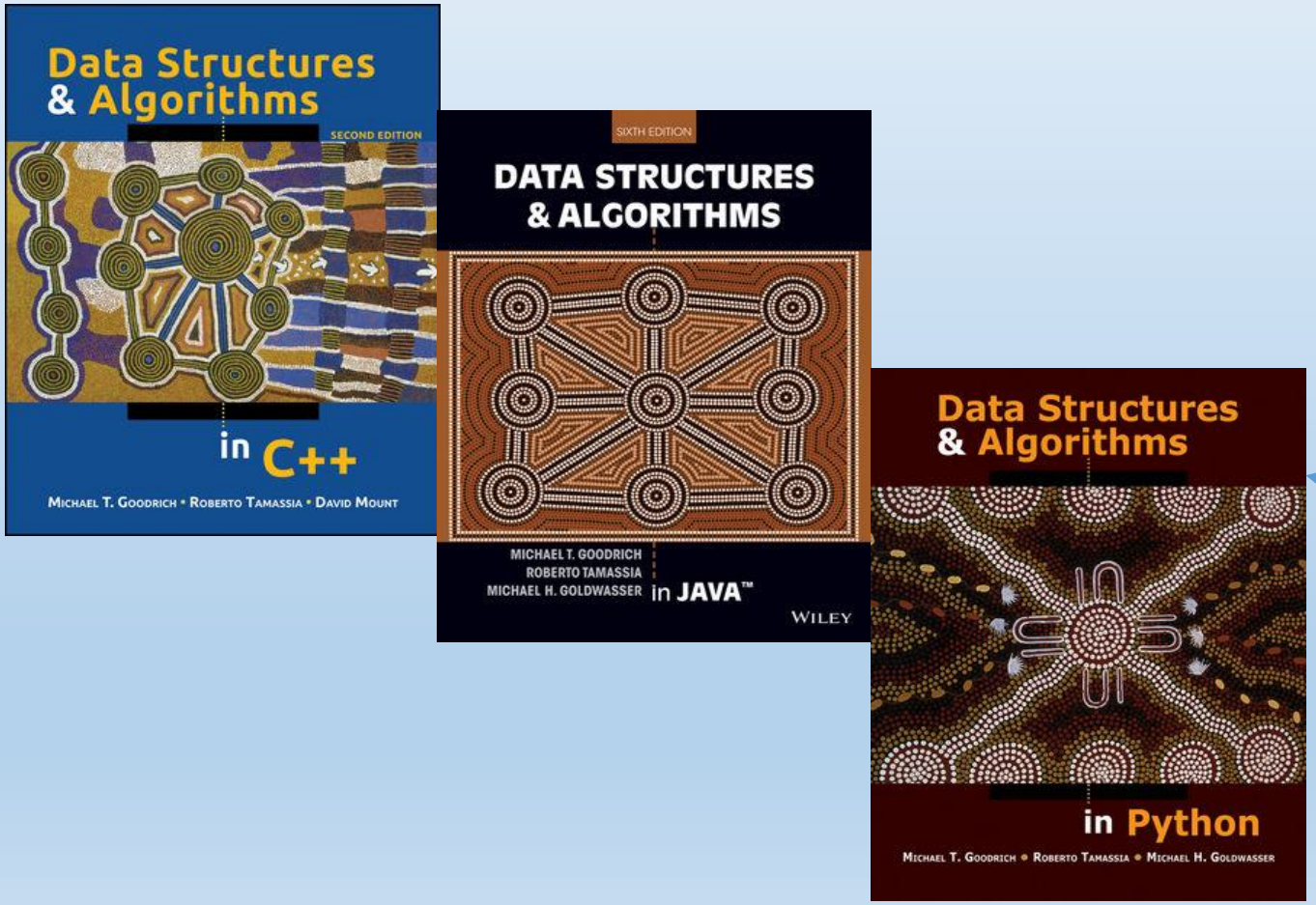
- É uma linguagem de propósito geral





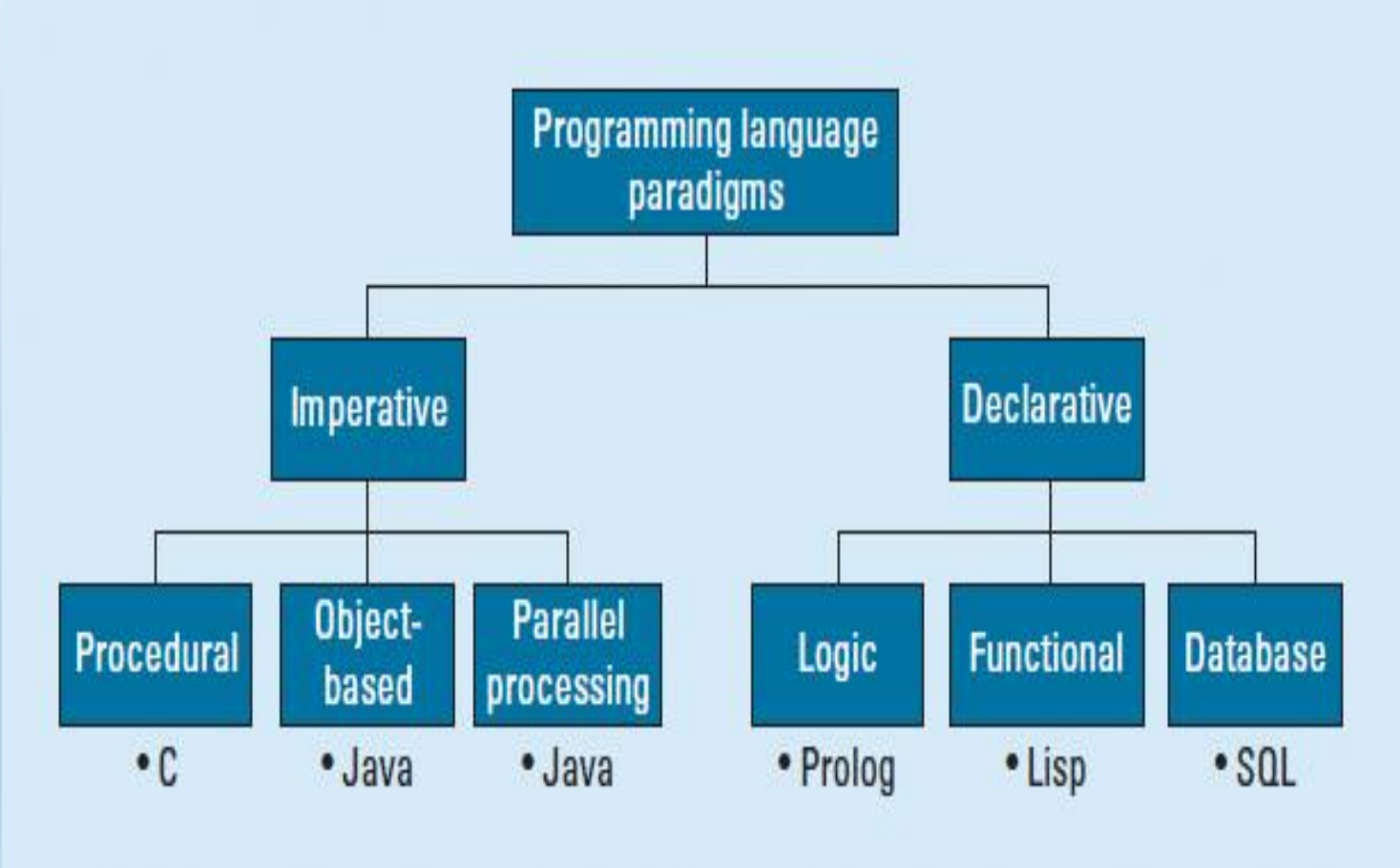
Características do Python

- Vem sendo bem difundida nas universidades



Características do Python

- Paradigmas de Programação
  - Procedural
  - Orientada a Objetos
  - Funcional



Características do Python

- Possui tipagem dinâmica e forte

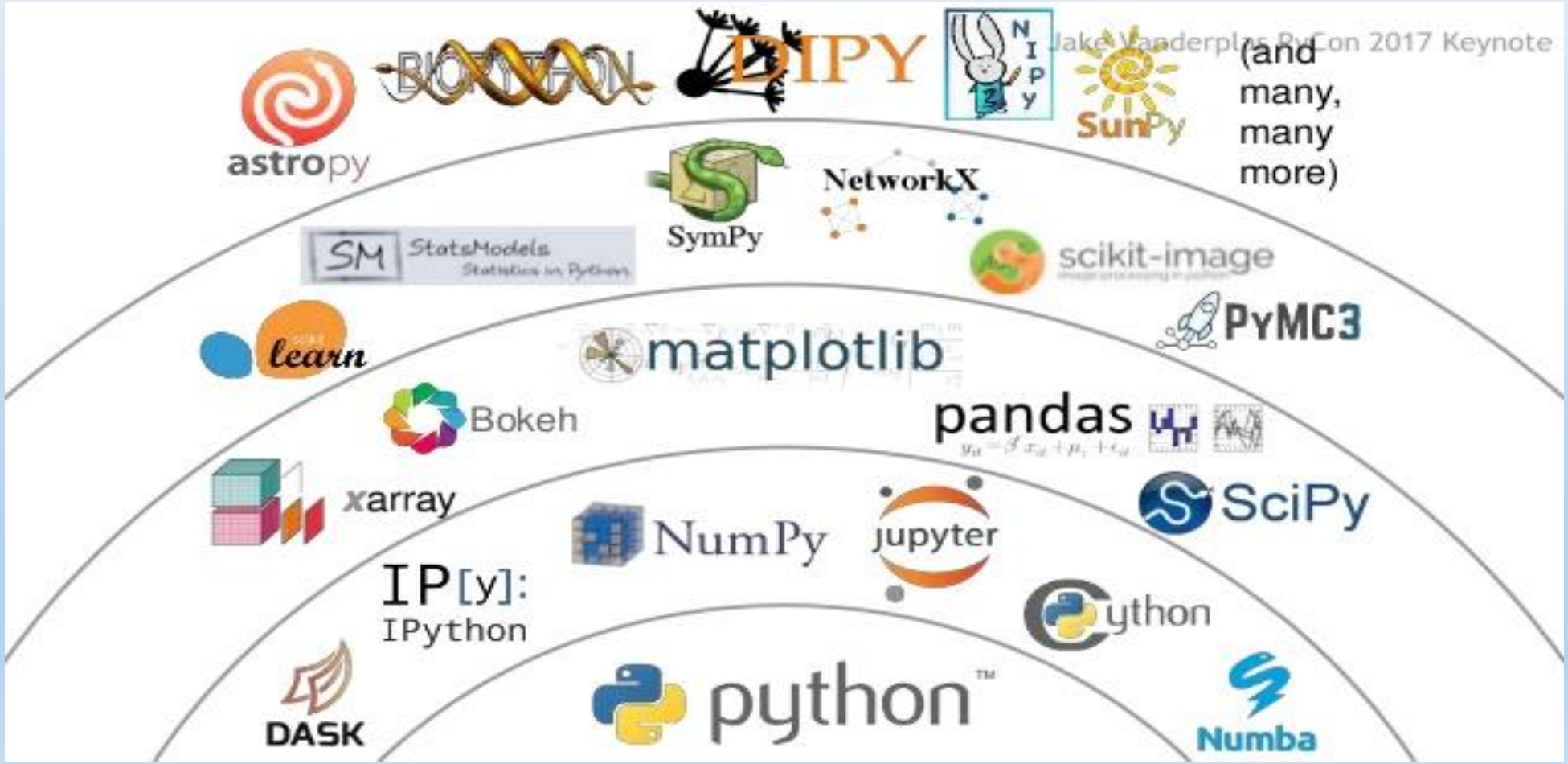
```
>>> a = 5       # Tipagem Dinâmica
>>> b = "6"
>>> c = a + b      # Tipagem Forte
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Tipagem Dinâmica:
O tipo da variável é definido em tempo de execução (*run time*)

Tipagem Forte:
Uma vez que a variável tenha um tipo definido, ela não pode ser usado como se fosse de outro tipo

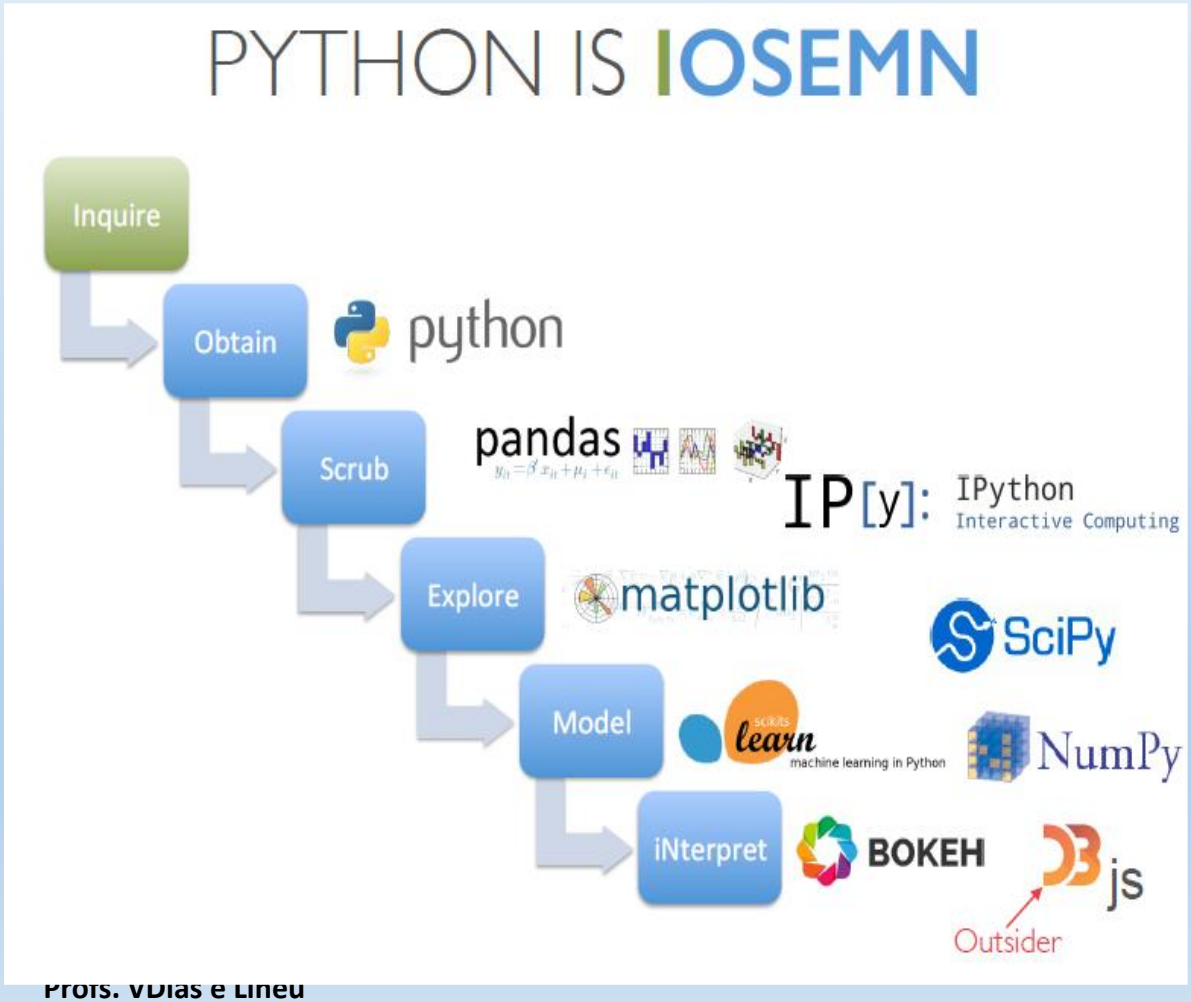
Características do Python

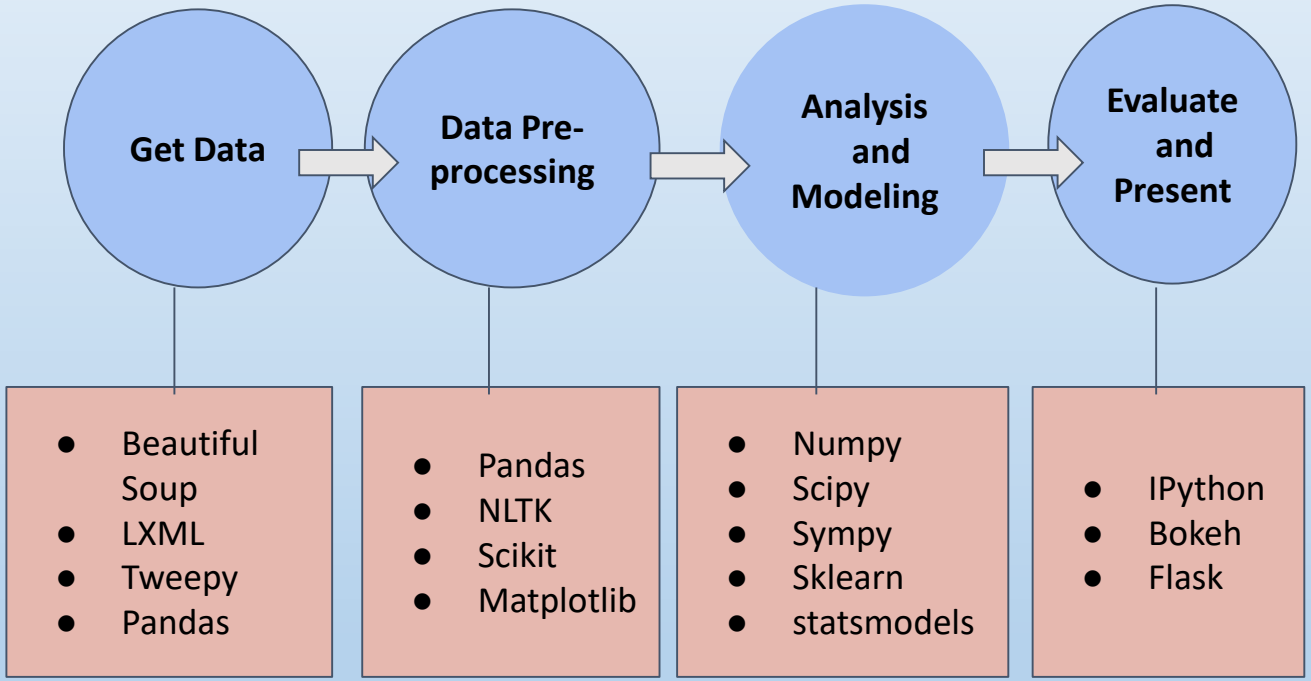
- Contém diversas coleções de bibliotecas



Características do Python

- Contém diversas coleções de bibliotecas





Características do Python

- Código aberto (GPL)

Python é distribuída sob uma licença própria (compatível com a GPL), que impõe poucas restrições. É