

Lecture 1

Truong Quang Nhat

Ngày 27 tháng 8 năm 2020

Nội dung

Dữ liệu là gì?

Cài đặt và sử dụng Python

Chương trình Hello World!

Các cú pháp cơ bản trong Python

Biến trong Python

Kiểu dữ liệu

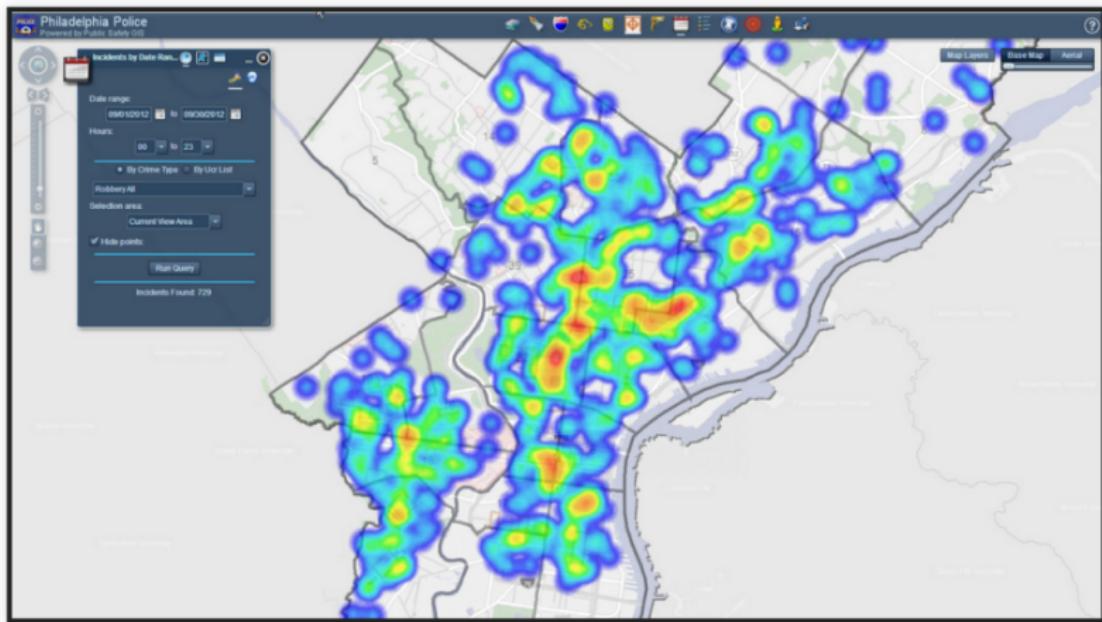
Các toán tử

Sử dụng các thư viện trong Python

Xử lý dữ liệu với thư viện pandas

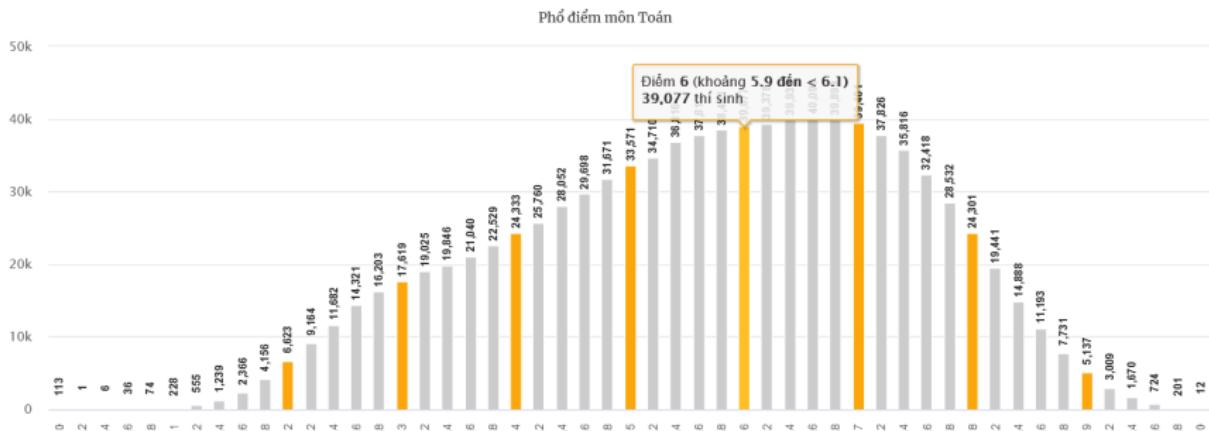
Dữ liệu là gì?

