

Mastering Python

الدرس # 1_1 مقدمة للغة بايثون

By:

Hussam Hourani

V1.0 - NOV 2019

المحتويات



- ما هي لغة بايثون
- لماذا بايثون
- كيف تقارن بايثون مع اللغات الأخرى؟
- الإعداد والتثبيت - Setup
- تركيب مكتبات بايثون
- كيف يبدو سبايدر Spyder؟
- تشغيل أول برنامج بايثون الخاص بك
- بايثون شل / وحدة التحكم

ما هي لغة بايثون؟

- بايثون هي لغة برمجة، كما هي C ++، C، جافا، # C، الخ
- بايثون هي لغة الحوسبة العامة والحديثة.
- بايثون هي لغة Interpreted
- تم اشتقاق بايثون من العديد من اللغات الأخرى، بما في ذلك: C ++، C، Modula-3، Algol-68، وغيرها من لغات البرمجة.
- يستخدم العلماء والمهندسين بايثون لتحليل البيانات وحل مشاكل العلوم وتحليل البيانات الخاصة بهم.
- وقد اطلق Guido van Rossum أول إصدار بايثون 1989.
- كان Guido يقرأ النصوص المنشورة من "Monty Python's Flying Circus"، وهي سلسلة بي بي سي كوميدى من 1970. و قرر تسمية اللغة ببايثون لإعطائها بعض الغموض و وليكون الاسم مميز.
- وقد تم تنفيذ مشاريع كبيرة وكثيرة في بايثون (راجع: <https://www.python.org/about/success> للحصول على قائمة من المشاريع)

لماذا بايثون



- من السهل جدا أن تتعلمها ،وسهلة التعديل على البرمجيات ، و هي لغة قابلة جدا للقراءة.
- لغة برمجة تفاعلية ، و تقدم اطار البرمجة نحة الاشياء OOP.
- ذات مصدر مفتوح Open Source - بدون تكلفة (تحت رخصة مفتوحة المصدر).
- لغة مثالية لحل المشاكل العددية والتحليلية.
- منصة متعددة - Multi-platform: بايثون متاحه لمعظم أنظمة التشغيل الرئيسية: ويندوز، لينكس / ماك وس الخ...
- يمكنها التفاعل مع لغات أخرى مثل C و ++C وغيرها، ويمكن استخدامه كلغة البرمجة نصية - scripting language
- قابل للتمديد Extendable- ويمكنك إضافة مكتبات إلى بايثون
- لديها العديد من المكتبات المتخصصة مثل Numpy, Scipy, Matplotlib, Pandas والعديد من المكتبات الأخرى
- مجتمع بايثون "python community" ينمو بسرعة كبيرة.

قوة وسحر لغة بايثون

- بايثون هي لغة ذات مستوى عال، وكتبت بشكل حيوي، وغالبا ما يقال ان برامج بايثون تكون تقريبا مثل **پسيودوكود** **pseudocode**، لأنه يسمح للمبرمج للتعبير عن أفكار قوية جدا في عدد قليل جدا من التعليمات البرمجية في حين تكون مقروء جدا. كمثال:

```
def quicksort(arr):  
    if len(arr) <= 1: return arr  
    pivot = arr[len(arr) // 2]  
    left = [x for x in arr if x < pivot]  
    middle = [x for x in arr if x == pivot]  
    right = [x for x in arr if x > pivot]  
    return quicksort(left) + middle + quicksort(right)  
  
print quicksort([3,6,8,10,1,2,1])
```

Output

[1, 1, 2, 3, 6, 8, 10]