permitida a distribuição, comercial ou não, tanto da linguagem quanto de aplicações desenvolvidas nela, em formato binário ou código fonte, bastando cumprir a exigência de manter o aviso de Copyright da PSF (*Python Software Foundation*). Veja a licença em mais detalhes aqui: Licença do Python 2.x ou Licença do Python 3.x.

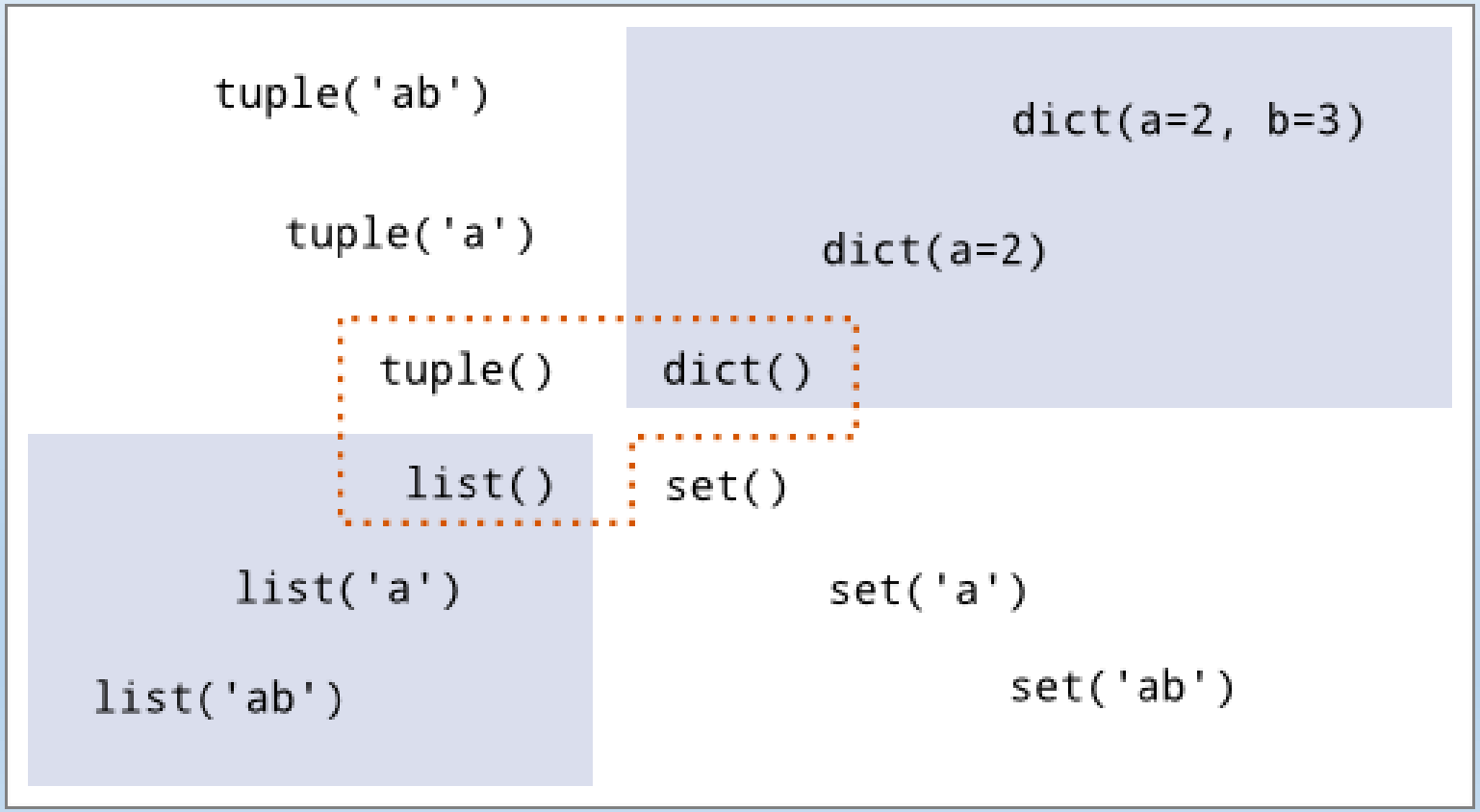
https://wiki.python.org.br/PerguntasFrequentes/SobrePython

All Python releases are Open Source (see https://opensource.org/ for the Open Source Definition). Historically, most, but not all, Python