POLICING



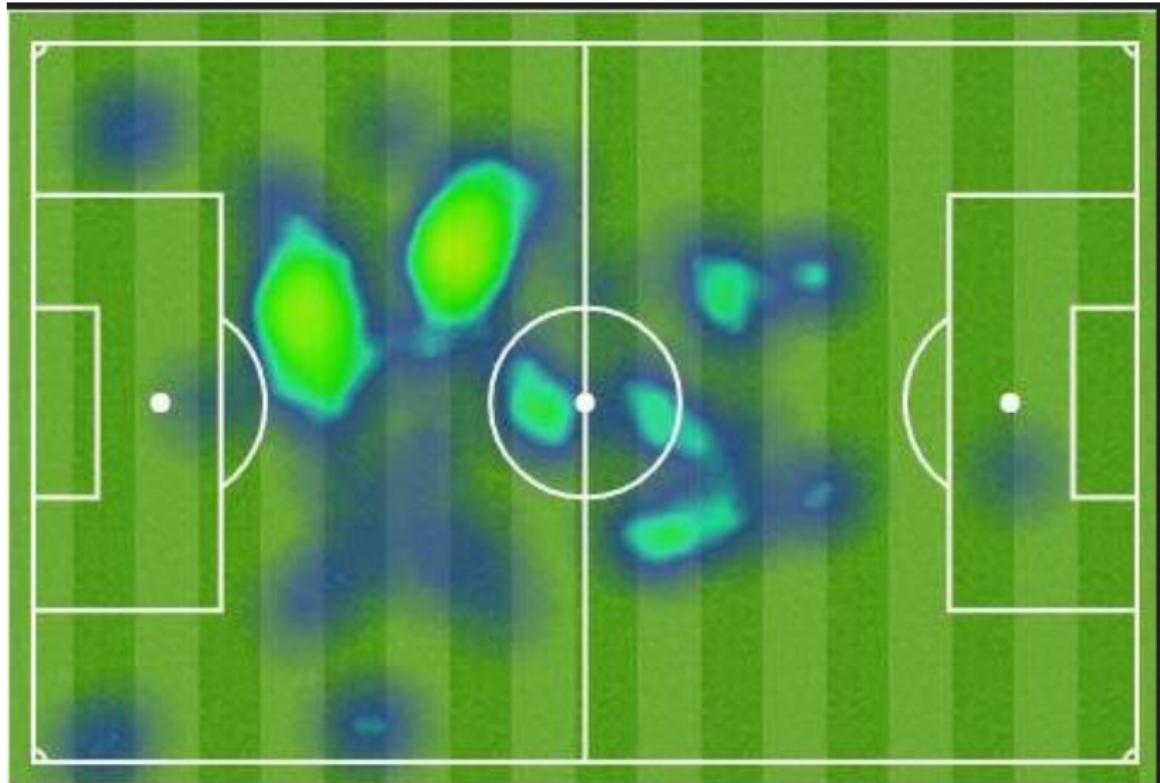
Source: technical.ly

Dữ liệu là gì?



Hình: Điểm thi THPT 2019 môn toán (Nguồn: VNExpress)

Dữ liệu là gì?



Hình: Sơ đồ nhiệt thể hiện phạm vi di chuyển của Messi ở Vicente Calderon

Dữ liệu là gì?

DATA SCIENCE

Amazon.com Recommendations Understand Area Woman Better Than Husband

NEWS

January 9, 2007

VOL 45 ISSUE 01

Human Interest · Science & Technology · Old Internet · Shopping · Relationships



SANDUSKY, OH—Area resident Pamela Meyers was delighted to receive yet another thoughtful CD recommendation from Amazon.com Friday, confirming that the online retail giant has a more thorough, individualized, and nuanced understanding of Meyers' taste than the man who occasionally claims to love her, husband Dean Meyers.



Meyers said she was pleasantly surprised to receive three e-mails from Amazon today alone.

Amazon, which has been tracking Meyers' purchases since she first used the site to order *Football For Dummies* in preparation for attending the 2004 Citrus Bowl as part of her husband's 10th wedding anniversary plans, has shown impressive accuracy at recommending books, movies, music, and even clothing that perfectly match Meyers' tastes. While the powerful algorithms that power Amazon's recommendations generator do not have the advantage of being able to observe Meyers' body language, verbal intonation, or current personal possessions, they have nonetheless proven more effective than Dean, who bases his gift-giving choices primarily on what is needed around the house, what he would like to own, and, most notably, what objects are nearby.

TV + Internet + Voice
From \$29.99/mo each for 12 mos when bundled*
FREE DVR SERVICE

ONLINE ONLY OFFER

Charter Spectrum [LEARN MORE](#) Restrictions apply

ONION VIDEO

[WATCH MORE](#)



Onion Explains: Putin's Russia

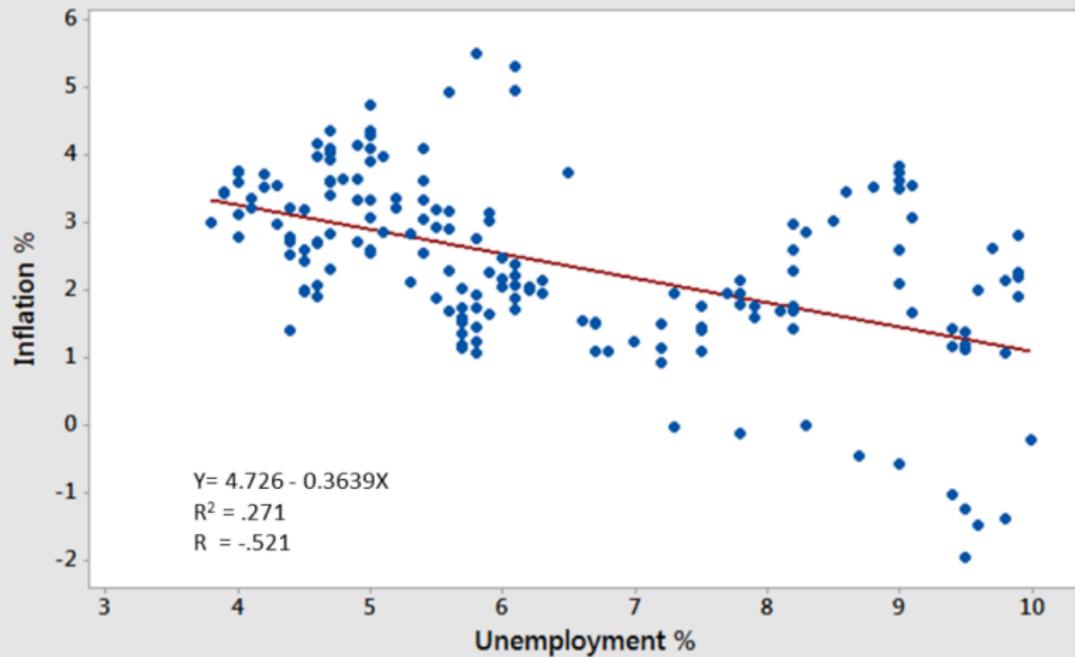


Onion Explains: The Rise Of China

Source: *The Onion*

Dữ liệu là gì?