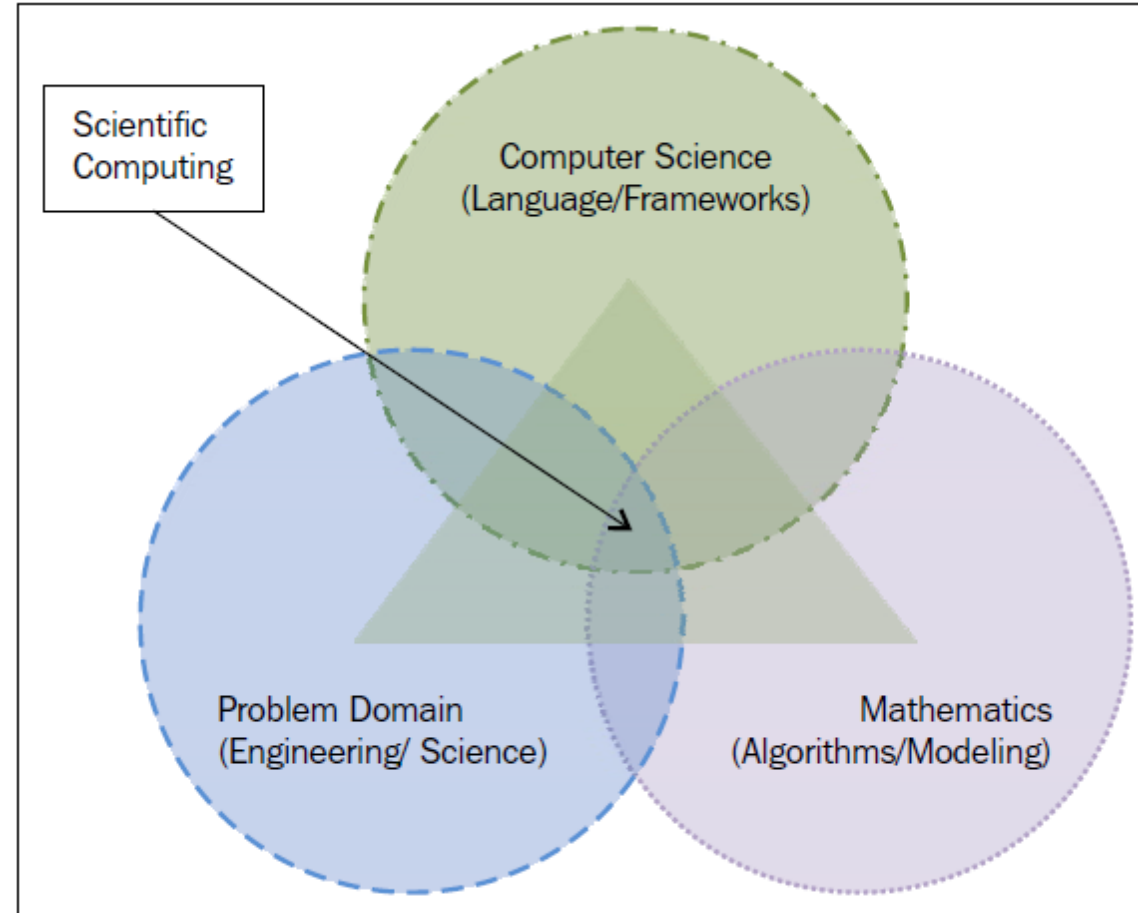
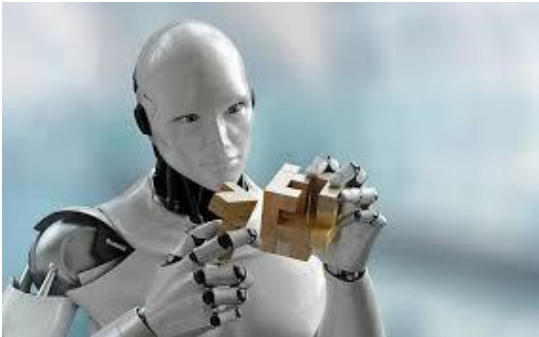


كيف تقارن بايثون مع اللغات الأخرى؟

- ان لغات مثل ++C، جافا وغيرها هي لغات صعبة لغير المبرمجين. في حين أن بايثون من السهل أن تتعلم حتى بالنسبة لغير المبرمجين.
- تعتبر بعض الحلول واللغات محدودة ويمكن أن تقيد المستخدمين المتقدمين. في حين بايثون غنية جدا وقابلة للتمديد extendable.
- تقتصر اللغات الأخرى على عدد خوارزميات محدودة، في حين بايثون لديها المكتبات الوظيفية العددية والغنية.
- بايثون لغة من السهل جدا تعلمها، وتحتوي على العديد من المكتبات تدعم الحوسبة العلمية (Scientific Computing) (Networks, Serial port access, Web Server) (Databases programming etc.)
- مجموعة متنوعة من بيئات التطوير المتكاملة القوية والتفاعلية (IDE) مثل :
 - IPython, Spyder, Jupyter notebooks, Pycharm وغيرها....



بايثون في الحوسبة العلمية - Scientific Computing



Scientific computing as an interdisciplinary field

Ref Book :Mastering Python Scientific Computing by : Hemant Kumar Mehta

التثبيت : Setup and installation

- Download latest version of Python from : <https://www.python.org>
- Install Python setup file in a specific Folder (Lets assume the folder : c:\Python\)

- Install PyQt5 () , by executing the commands

```
C:\python\scripts\pip3 install PyQt5
```

- Install Spyder and its dependencies by running this command:

```
C:\python\scripts\pip install spyder
```

- **Or** you can skip the above steps and directly download one of Python's IDE systems as following (they have all above but you need to upgrade the packages and Spyder) :
 - Anaconda (<https://www.anaconda.com/download/>)
 - WinPython (<https://winpython.github.io/>)

Python Packages installation

- We will use the most creative libraries in Python , so make sure you install/upgrade all the below packages by executing the below (before you start:

```
C:\python\scripts\pip install --upgrade Numpy Scipy Sympy skLearn Matplotlib Pandas Seaborn  
Statsmodels scikit-image openpyxl
```

- Or install each library as you require (before each related lesson) :

```
C:\python\scripts\pip install --upgrade Numpy  
C:\python\scripts\pip install -upgrade Scipy  
C:\python\scripts\pip install -upgrade Sympy  
C:\python\scripts\pip install -upgrade skLearn  
C:\python\scripts\pip install -upgrade matplotlib  
C:\python\scripts\pip install -upgrade Pandas  
C:\python\scripts\pip install -upgrade Seaborn  
C:\python\scripts\pip install -upgrade Statsmodels  
C:\python\scripts\pip install -upgrade scikit-image  
C:\python\scripts\pip install -upgrade openpyxl
```

How Spyder looks like?

Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - C:\Work\Python\Mystuff\Plot13.py

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 def mandelbrot( h,w, maxit=20 ):
5     """Returns an image of the Mandelbrot fractal"""
6     y,x = np.ogrid[ -1.4:1.4:h*1j, -2:0.8:w*1j]
7     c = x+y*1j
8     z = c
9     divtime = maxit + np.zeros(z.shape, dtype=int)
10
11     for i in range(maxit):
12         z = z**2 + c
13         diverge = z*np.conj(z) > 2**2
14         div_now = diverge & (divtime==maxit)
15         divtime[div_now] = i
16         z[diverge] = 2
17
18     return divtime
19
20 plt.imshow(mandelbrot(400,400))
21 plt.show()
```

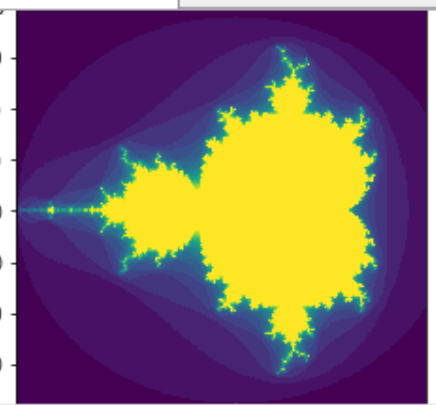
Variable explorer

Name	Type	Size	Value
X	float64	(40, 40)	array([[-2. , -1.9, -1.8, ..., 1.7, 1.8, ... [-2. , -1.9, ...
Y	float64	(40, 40)	array([[-2. , -2. , -2. , ..., -2. , -2. , -... [-1.9, -1.9, ...
a	int32	(3, 3)	matrix([[1, 2, 3], [4, 5, 6],
b	int32	(3, 3)	matrix([[4, 5, 6], [7, 8, 9],
mu	int	1	2
mx1	int32	(3, 3)	matrix([[1, 2, 3],

Variable explorer File explorer Help

IPython console

Console 1/A



IPython console History log

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 20 Column: 1 Memory: 73 %

Run your first Python Program

- Write the following program :

The screenshot shows the Spyder Python IDE interface. The main editor window displays a Python script named `Hello.py` with the following code:

```
1 st='Hello. This is my first program'
2 print (st)
```

Three green callout boxes with arrows point to specific parts of the interface:

- 1: Type the 2 lines** points to the code in the editor.
- 2: Run the code** points to the Run button (a green play icon) in the toolbar.
- 3: output** points to the output text in the IPython console.

The IPython console shows the execution of the code:

```
In [40]: runfile('C:/Python/My Stuff/Hello.py',
wdir='C:/Python/My Stuff')
Hello. This is my first program
```

The Variable explorer on the right shows the variable `st` of type `str` with the value `Hello. This is my first program`.

The status bar at the bottom indicates: Permissions: RW, End-of-lines: CRLF, Encoding: UTF-8, Line: 1, Column: 22, Memory: 72 %.

Python Shell / Console

The image shows a screenshot of an IPython console window. The window title is "IPython console". The interface includes a tab labeled "Console 1/A" and a scrollable text area containing the following input and output pairs:

```
In [132]: 1+1  
Out[132]: 2  
  
In [133]: a=10  
  
In [134]: b=20  
  
In [135]: a+b  
Out[135]: 30  
  
In [136]: 7 // 2  
Out[136]: 3  
  
In [137]: 7 % 2  
Out[137]: 1  
  
In [138]: 7.345667 * 8575342.3245 * 345 + 102927 / 2  
Out[138]: 21732156612.240112  
  
In [139]:
```

Three green callout bubbles with arrows pointing to specific parts of the console:

- A bubble labeled "Your commands" points to the input line `In [132]: 1+1`.
- A bubble labeled "2: Output" points to the output line `Out[132]: 2`.
- A bubble labeled "Shell as calculator" points to the input line `In [138]: 7.345667 * 8575342.3245 * 345 + 102927 / 2`.

The bottom of the window features a status bar with the following information:

- IPython console
- History log
- Permissions: **RW**
- End-of-lines: **CRLF**
- Encoding: **ASCII**
- Line: **1**
- Column: **1**
- Memory: **67 %**



Master in Software Engineering

Hussam Hourani has over 25 years of Organizations Transformation, VROs, PMO, Large Scale and Enterprise Programs Global Delivery, Leadership, Business Development and Management Consulting. His client experience is wide ranging across many sectors but focuses on Performance Enhancement, Transformation, Enterprise Program Management, Artificial Intelligence and Data Science.