releases have also been GPL-compatible; the table below summarizes the various releases.

https://docs.python.org/3/license.html

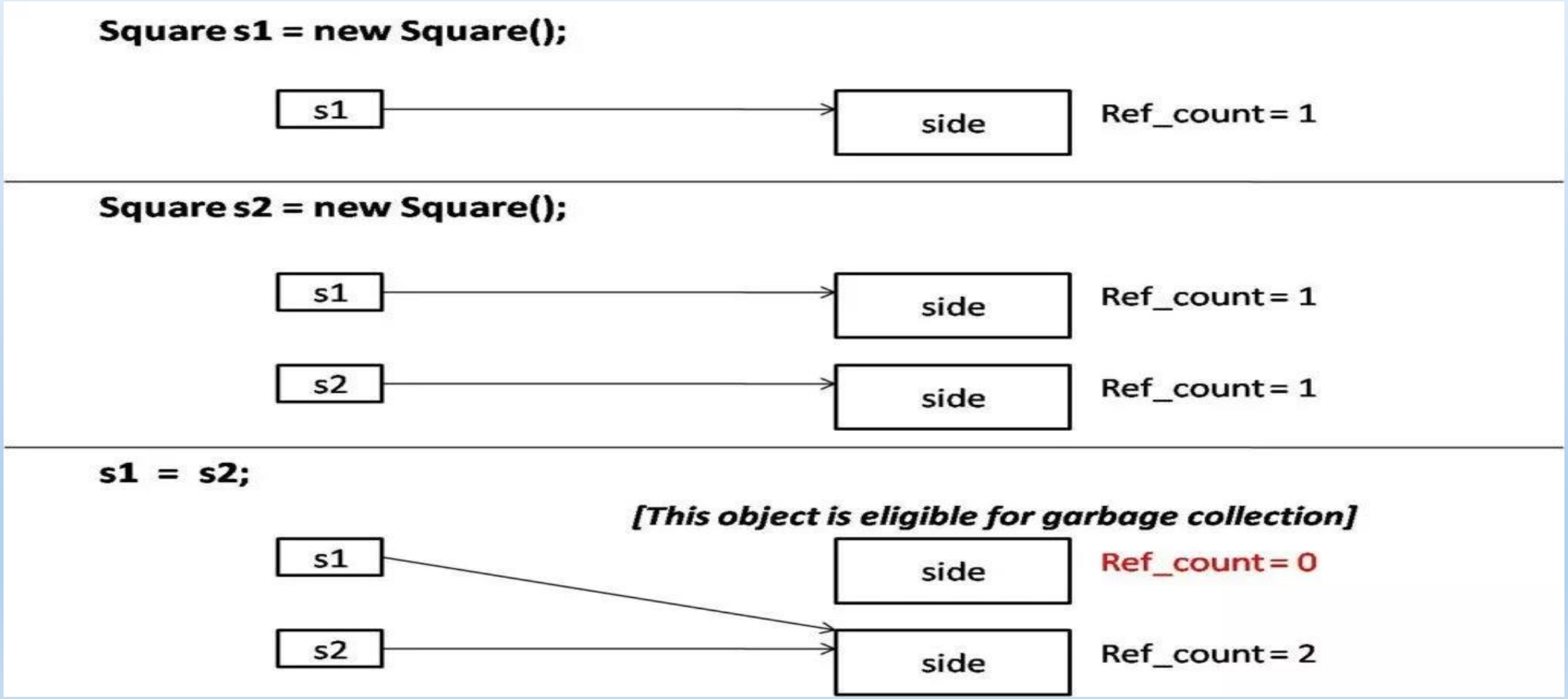
Características do Python

- Possui diversas estruturas de dados nativas: Lista, Tupla, Dicionário, Conjunto



Características do Python

- Tem gerenciamento de memória automático (*garbage collection*)



Características do Python

- Possui mecanismo para tratamento de exceções

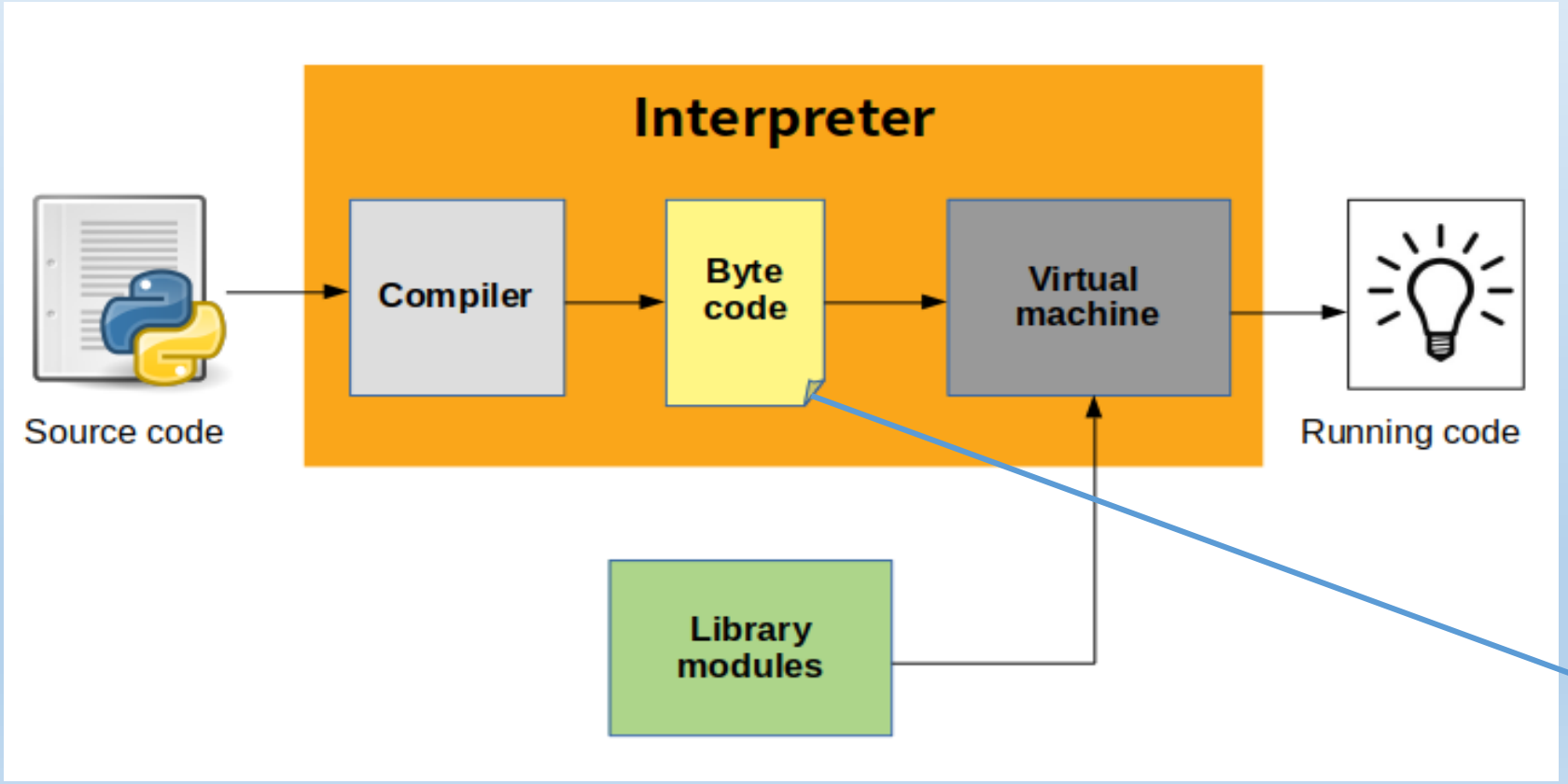
Características do Python

- Permite sobrecarga de operadores

```
>>> # Adds the two numbers
>>> 1 + 2
3

>>> # Concatenates the two strings
>>> 'Real' + 'Python'
'RealPython'


>>> # Gives the product
>>> 3 * 2
6

>>> # Repeats the string
>>> 'Python' * 3
'PythonPythonPython'
```

Características do Python

- Tem portabilidade (multiplataforma) e interoperabilidade
  - Executa em Windows, Mac e Linux, entre outras plataformas



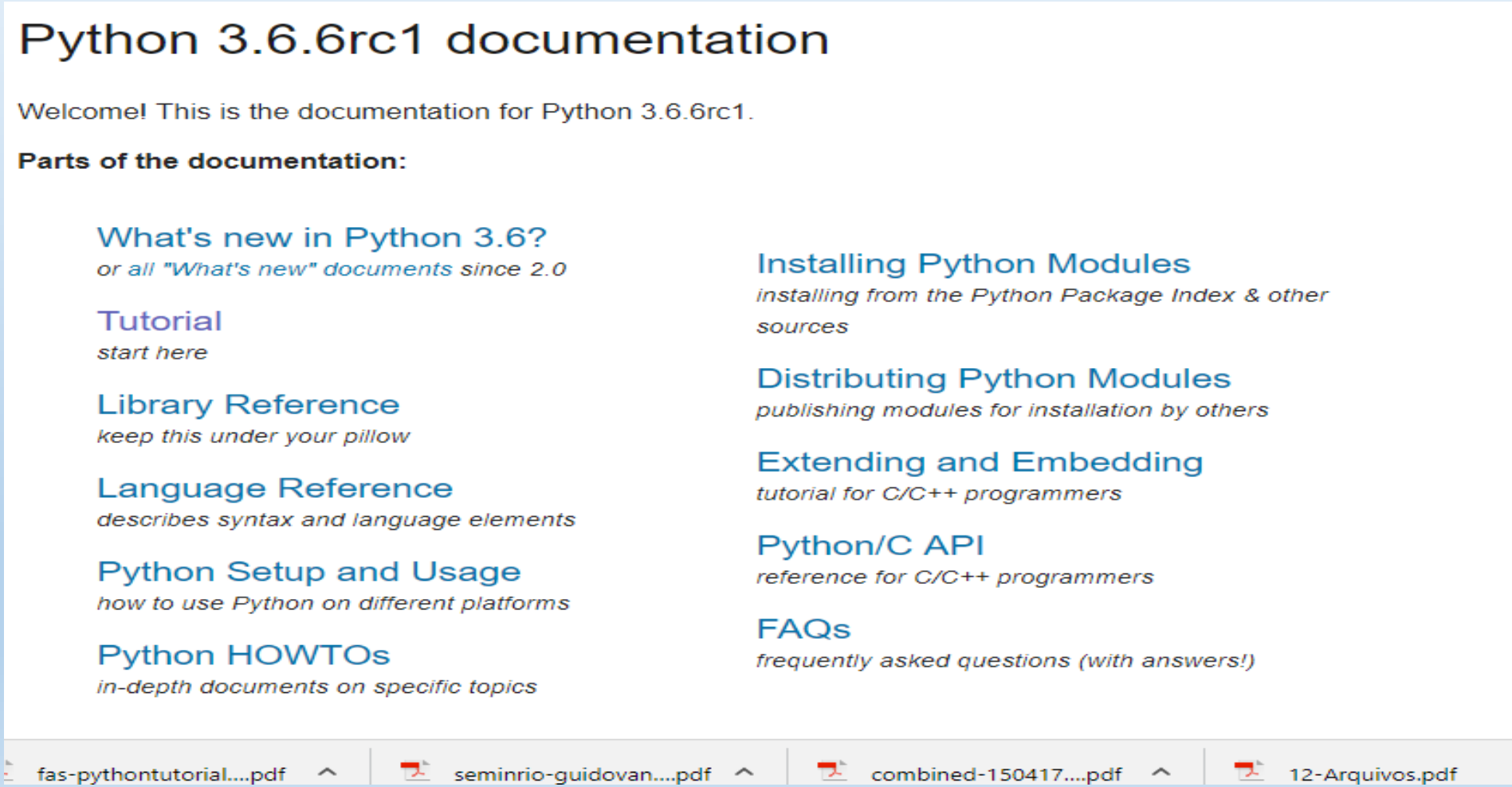
- Jython
- CPython
- IronPython
- PyPy
- PyObjC (Mac OSX middleware)
- Python for Delphi
- Brython

Características do Python

- Tem uma grande comunidade de desenvolvedores

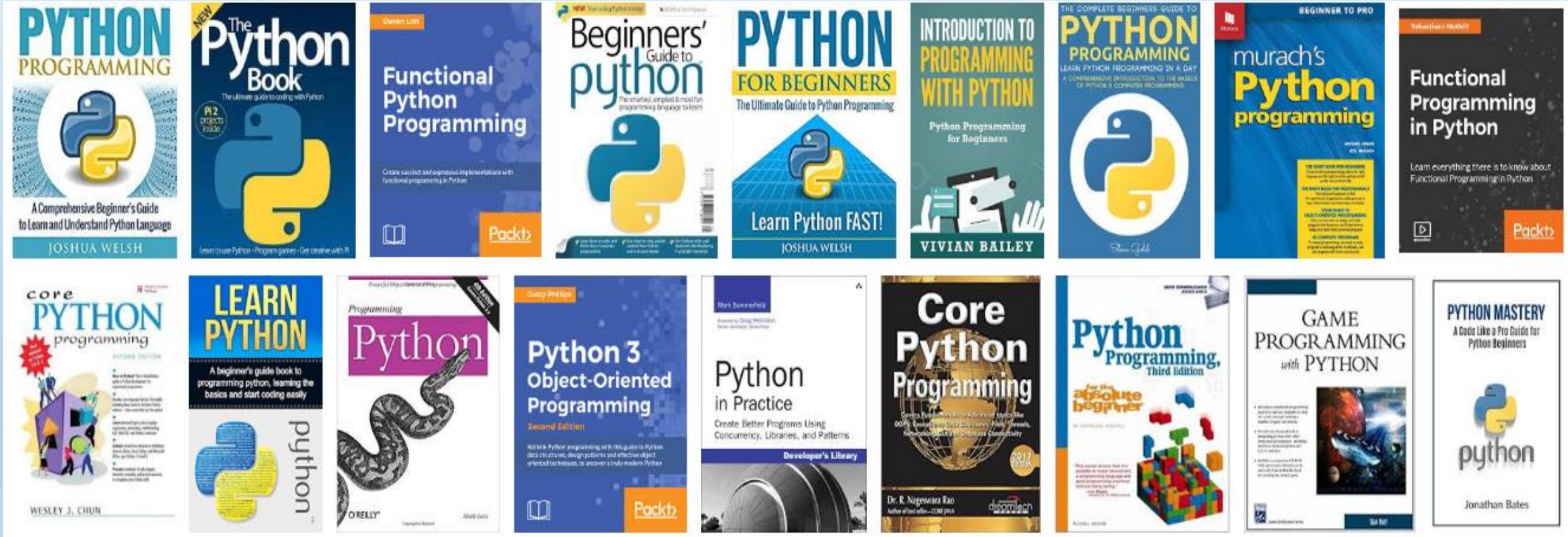
Características do Python

- Disponibiliza uma documentação adequada

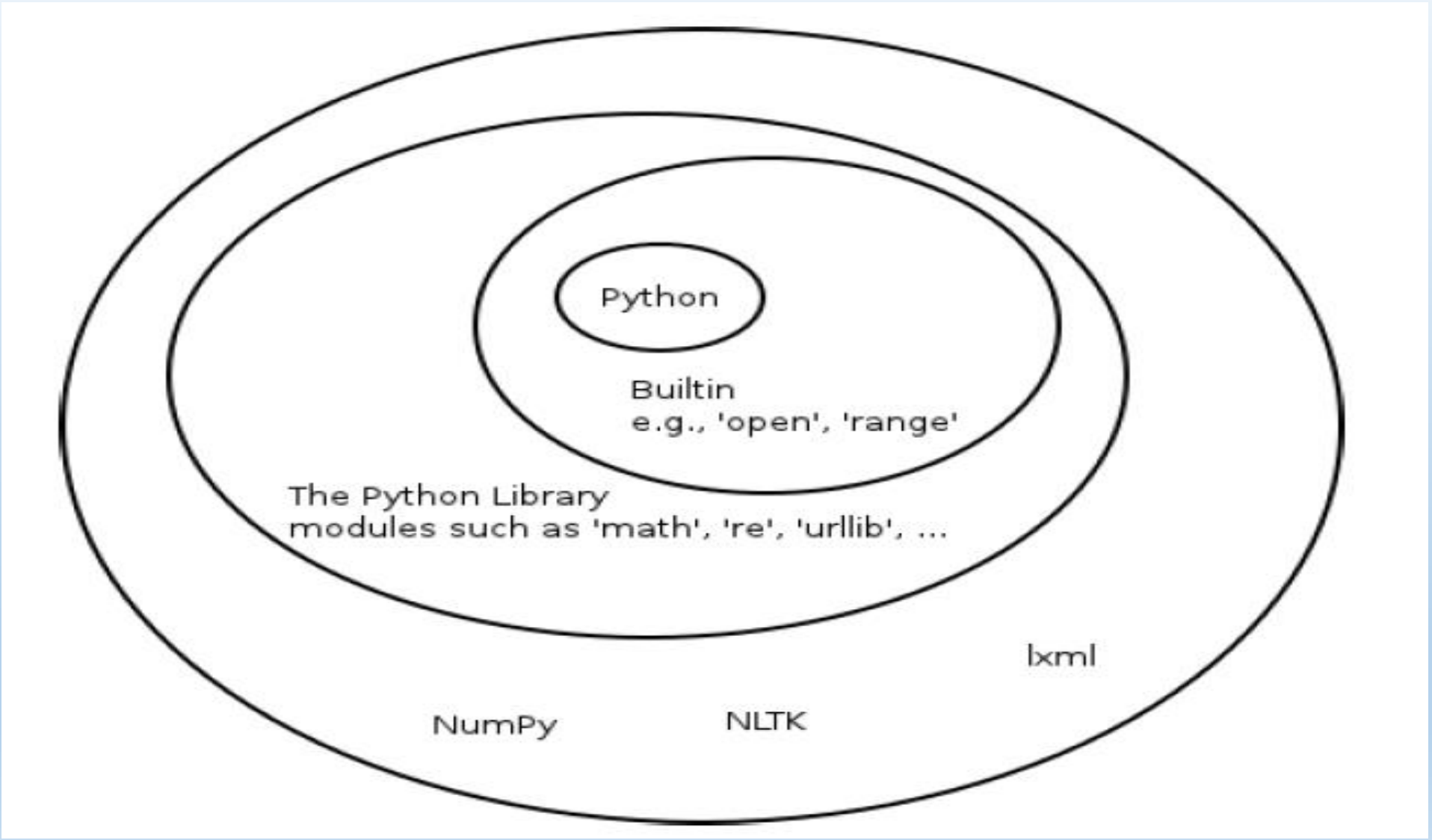


Características do Python

- Possui razoável bibliografia

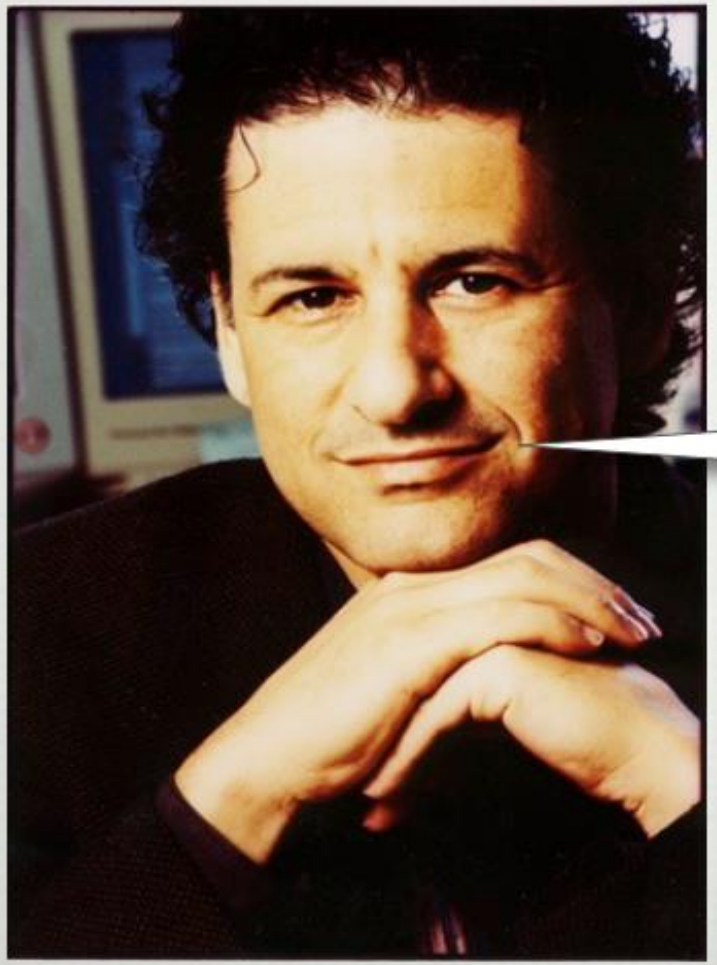


Características do Python



Características do Python

Daniel Levitin,