U.S. Phillips Curve: Inflation vs Unemployment - 1/2000 to 8/2014



Hình: Mối tương quan giữa lạm phát và thất nghiệp

Khai thác dữ liệu

- ▶ Find data
- ▶ Load data
- ▶ Clean data
- ▶ Summarize data
- ▶ Visualize data
- ▶ Present data

Finding data

 Social Security
Official Social Security Website

Accessibility Contact Us FAQs Español Other Languages Sign In

Search...

Home Numbers & Cards Benefits Information for... Business & Government Our Agency

Popular Baby Names

[Popular Baby Names](#)

[Background information](#)

[Number of applications for Social Security cards](#)

Beyond the Top 1000 Names

To provide *popular names* and maintain an acceptable performance level on our servers, we provide only the top 1000 names through [our forms](#). However, we provide almost all names for researchers interested in naming trends.

To safeguard privacy, we restrict our list of names to those with at least 5 occurrences. We provide these data on both a national and state-specific basis, in two separate collections of files, each zipped into a single file. The format of the data in the two files is described in a "readme" file contained in the respective zip files.

- National data  (8Mb)
- State-specific data  (23MB)

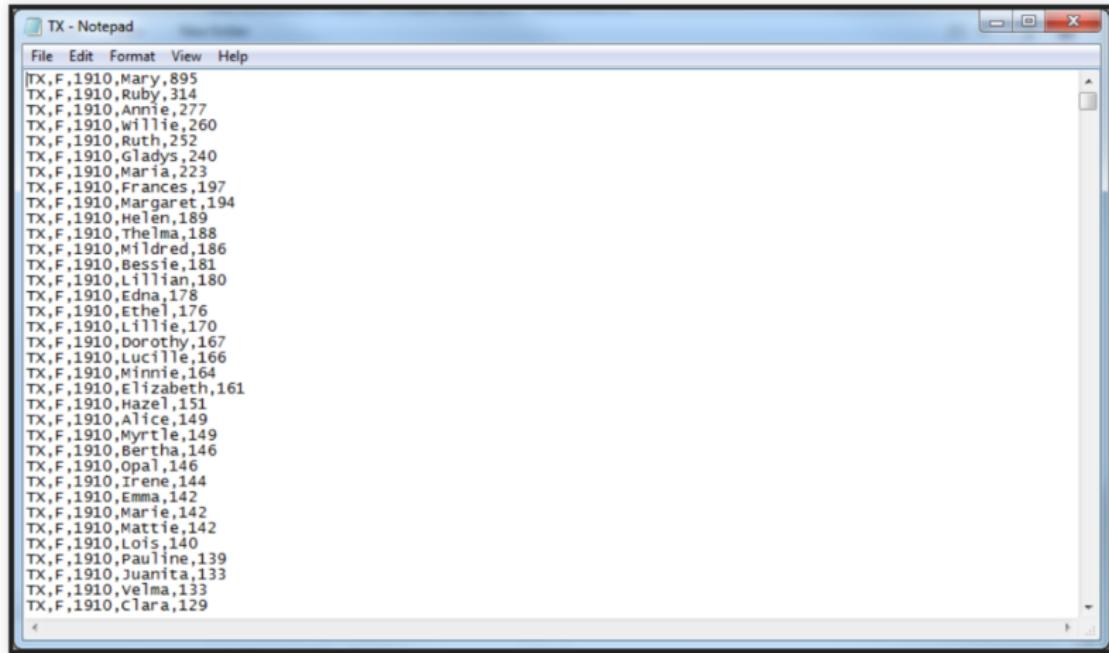
What percentage of all names are represented in the top 1000 names?
For U.S. births in 2014, the top 1000 names represent about 74 percent of all names.

Percentage of all names represented in the top 1000 names

Year of birth	Male	Female	Total
2014	78.96	67.86	73.54
2013	78.89	67.32	73.24
2012	78.61	66.81	72.84
2011	78.76	66.89	72.97
2010	78.69	66.75	72.86
2009	79.02	66.85	73.08
2008	79.51	67.24	73.51
2007	80.10	67.74	74.06
2006	80.63	68.48	74.70
2005	81.34	69.40	75.51



Loading and cleaning data



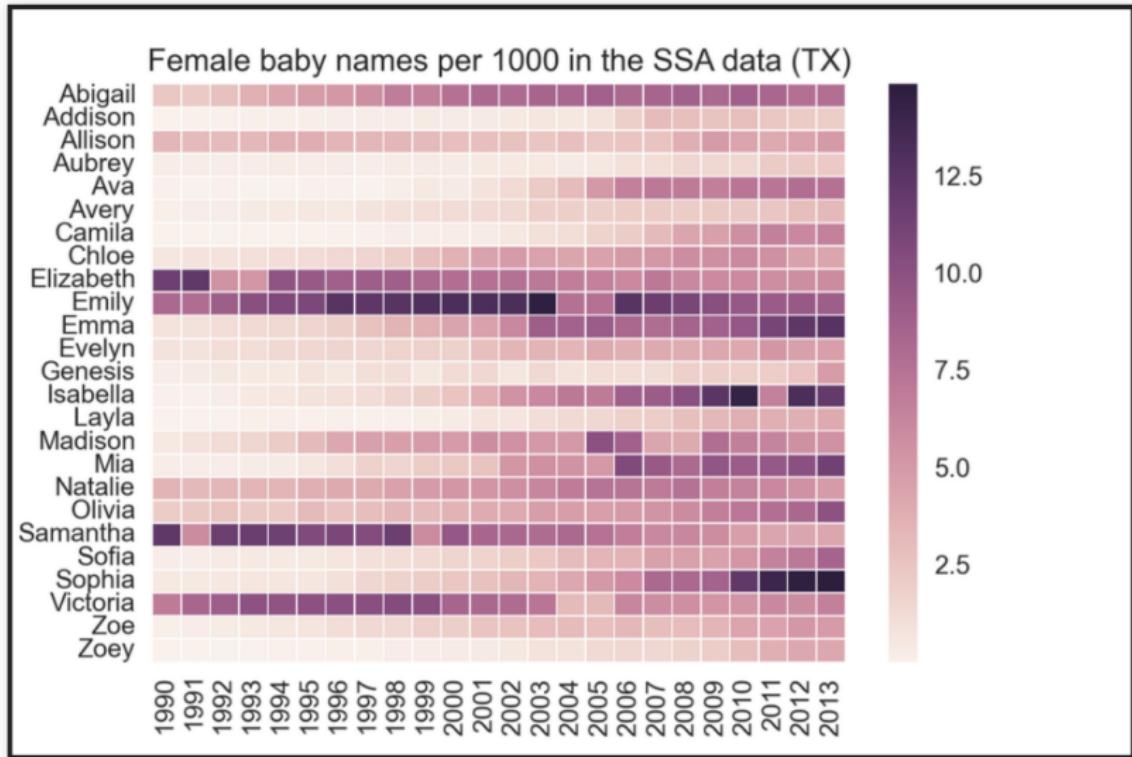
A screenshot of a Windows Notepad window titled "TX - Notepad". The window contains a list of names and their corresponding counts, separated by commas. The names are listed in descending order of count. The file menu includes File, Edit, Format, View, and Help. The window has standard Windows-style borders and a scroll bar on the right side.

Name	Count
TX,F,1910,Mary	895
TX,F,1910,Ruby	314
TX,F,1910,Annie	277
TX,F,1910,Willie	260
TX,F,1910,Ruth	252
TX,F,1910,Gladys	240
TX,F,1910,Maria	223
TX,F,1910,Frances	197
TX,F,1910,Margaret	194
TX,F,1910,Helen	189
TX,F,1910,Thelma	188
TX,F,1910,Mildred	186
TX,F,1910,Bessie	181
TX,F,1910,Lillian	180
TX,F,1910,Edna	178
TX,F,1910,Ethel	176
TX,F,1910,Lillie	170
TX,F,1910,Dorothy	167
TX,F,1910,Lucille	166
TX,F,1910,Minnie	164
TX,F,1910,Elizabeth	161
TX,F,1910,Hazel	151
TX,F,1910,Alice	149
TX,F,1910,Myrtle	149
TX,F,1910,Bertha	146
TX,F,1910,Opal	146
TX,F,1910,Irene	144
TX,F,1910,Emma	142
TX,F,1910,Marie	142
TX,F,1910,Mattie	142
TX,F,1910,Lois	140
TX,F,1910,Pauline	139
TX,F,1910,Juanita	133
TX,F,1910,Velma	133
TX,F,1910,Clara	129

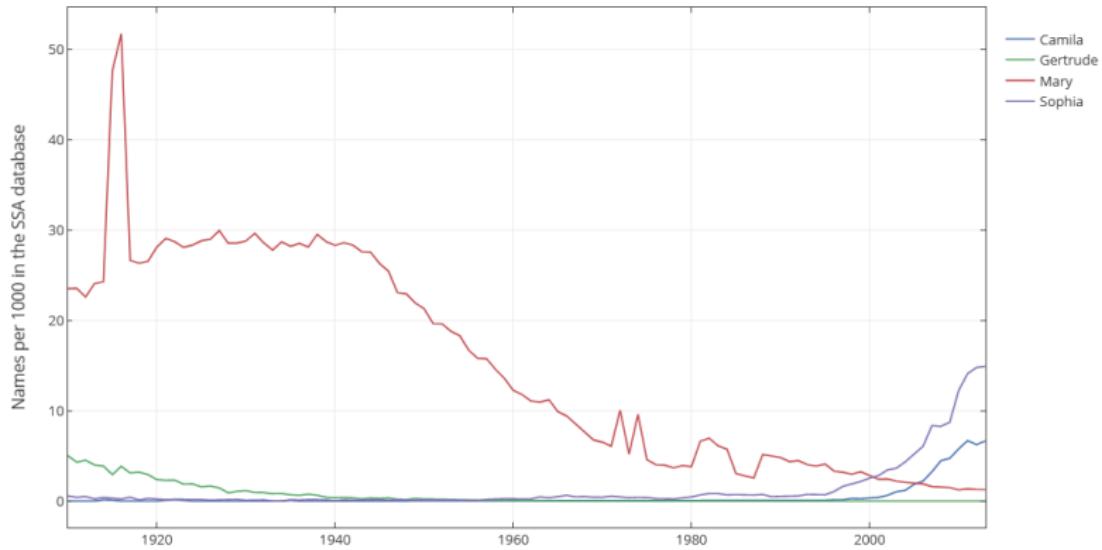
Summarizing data

	state	gender	year	name	count	per1000	rank
177374	TX	F	2013	Sophia	2381	14.905098	1
177375	TX	F	2013	Emma	2043	12.789213	2
177376	TX	F	2013	Isabella	1941	12.150691	3
177377	TX	F	2013	Mia	1830	11.455829	4
177378	TX	F	2013	Olivia	1598	10.003506	5
177379	TX	F	2013	Emily	1449	9.070763	6
177380	TX	F	2013	Sofia	1380	8.638822	7
177381	TX	F	2013	Abigail	1250	7.825020	8
177382	TX	F	2013	Ava	1227	7.681040	9
177383	TX	F	2013	Victoria	1067	6.679437	10