neuroscientist



In study after study, of composers, basketball players, fiction writers, ice-skaters, concert pianists, chess players, master criminals... this number comes up again and again. Ten thousand hours is equivalent to roughly three hours a day, or 20 hours a week, of practice over 10 years... No one has yet found a case in which true world-class expertise was accomplished in less time. It seems that it takes the brain this long to assimilate all that it needs to know to achieve true mastery.

10,000

Características do Python

- Quem usa Phyton?

Características do Python

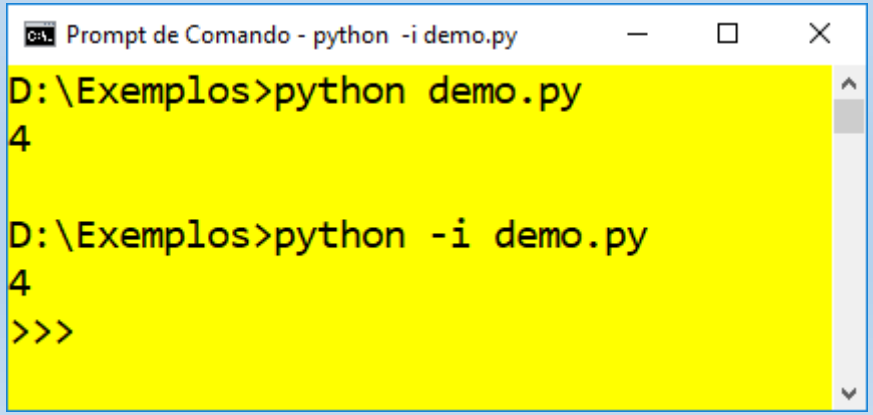
- Quem usa Phyton?

Interpretador Python

- Python é uma linguagem interpretada
  - Suas instruções são executadas pelo interpretador Python, que recebe um comando, avalia-o e reporta o resultado
- O interpretador pode ser usado interativamente

```
PyDev console: starting.

Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018,
16:22:17) [MSC v.1500 32 bit (Intel)] on win32
>>> 2 + 3
5
```

- Ou usando-se um arquivo com código fonte



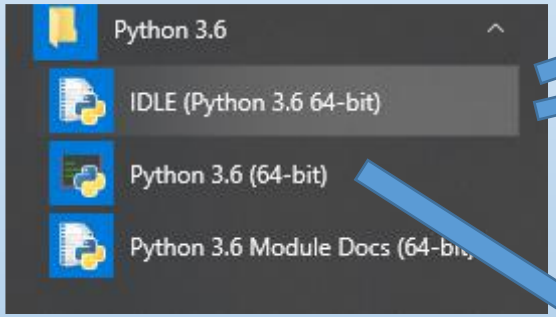
Integrated Development Environments - IDEs

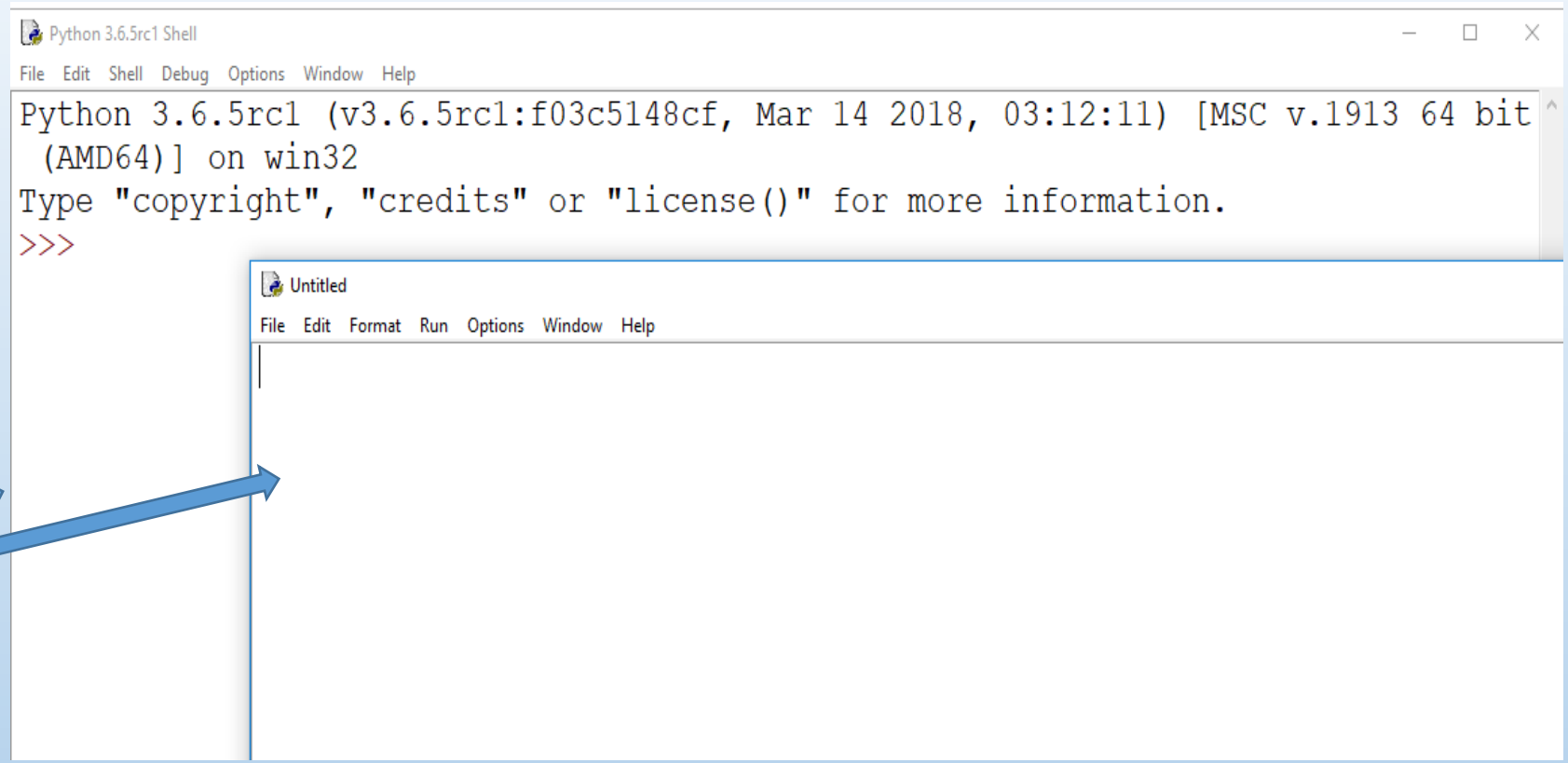
- Muitos Ambientes Integrados de Desenvolvimento (*Integrated Development Environments* – IDEs) disponibilizam plataformas de softwares para desenvolvimento de aplicações em Python

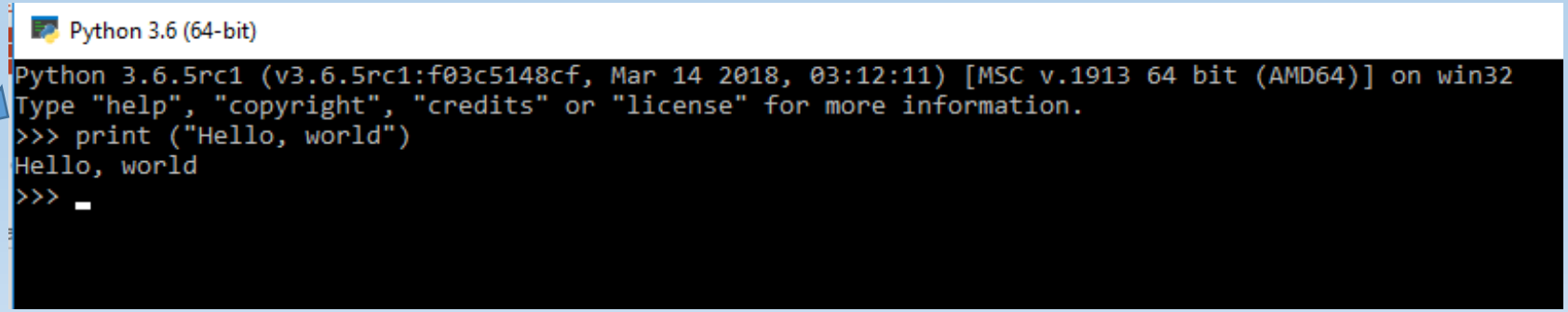
IDLE

- IDLE is Python's Integrated Development and Learning Environment
- IDLE has the following features:
  - coded in 100% pure Python, using the tkinter GUI toolkit
  - cross-platform, works mostly the same on Windows, Unix, and Mac OS X
  - Python shell window (interactive interpreter) with colorizing of code input, output, and error messages
  - multi-window text editor with multiple undo, Python colorizing, smart indent, call tips, auto completion, and other features
  - search within any window, replace within editor windows, and search through multiple files (grep)
  - debugger with persistent breakpoints, stepping, and viewing of global and local namespaces configuration, browsers, and other dialogs

IDLE

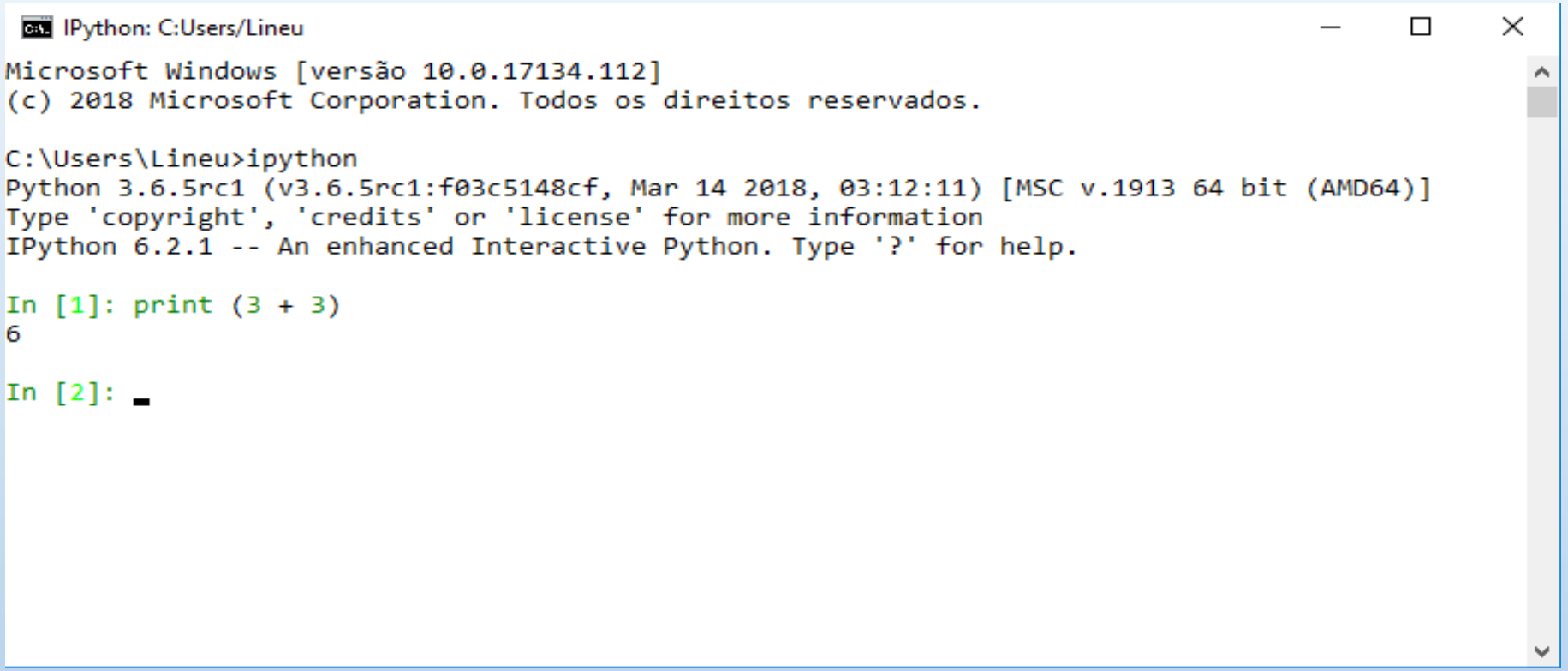

Python 3.6
IDLE (Python 3.6 64-bit)
Python 3.6 (64-bit)
Python 3.6 Module Docs (64-bi


Python 3.6.5rc1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.5rc1 (v3.6.5rc1:f03c5148cf, Mar 14 2018, 03:12:11) [MSC v.1913 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
Untitled
File Edit Format Run Options Window Help


Python 3.6 (64-bit)
Python 3.6.5rc1 (v3.6.5rc1:f03c5148cf, Mar 14 2018, 03:12:11) [MSC v.1913 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print ("Hello, world")
Hello, world
>>>

IPython

- Outro shell para o Python



IPython is an interactive shell that addresses the limitation of the standard python interpreter, and it is a work-horse for scientific use of python. It provides an interactive prompt to the python interpreter with a greatly improved user-friendliness.

Visão Prévia de um Programa Python

```
print('Welcome to the GPA calculator.')
print('Please enter all your letter grades, one per line.')
print('Enter a blank line to designate the end.')
# map from letter grade to point value
points = {'A+':4.0, 'A':4.0, 'A-':3.67, 'B+':3.33, 'B':3.0, 'B-':2.67,
          'C+':2.33, 'C':2.0, 'C':1.67, 'D+':1.33, 'D':1.0, 'F':0.0}
num_courses = 0
total_points = 0
done = False
while not done:
  grade = input( )                                  # read line from user
  if grade == '':                                   # empty line was entered
    done = True
  elif grade not in points:                         # unrecognized grade entered
    print("Unknown grade '{0}' being ignored".format(grade))
  else:
    num_courses += 1
    total_points += points[grade]
if num_courses > 0:                                 # avoid division by zero
  print('Your GPA is {0:.3}'.format(total_points / num_courses))
```

Visão Prévia de um Programa Python

```
print('Welcome to the GPA calculator.')
print('Please enter all your letter grades, one per line.')
print('Enter a blank line to designate the end.')
# map from letter grade to point value
points = {'A+':4.0, 'A':4.0, 'A-':3.67, 'B+':3.33, 'B':3.0, 'B-':2.67,
          'C+':2.33, 'C':2.0, 'C':1.67, 'D+':1.33, 'D':1.0, 'F':0.0}
num_courses = 0
total_points = 0
done = False
while not done:
  grade = input( )                          # read line from user
  if grade == '':                           # empty line was entered
    done = True
  elif grade not in points:                 # unrecognized grade entered
    print("Unknown grade '{0}' being ignored".format(grade))
  else:
    num_courses += 1
    total_points += points[grade]
if num_courses > 0:                         # avoid division by zero
  print('Your GPA is {0:.3}'.format(total_points / num_courses))
```

Instruções são concluídas com enter (*return followed by linefeed*)

Elas podem ser estendidas de uma linha para outra abrindo um delimitador que não tenha sido fechado

Ou podem ser estendidas também com a barra invertida \ (*backslash caracter*)

Visão Prévia de um Programa Python

Comentários são implementados por meio do caractere #

Espaço em branco (*whitespace*) é importante na delimitação do corpo de estruturas de controle

```
print('Welcome to the GPA calculator.')
print('Please enter all your letter grades, one per line.')
print('Enter a blank line to designate the end.')
# map from letter grade to point value
points = {'A+':4.0, 'A':4.0, 'A-':3.67, 'B+':3.33, 'B':3.0, 'B-':2.67,
          'C+':2.33, 'C':2.0, 'C':1.67, 'D+':1.33, 'D':1.0, 'F':0.0}
num_courses = 0
total_points = 0
done = False
while not done:
  grade = input( )                          # read line from user
  if grade == '':                           # empty line was entered
    done = True
  elif grade not in points:                 # unrecognized grade entered
    print("Unknown grade '{0}' being ignored".format(grade))
  else:
    num_courses += 1
    total_points += points[grade]
if num_courses > 0:                         # avoid division by zero
  print('Your GPA is {0:.3}'.format(total_points / num_courses))
```

Exercício # 3– Entrega 06/07

- Pesquisar na Internet:
  - Quais são as 10 linguagens de programação mais populares
  - Quais as linguagens com maior empregabilidade
  - Quais as linguagens que pagam os maiores salários
  - Quais são as principais linguagens utilizadas em Data Science