Visualizing data



Presenting data

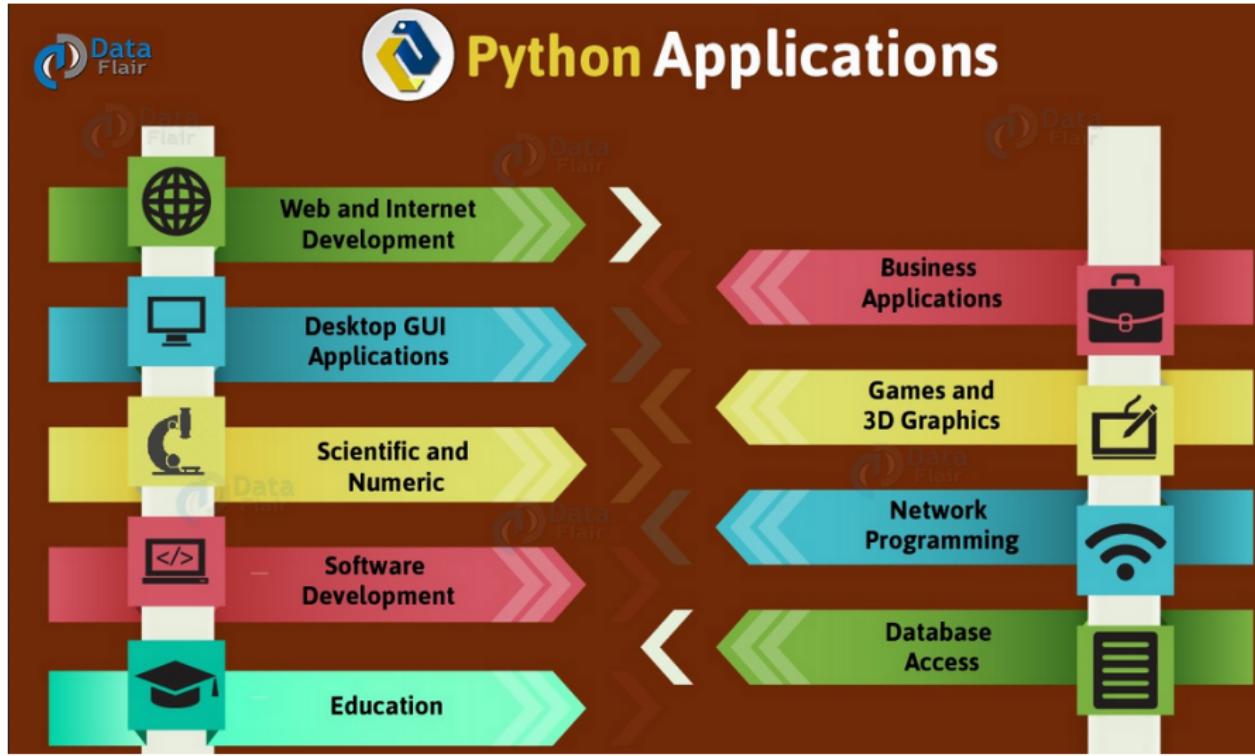


Python



- ▶ Được tạo ra bởi Guido Van Rossum.
- ▶ Thiết kế bắt đầu vào cuối những năm 1980 và được phát hành lần đầu tiên vào tháng 2 năm 1991.

Ứng dụng của Python



Cài đặt Python

Cài đặt Python:

- ▶ Windows: <https://www.python.org/downloads/windows/>
- ▶ MacOs: <https://www.python.org/downloads/mac-osx/>
- ▶ Other Platforms:
<https://www.python.org/download/other/>

Cài đặt IDE:

- ▶ Spyder:
<https://docs.spyder-ide.org/installation.html>
- ▶ PyCharm: <https://www.jetbrains.com/pycharm/>
- ▶ VS Code: <https://code.visualstudio.com/>

Làm việc với Jupyter notebook

Cài đặt: https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html

Làm việc với Jupyter notebook

Cài đặt: https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html

Giao diện:

The screenshot shows a Jupyter Notebook interface running in a web browser at `localhost:8888/notebooks/1.ipynb`. The title bar says "jupyter Errors (unsaved changes)". The toolbar includes File, Edit, View, Insert, Cell, Kernel, Widgets, Help, Trusted, Python 3, and Logout. Below the toolbar is a toolbar with icons for file operations like Open, Save, Print, and Cell types like Code, Markdown, and Rich.

Cell 2 contains the following code:

```
2. 4_total = book print(total + " I am reading")
3. pirst"my eyes are black")
4. print'Save the Jupyter notebook')
5. first_name = "Rubina" print(FIRST_NAME)
```

Cell 79 contains the code:

```
In [79]: print('Saima Academy')
```

Output for Cell 79 shows the error:

```
File "<ipython-input-79-044c48fe3c75>", line 1
    print('Saima Academy')
               ^
SyntaxError: EOL while scanning string literal
```

Cell 80 contains the code:

```
In [80]: Print("Saima Academy")
```

The bottom status bar shows "In [79]".

Hình: Giao diện Jupyter Notebook

Hello World

In ra dòng chữ "Hello World" trong Python.

```
>>> a="Hello World"  
>>> print a  
Hello World
```

hoặc đơn giản hơn:

```
>>> print("Hello World")  
Hello World
```

Định danh

Một định danh (identifier) trong Python là một tên được sử dụng để nhận diện một biến, một hàm, một lớp, hoặc một đối tượng.

Các quy tắc:

- ▶ Một định danh là một dãy ký tự hoặc chữ số.
- ▶ Không có ký tự đặc biệt nào được sử dụng (ngoại trừ dấu gạch dưới) như một định danh.
- ▶ Không đặt trùng các từ khoá
- ▶ Tên lớp bắt đầu với một chữ cái hoa. Tất cả định danh khác bắt đầu với một chữ cái thường.
- ▶ Python phân biệt chữ hoa và chữ thường.

Các từ khóa trong Python

**and, exec, not, assert, finally, or, break, for, pass, class, from,
print, continue, global, raise, def, if, return, del, import, try,
elif, in, while, else, is, with, except, lambda, yield**

Trích dẫn trong Python

Python chấp nhận trích dẫn đơn ('), kép ("") và trích dẫn tam (''' hoặc """) để biểu thị các hằng chuỗi, miễn là các trích dẫn này có cùng kiểu mở và đóng.

Trích dẫn tam được sử dụng để trải rộng chuỗi được trích dẫn qua nhiều dòng. Dưới đây là tất cả các trích dẫn hợp lệ:

```
word = 'word'  
sentence = "This is a sentence."  
paragraph = """This is a paragraph. It is  
made up of multiple lines and sentences."""
```

Comment trong Python

Ta có thể sử dụng dấu `#` để comment 1 dòng đơn trong Python, hoặc sử dụng cặp dấu trích dẫn `"""` để comment nhiều dòng. Cụ thể như ví dụ sau:

```
#single line comment
print("Hello Python")
"""This is
multiline comment """
```

Phép gán trong Python

Trong Python, chúng ta không cần khai báo biến một cách tường minh. Khi bạn gán bất cứ giá trị nào cho biến thì biến đó được khai báo một cách tự động. Phép gán được thực hiện bởi toán tử "`=`". Toán hạng trái của toán tử "`=`" là tên biến và toán hạng phải là giá trị được lưu trữ trong biến. Ví dụ:

```
a = 20          # Phep gan a bang so nguyen 20
b = 100.1        # Phep gan b bang so thuc 100
ten = "Nhat"      # Phep gan bien "ten" bang chuoi
                  gia tri 'Nhat'

print(a)
print(b)
print(ten)
```

Ta được kết quả dưới đây:

```
20
100.1
Nhat
```

Gán nhiều biến trong Python

```
a = b = c = 1 #Gan nhieu bien cung bang gia tri 1  
x, y, z = 5, 10, 15 #Gan nhieu bien bang nhieu gia  
                     tri khac nhau
```

ta được kết quả:

```
>>> print(x)  
5  
>>> print(y)  
10  
>>> print(z)  
15
```

Kiểu dữ liệu

Python có 5 kiểu dữ liệu chuẩn là:

- ▶ Kiểu Number
- ▶ Kiểu String
- ▶ Kiểu List
- ▶ Kiểu Tuple
- ▶ Kiểu Dictionary

Các toán tử

Python hỗ trợ các loại toán tử sau:

- ▶ Toán tử số học
- ▶ Toán tử quan hệ (còn gọi là toán tử so sánh)
- ▶ Toán tử gán
- ▶ Toán tử logic
- ▶ Toán tử membership
- ▶ Toán tử identify
- ▶ Toán tử thao tác bit

Toán tử số học

- ▶ // Thực hiện phép chia lấy phần nguyên
- ▶ + Phép cộng
- ▶ - Phép trừ
- ▶ * Phép nhân
- ▶ / Phép chia
- ▶ % Phép chia lấy phần dư
- ▶ ** Phép lấy số mũ (ví dụ $2^{**}3$ cho kết quả là 8)

Ví dụ

```
>>> 10 + 20  
30  
>>> 20 - 10  
10  
>>> 10 * 2  
20  
>>> 10 / 2  
5  
>>> 10 % 3  
1  
>>> 2 ** 3  
8  
>>> 10 // 3  
3
```

Toán tử so sánh

Python hỗ trợ các toán tử quan hệ (toán tử so sánh) sau:

- ▶ < Nhỏ hơn.
- ▶ > Lớn hơn
- ▶ <= Nhỏ hơn hoặc bằng
- ▶ >= Lớn hơn hoặc bằng
- ▶ == Bằng
- ▶ != Không bằng
- ▶ <> Không bằng (tương tự !=)

Ví dụ

```
>>> 10<20
True
>>> 10>20
False
>>> 10<=10
True
>>> 20>=15
True
>>> 5==6
False
>>> 5 !=6
True
>>> 10<>2
True
```

Toán tử gán

- ▶ = Phép gán
- ▶ /= Chia toán hạng trái cho toán hạng phải, và gán kết quả cho toán hạng trái
- ▶ += Cộng và gán
- ▶ -= Trừ và gán
- ▶ *= Nhân và gán
- ▶ %= Chia lấy phần dư và gán
- ▶ **= Lấy số mũ và gán
- ▶ //= Thực hiện phép chia // và gán

Ví dụ

```
>>> c = 10
>>> c
10
>>> c += 5
>>> c
15
>>> c -= 5
>>> c
10
>>> c *= 2
>>> c
20
>>> c /= 2
>>> c
10
>>> c %= 3
>>> c
1
>>> c = 5
>>> c **= 2
>>> c
25
```

Toán tử logic

Python hỗ trợ các toán tử logic sau:

- ▶ **and** Phép Và. Nếu cả hai điều kiện là true thì kết quả sẽ là true
- ▶ **or** Phép Hoặc. Nếu một trong hai điều kiện là true thì kết quả là true
- ▶ **not** Phép phủ định. Được sử dụng để đảo ngược trạng thái logic của toán hạng

Ví dụ

```
>>> a=5>4 and 3>2
>>> a
True
>>> b=5>4 or 3<2
>>> b
True
>>> c=not(5>4)
>>> c
False
```

Toán tử membership

Toán tử membership trong Python kiểm tra xem biến này có nằm trong dãy (có là một trong các thành viên của dãy) hay không. Có hai toán tử membership trong Python là:

- ▶ `in` Trả về true nếu một biến là nằm trong dãy các biến, nếu không là false
- ▶ `not in` Trả về true nếu một biến là không nằm trong dãy các biến, nếu không là false

Ví dụ

```
list1=[1,2,3,4,5]
list2=[6,7,8,9]
for item in list1:
    if item in list2:
        print("Co trung lap")
    else:
        print("Khong trung lap")
```

Ta thu được kết quả:

Khong trung lap

Sử dụng các thư viện trong Python

Cài đặt thư viện:

Thông thường, ta sẽ cài đặt thông qua trình quản lý thư viện pip với cú pháp: pip install ten_thu_vien. Chẳng hạn, để cài đặt các thư viện Seaborn, Bokeh, Plotly, ta tiến hành chạy các câu lệnh sau:

```
pip install seaborn  
pip install bokeh  
pip install plotly
```

Sử dụng các thư viện trong Python

Cài đặt thư viện:

Thông thường, ta sẽ cài đặt thông qua trình quản lý thư viện pip với cú pháp: pip install ten_thu_vien. Chẳng hạn, để cài đặt các thư viện Seaborn, Bokeh, Plotly, ta tiến hành chạy các câu lệnh sau:

```
pip install seaborn  
pip install bokeh  
pip install plotly
```

Sử dụng thư viện:

Để sử dụng một thư viện, ta sử dụng lệnh `import`. Chẳng hạn, để sử dụng thư viện seaborn ta tiến hành như sau:

```
import seaborn
```

hoặc, nếu muốn viết tắt tên thư viện để tiện sử dụng, ta có thể làm như dưới đây:

```
import seaborn as sns
```

Xử lý dữ liệu với thư viện pandas

Load thư viện pandas và seaborn:

```
#Load pandas library
import pandas as pd
import seaborn as sns
```

Load bộ data diamonds

```
diamonds_df = sns.load_dataset('diamonds')
```

Lệnh .head()

Ta có thể xem nhanh các giá trị đầu tiên của bộ dữ liệu trên như sau:

```
diamonds_df.head()
```

Ta được kết quả dưới đây:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75

Hình: 5 giá trị đầu tiên của bộ dữ liệu diamonds

Lệnh .shape

Để biết số hàng và cột của một DataFrame, ta có thể sử dụng lệnh `shape` như sau:

```
>>> diamonds_df.shape  
(53940, 10)
```

Như vậy, DataFrame của ta có 53940 hàng, 10 cột.

Lệnh describe()

Để tính toán các giá trị thống kê của data như mean, median, min, max, cùng các phân vị, ta sử dụng lệnh describe() như sau:

```
diamonds_df.describe()
```

Ta được kết quả dưới đây:

	carat	depth	table	price	x	y	z
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000
mean	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734
std	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699
min	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000
25%	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000
50%	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000
75%	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000
max	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000

Lệnh describe()

Để phân tích các biến định tính, ta có thể sử dụng thuộc tính `include=object` trong lệnh `describe()` như sau:

```
diamonds_df.describe(include=object)
```

Ta được kết quả dưới đây:

	cut	color	clarity
count	53940	53940	53940
unique	5	7	8
top	Ideal	G	SI1
freq	21551	11292	13065

Lệnh info()

Để xem thông tin các biến trong DataFrame, ta sử dụng lệnh dưới đây:

```
diamonds_df.info()
```

Ta thu được kết quả sau:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 10 columns):
carat      53940 non-null float64
cut        53940 non-null object
color      53940 non-null object
clarity    53940 non-null object
depth      53940 non-null float64
table      53940 non-null float64
price      53940 non-null int64
x          53940 non-null float64
y          53940 non-null float64
z          53940 non-null float64
dtypes: float64(6), int64(1), object(3)
memory usage: 4.1+ MB
```

Truy xuất dữ liệu cột

Để chọn một cột bất kỳ trong DataFrame, ta sẽ sử dụng toán tử `"."` hoặc `"[]"`. Chẳng hạn, để chọn cột **cut** của bộ dữ liệu trên ta viết: `diamonds_df.cut` hoặc `diamonds_df['cut']`.

Cụ thể hơn, nếu muốn chọn ra các dữ liệu mà giá trị ở cột **cut** là **Ideal**, ta kết hợp với lệnh `loc()` như sau:

```
diamonds_low_df = diamonds_df.loc[diamonds_df['cut'] ==  
                                   'Ideal']  
diamonds_low_df.head()
```

Tâ được kết quả dưới đây:

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
11	0.23	Ideal	J	VS1	62.8	56.0	340	3.93	3.90	2.46
13	0.31	Ideal	J	SI2	62.2	54.0	344	4.35	4.37	2.71
16	0.30	Ideal	I	SI2	62.0	54.0	348	4.31	4.34	2.68
39	0.33	Ideal	I	SI2	61.8	55.0	403	4.49	4.51	2.78

Thêm cột vào DataFrame

Để thêm một cột price_per_carat vào DataFrame đã cho, ta tiến hành như sau:

```
diamonds_df ['price_per_carat'] = diamonds_df ['price'] /  
                                 diamonds_df ['carat']  
diamonds_df .head()
```

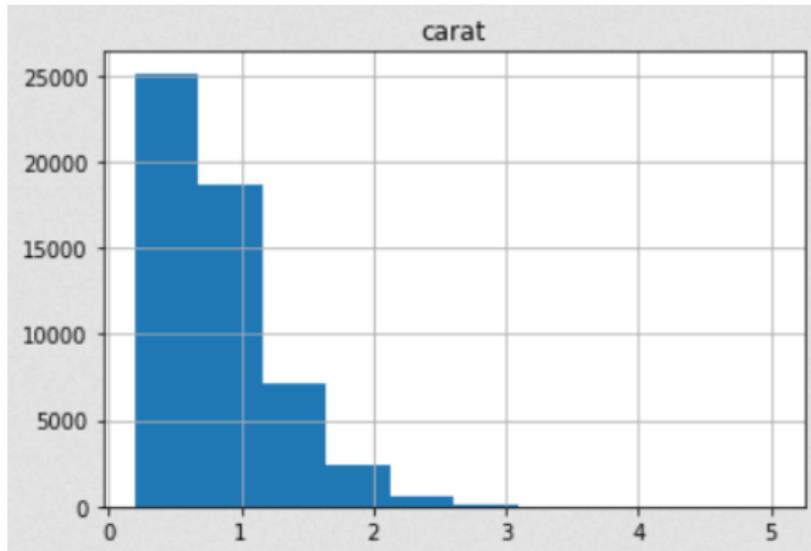
	carat	cut	color	clarity	depth	table	price	x	y	z	price_per_carat
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43	1417.391304
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31	1552.380952
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31	1421.739130
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63	1151.724138
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75	1080.645161

Vẽ biểu đồ Histogram

Ta có thể vẽ Histogram của bộ data trên như sau:

```
diamonds_df.hist(column='carat')
```

Kết quả:

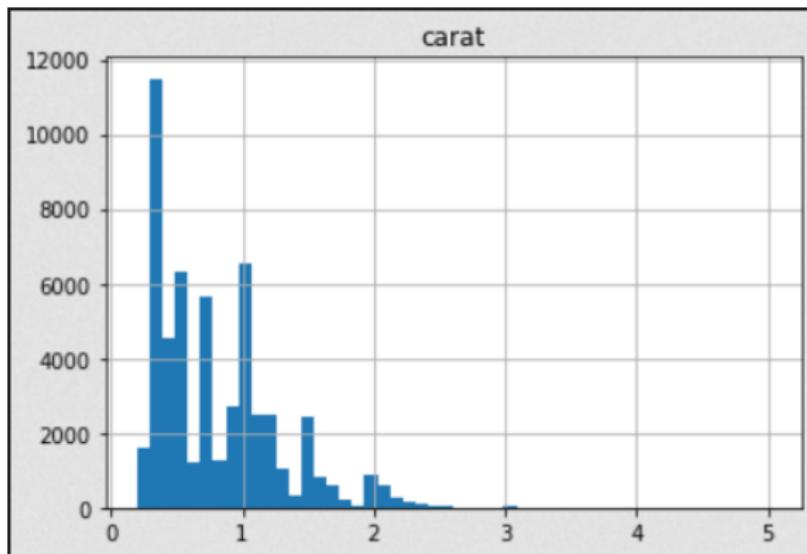


Histogram

Ngoài ra, ta có thể thay đổi giá trị bins của hàm hist như sau:

```
diamonds_df.hist(column='carat', bins=50)
```

Kết quả:

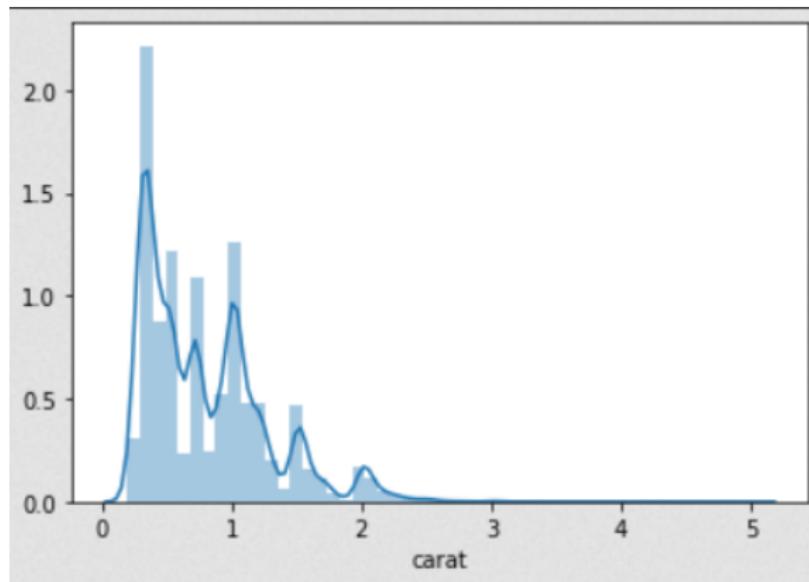


Histogram

Ta cũng có thể sử dụng seaborn để vẽ histogram như sau:

```
sns.distplot(diamonds_df.carat)
```

Kết quả:



Histogram

Loại bỏ đồ thị đường

```
sns.distplot(diamonds_df.carat, kde = False)
```

Kết quả:

