### **Thống kê - Python Pandas**

**DataFrame** − “index” (axis=0, default), “columns” (axis=1)

Ta hãy tạo một DataFrame và sử dụng đối tượng này trong suốt bài này

**import** pandas **as** pd

**import** numpy **as** np

#Create a Dictionary of series

d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack',

'Lee','David','Gasper','Betina','Andres']),

'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

}

#Create a DataFrame

df = pd.DataFrame(d)

**print** df

Age Name Rating

0 25 Tom 4.23

1 26 James 3.24

2 25 Ricky 3.98

3 23 Vin 2.56

4 30 Steve 3.20

5 29 Smith 4.60

6 23 Jack 3.80

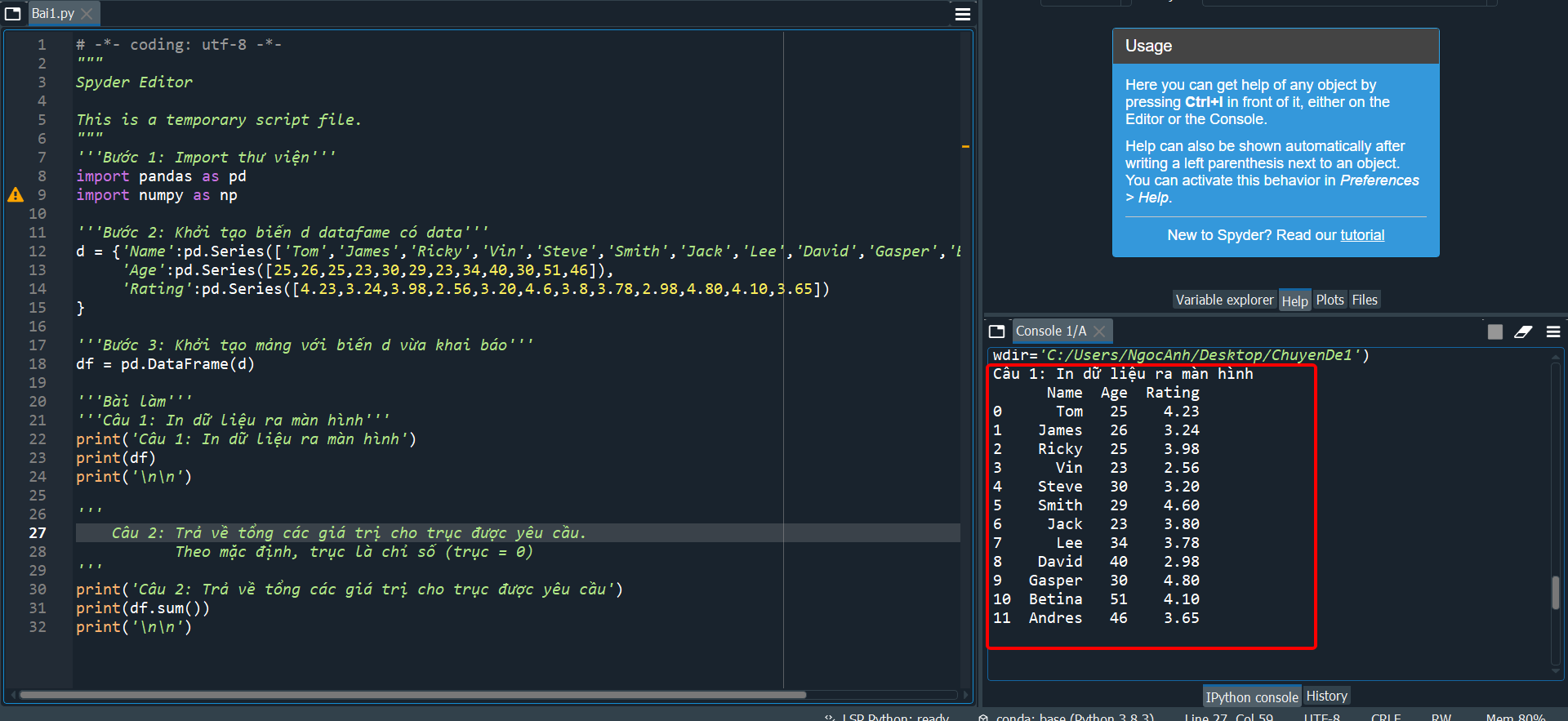
7 34 Lee 3.78

8 40 David 2.98

9 30 Gasper 4.80

10 51 Betina 4.10

11 46 Andres 3.65



### **1. sum()**

Trả về tổng các giá trị cho trục được yêu cầu. Theo mặc định, trục là chỉ số (trục = 0)

**import** pandas **as** pd

**import** numpy **as** np

#Create a Dictionary of series

d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack',

'Lee','David','Gasper','Betina','Andres']),

'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

}

#Create a DataFrame

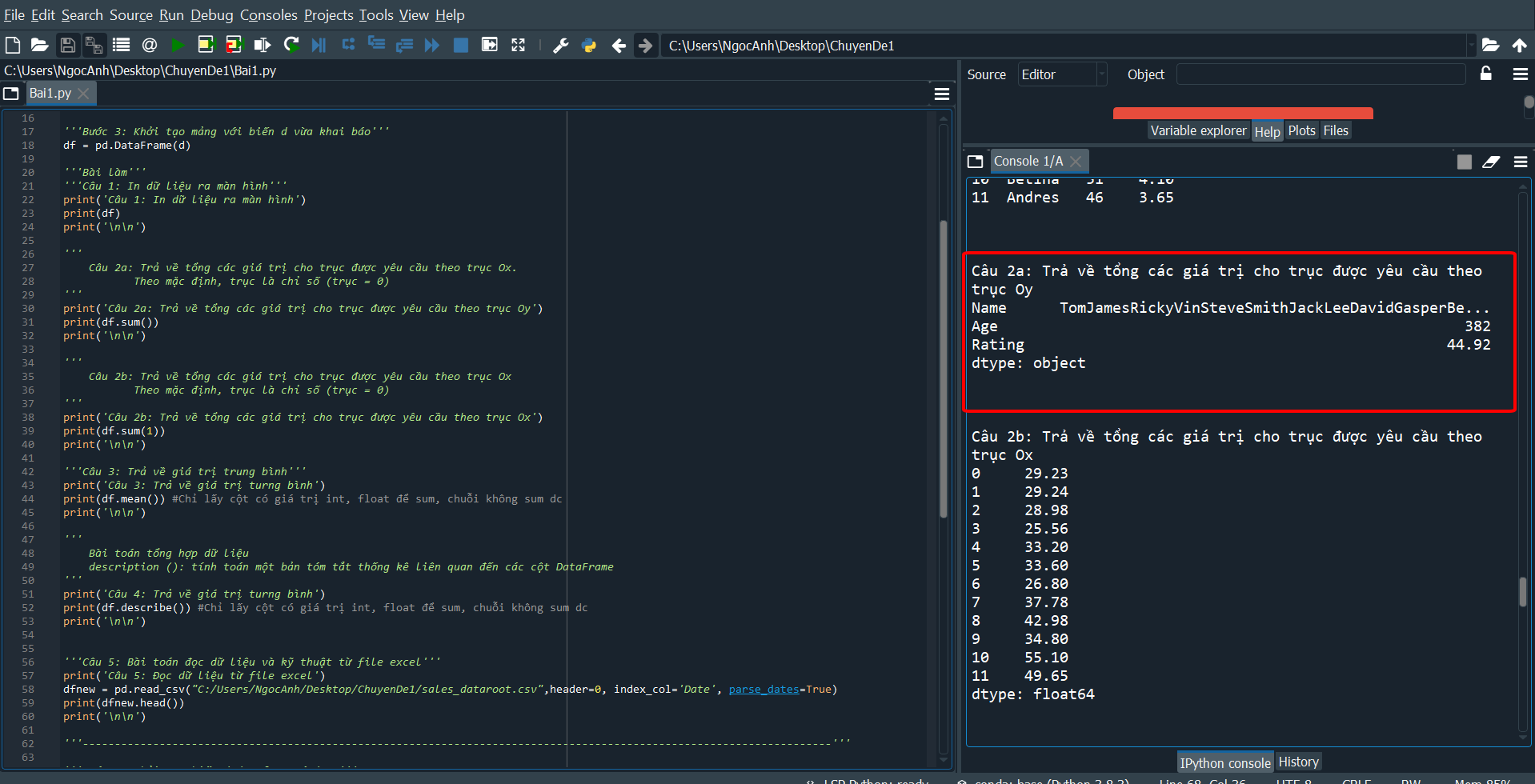
df = pd.DataFrame(d)

Age 382

Name TomJamesRickyVinSteveSmithJackLeeDavidGasperBe...

Rating 44.92

dtype: object



**import** pandas **as** pd

**import** numpy **as** np

#Create a Dictionary of series

d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack',

'Lee','David','Gasper','Betina','Andres']),

'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

}

#Create a DataFrame

df = pd.DataFrame(d)

**print** df.sum(1)

0 29.23

1 29.24

2 28.98

3 25.56

4 33.20

5 33.60

6 26.80

7 37.78

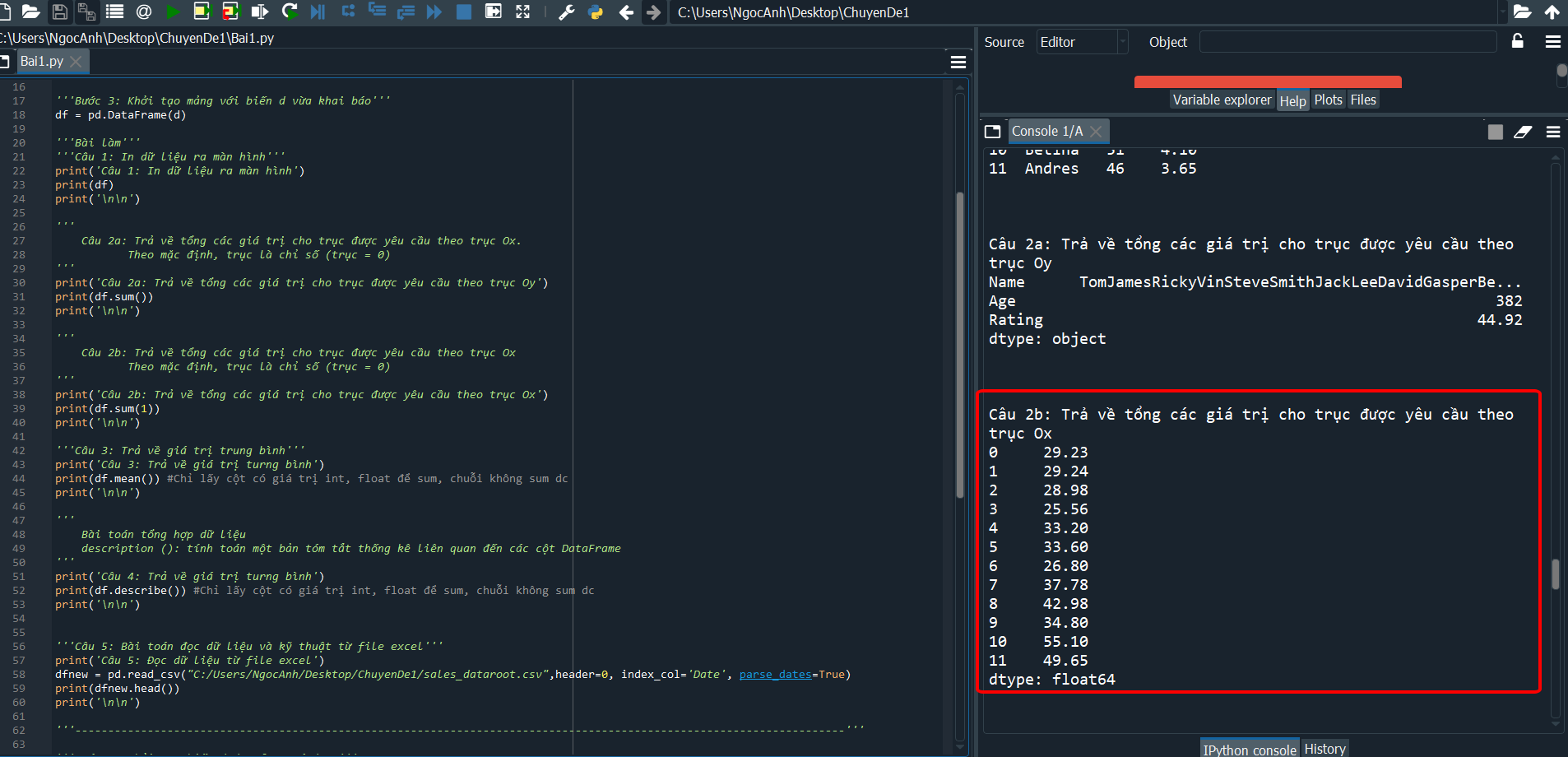
8 42.98

9 34.80

10 55.10

11 49.65

dtype: float64



### **mean()**

Trả về giá trị trung bình

**import** pandas **as** pd

**import** numpy **as** np

#Create a Dictionary of series

d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack',

'Lee','David','Gasper','Betina','Andres']),

'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

}

#Create a DataFrame

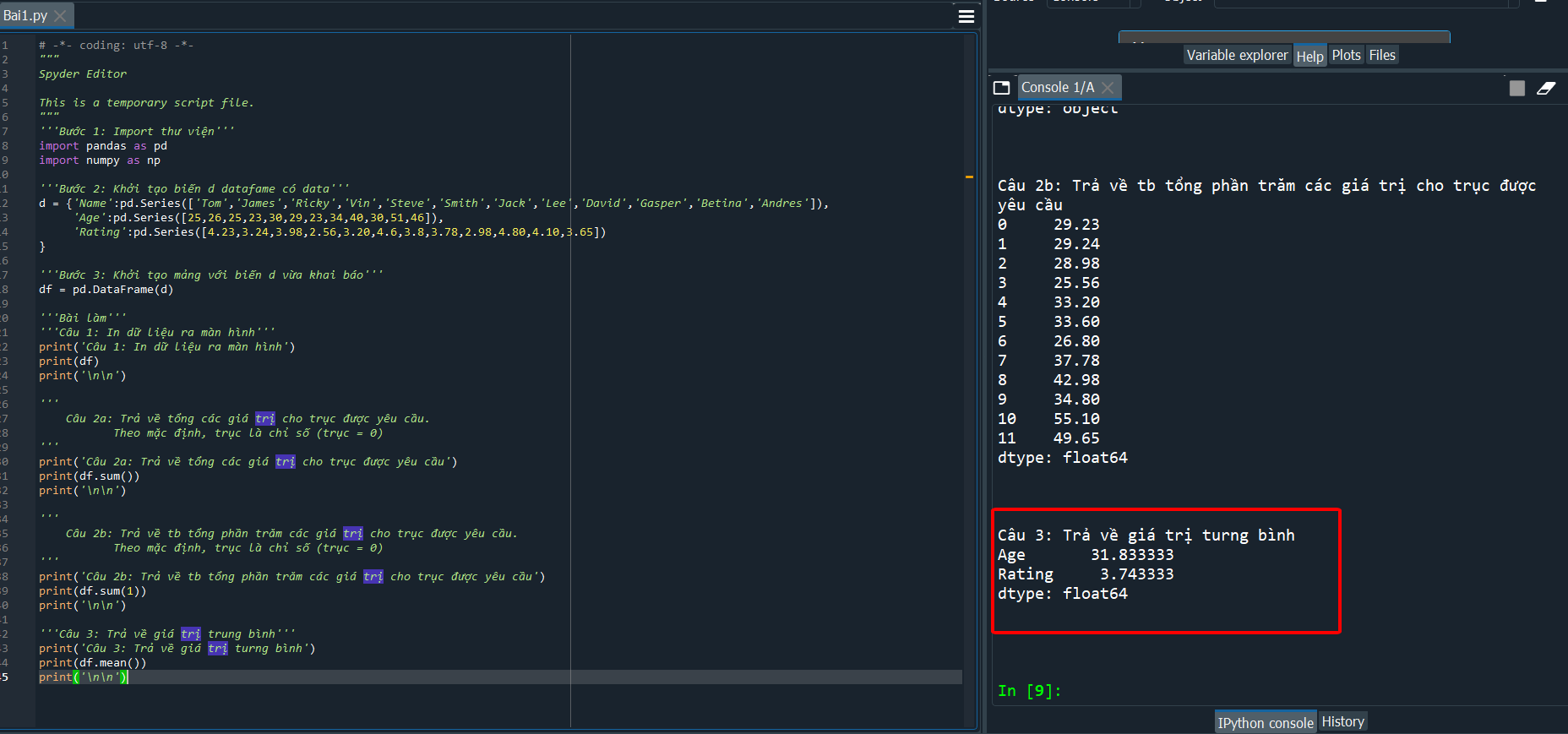
df = pd.DataFrame(d)

**print** df.mean()

Age 31.833333

Rating 3.743333

dtype: float64



## **Tổng hợp dữ liệu :**

Hàm **description ()** tính toán một bản tóm tắt thống kê liên quan đến các cột DataFrame.

**import** pandas **as** pd

**import** numpy **as** np

#Create a Dictionary of series

d = {'Name':pd.Series(['Tom','James','Ricky','Vin','Steve','Smith','Jack',

'Lee','David','Gasper','Betina','Andres']),

'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),

'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])

}

#Create a DataFrame

df = pd.DataFrame(d)

**print** df.describe()

Age Rating

count 12.000000 12.000000

mean 31.833333 3.743333

std 9.232682 0.661628

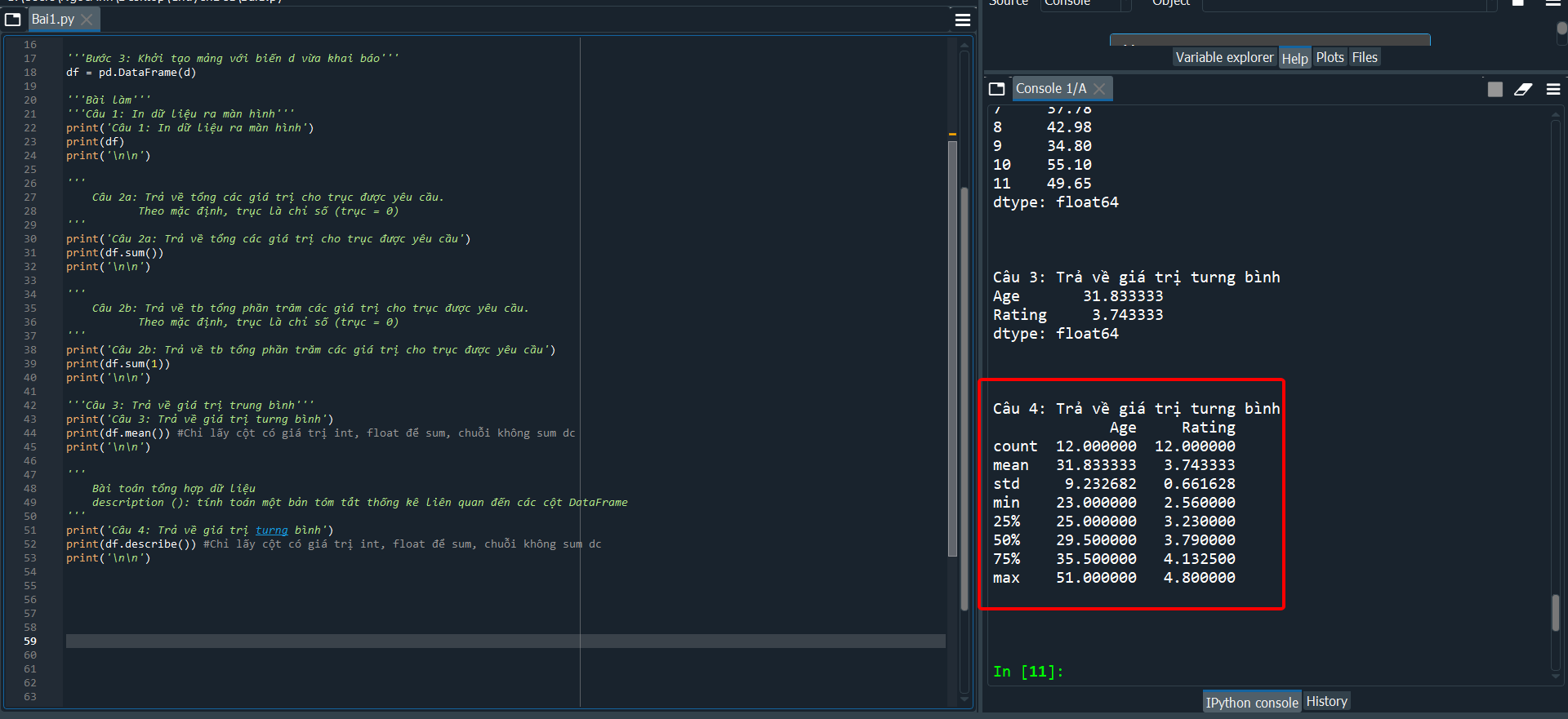
min 23.000000 2.560000

25% 25.000000 3.230000888888

50% 29.500000 3.790000

75% 35.500000 4.132500

max 51.000000 4.800000



### **Bài 10: Đọc dữ liệu và kĩ thuật reindexing - Python Pandas**

Ví dụ: Tôi sử dụng "sales\_14.csv" từ link sau <https://gitlab.com/bambootran89/vimentor_data/blob/master/sales_14.csv>.

>>> import pandas as pd

>>> df = pd.read\_csv("sales\_14.csv",header=0, index\_col='Date', parse\_dates=True)

>>> df.head()

Company Product Units

Date

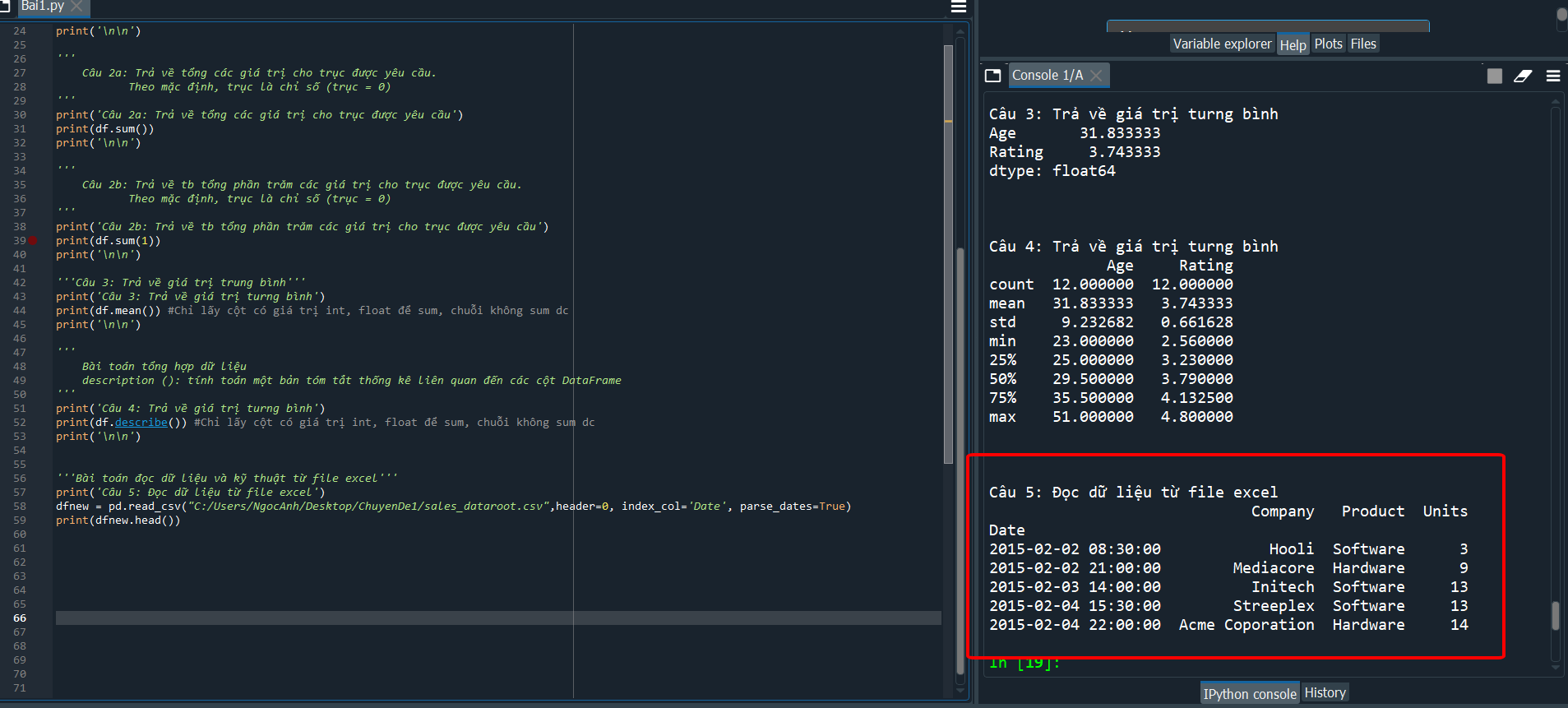
2015-02-02 08:30:00 Hooli Software 3

2015-02-02 21:00:00 Mediacore Hardware 9

2015-02-03 14:00:00 Initech Software 13

2015-02-04 15:30:00 Streeplex Software 13

2015-02-04 22:00:00 Acme Coporation Hardware 14



### **Bài 11: Làm việc với dữ liệu Text (P1) - Python Pandas**

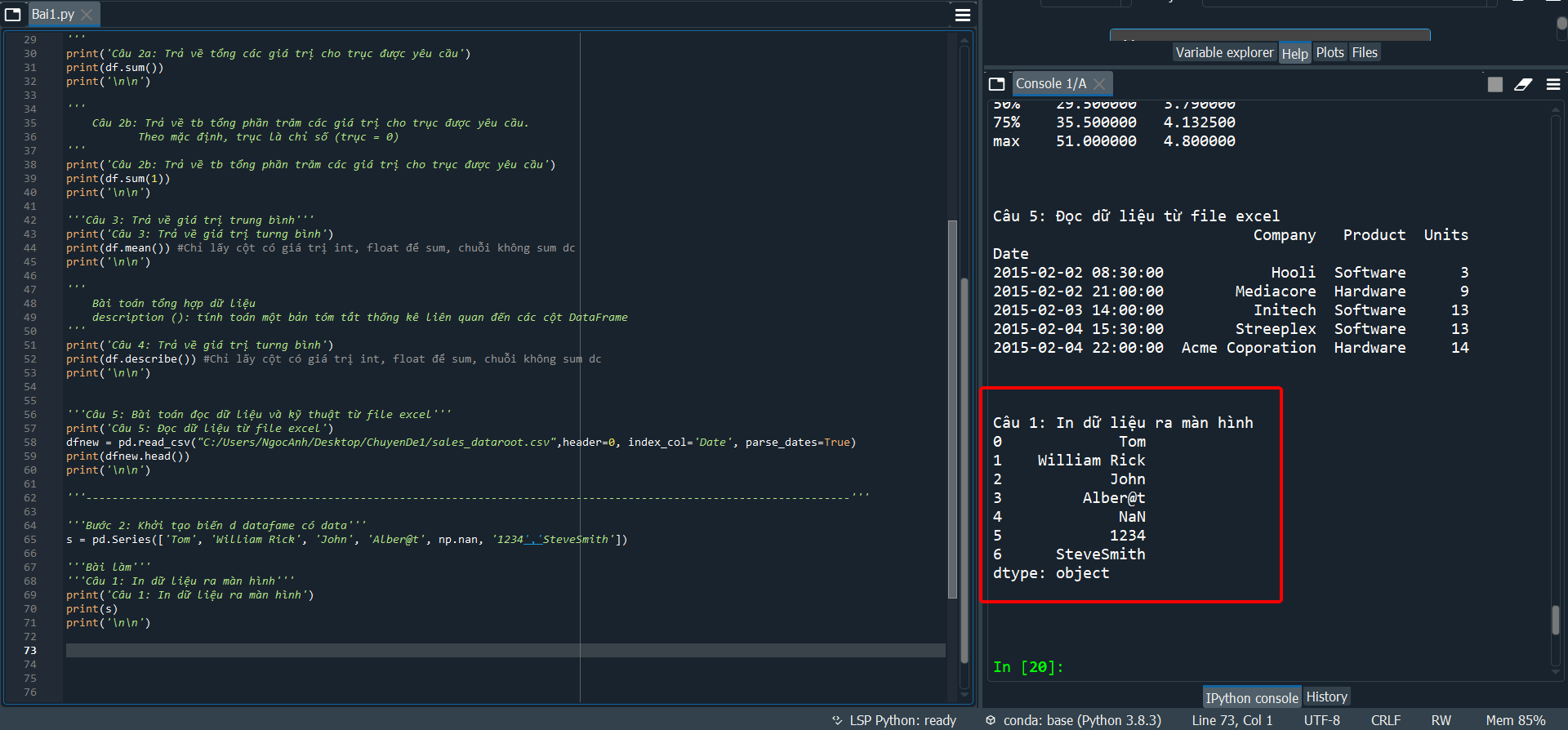
|  |  |
| --- | --- |
| **Số thứ tự** | **Hàm và chức năng** |
| 1 | lower() : chuyển chuỗi String thành chữ thường |
| 2 | upper() : chuyển chuỗi String thành chữ in hoa |
| 3 | len() : Độ dài chuỗi |
| 4 | strip() : Loại bỏ khoảng trắng |
| 5 | split(' ') : Cắt chuỗi |
| 6 | cat(sep=' ') : Nối chuỗi |
| 7 | get\_dummies() : Trả về DataFrame với các giá trị được mã hóa một lần. |
| 8 | contains(pattern) : Trả về giá trị Boolean True cho mỗi phần tử nếu chuỗi con chứa trong phần tử, ngược lại False. |
| 9 | replace(a,b) : Thay thế giá trị a và b |
| 10 | repeat(value) : Lặp lại từng phần tử với số lần xác định. |
| 11 | count(pattern):Trả về số lần xuất hiện của pattern trong mỗi phần tử. |
| 12 | startswith(pattern) : Trả về true nếu phần tử trong Chuỗi bắt đầu bằng pattern. |
| 13 | endswith(pattern):Trả về true nếu phần tử trong Chuỗi kết thúc bằng pattern. |
| 14 | find(pattern) : Trả về vị trí đầu tiên của lần xuất hiện đầu tiên của pattern. |
| 15 | findall(pattern) : Trả về danh sách tất cả các lần xuất hiện của pattern. |
| 16 | swapcase : Hoán đổi chữ hoa / chữ thường. |
| 17 | islower() : Kiểm tra xem tất cả các ký tự trong mỗi chuỗi có viết thường hay không. Trả về Boolean |
| 18 | isupper() : Kiểm tra xem tất cả các ký tự trong mỗi chuỗi có viết hoa hay không. Trả về Boolean. |
| 19 | isnumeric() : Kiểm tra xem tất cả các ký tự trong mỗi chuỗi có phải là số hay không. Trả về Boolean. |

**import** pandas **as** pd

**import** numpy **as** np

s = pd.Series(['Tom', 'William Rick', 'John', 'Alber@t', np.nan, '1234','SteveSmith'])

**print** s



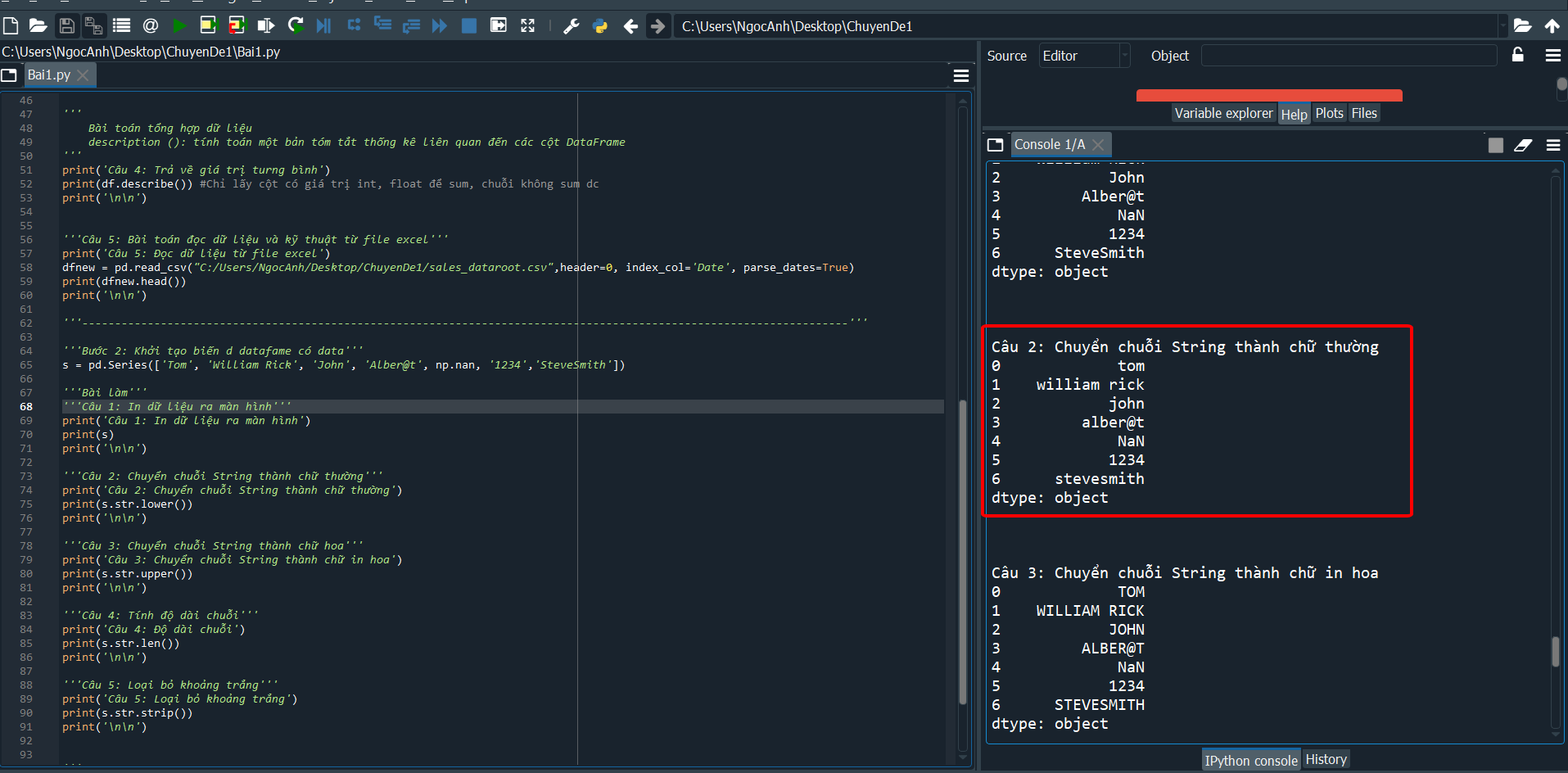
### **lower() :**

**import** pandas **as** pd

**import** numpy **as** np

s = pd.Series(['Tom', 'William Rick', 'John', 'Alber@t', np.nan, '1234','SteveSmith'])

**print** s.str.lower()



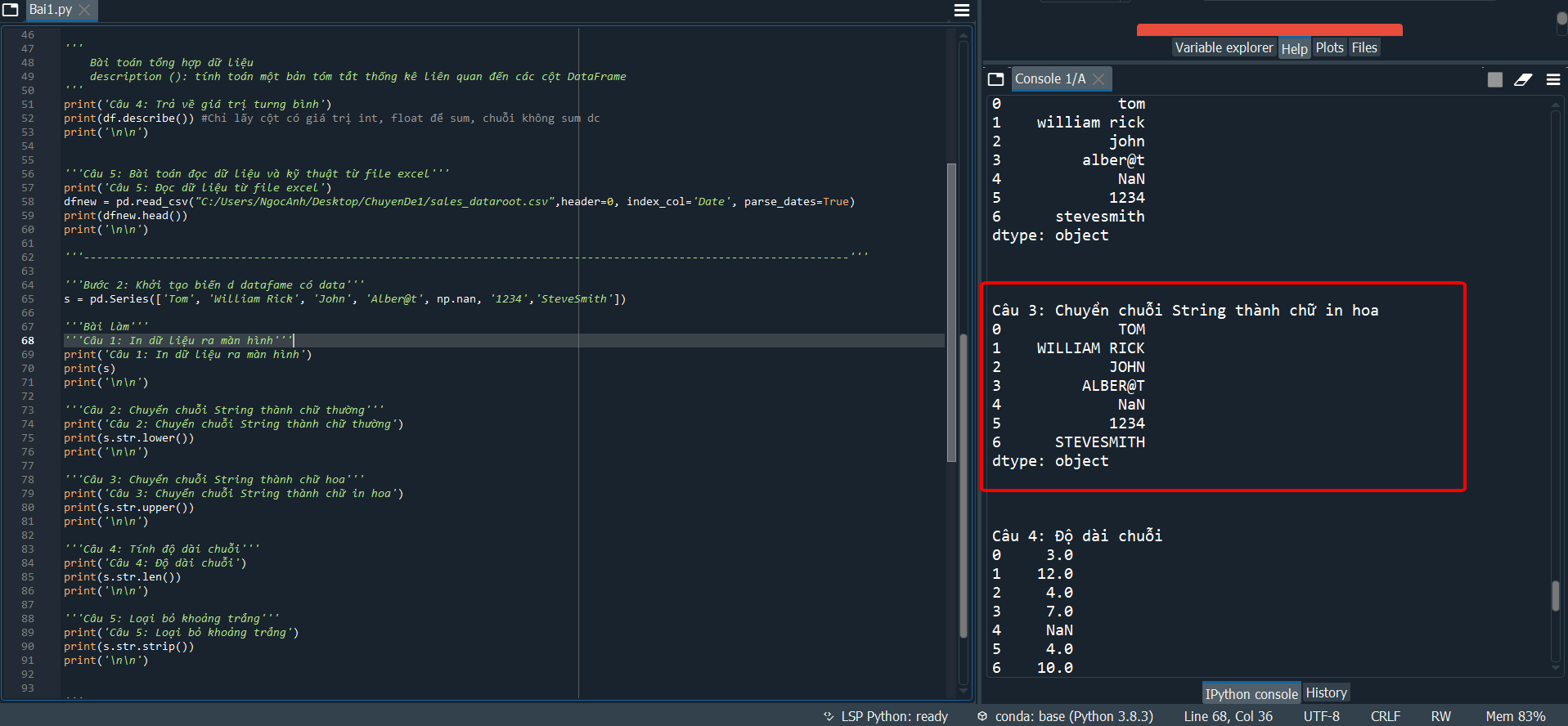
### **upper()**

**import** pandas **as** pd

**import** numpy **as** np

s = pd.Series(['Tom', 'William Rick', 'John', 'Alber@t', np.nan, '1234','SteveSmith'])

**print** s.str.upper()



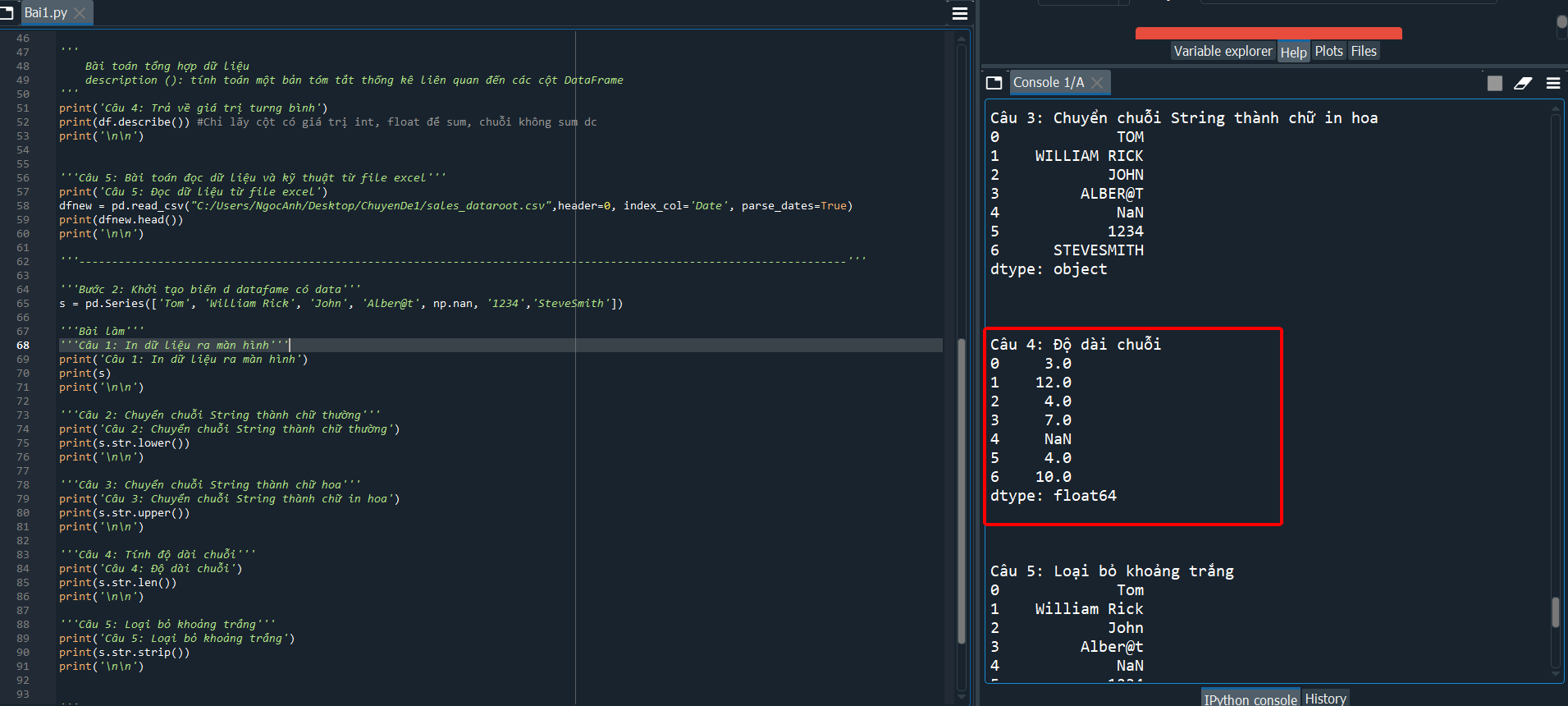
### **len()**

**import** pandas **as** pd

**import** numpy **as** np

s = pd.Series(['Tom', 'William Rick', 'John', 'Alber@t', np.nan, '1234','SteveSmith'])

**print** s.str.len()



### **strip()**

**import** pandas **as** pd

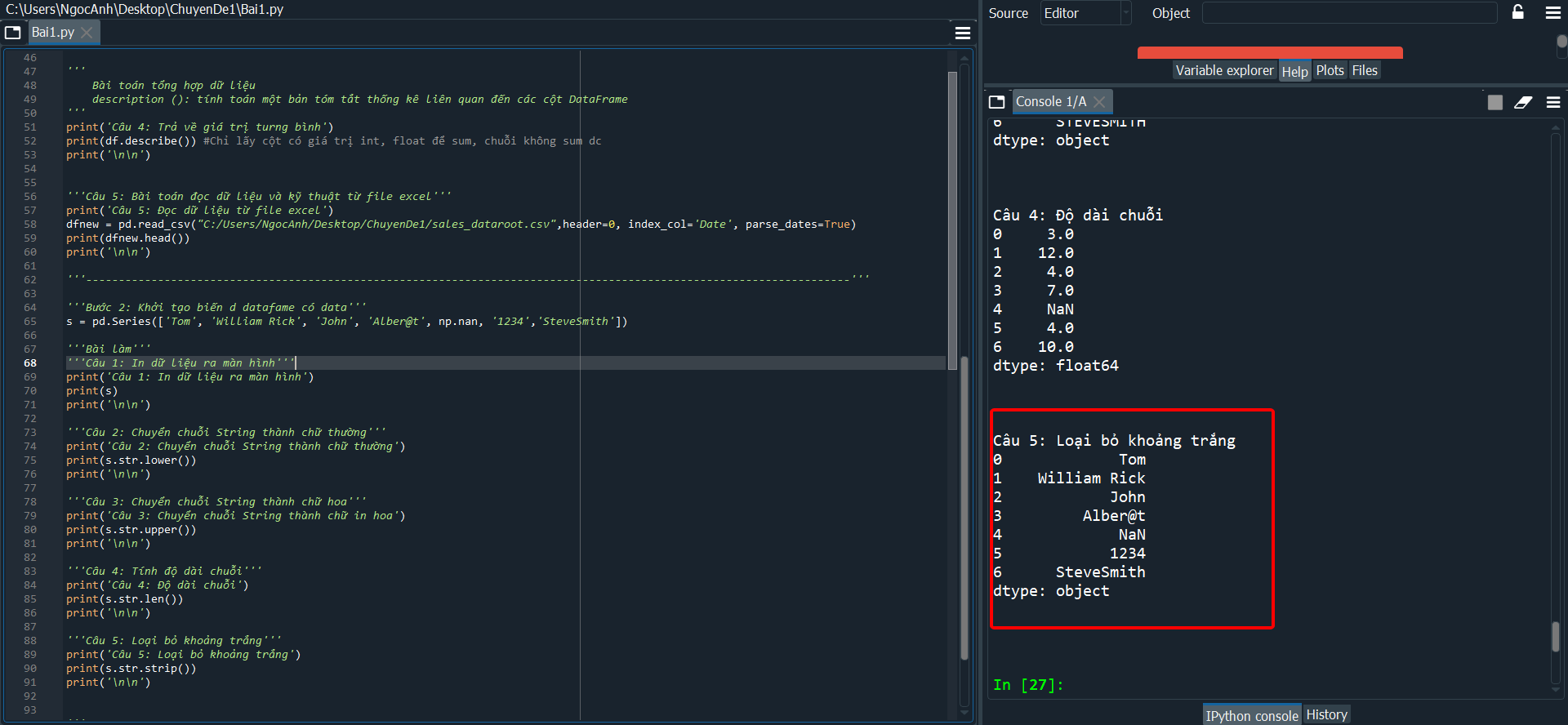
**import** numpy **as** np

s = pd.Series(['Tom ', ' William Rick', 'John', 'Alber@t'])

**print** s

**print** ("After Stripping:")

**print** s.str.strip()



###### **a. DataFrame là Dictionary**

Quay lại với ví dụ ở bài 2, ta có một DataFrame chứa các Series là dữ liệu dân số và diện tích của một số tỉnh / thành phố ở Việt Nam sau:

population = pd.Series({'TP.HCM': 8993, 'Hanoi': 8053, 'Lam Dong': 1297, 'Quang Tri': 623})

area = pd.Series({'TP.HCM': 2061, 'Hanoi': 3359, 'Lam Dong': 9765, 'Quang Tri': 4746})

data = pd.DataFrame({'Dân số': population, 'Diện tích': area})

data

Kết quả :

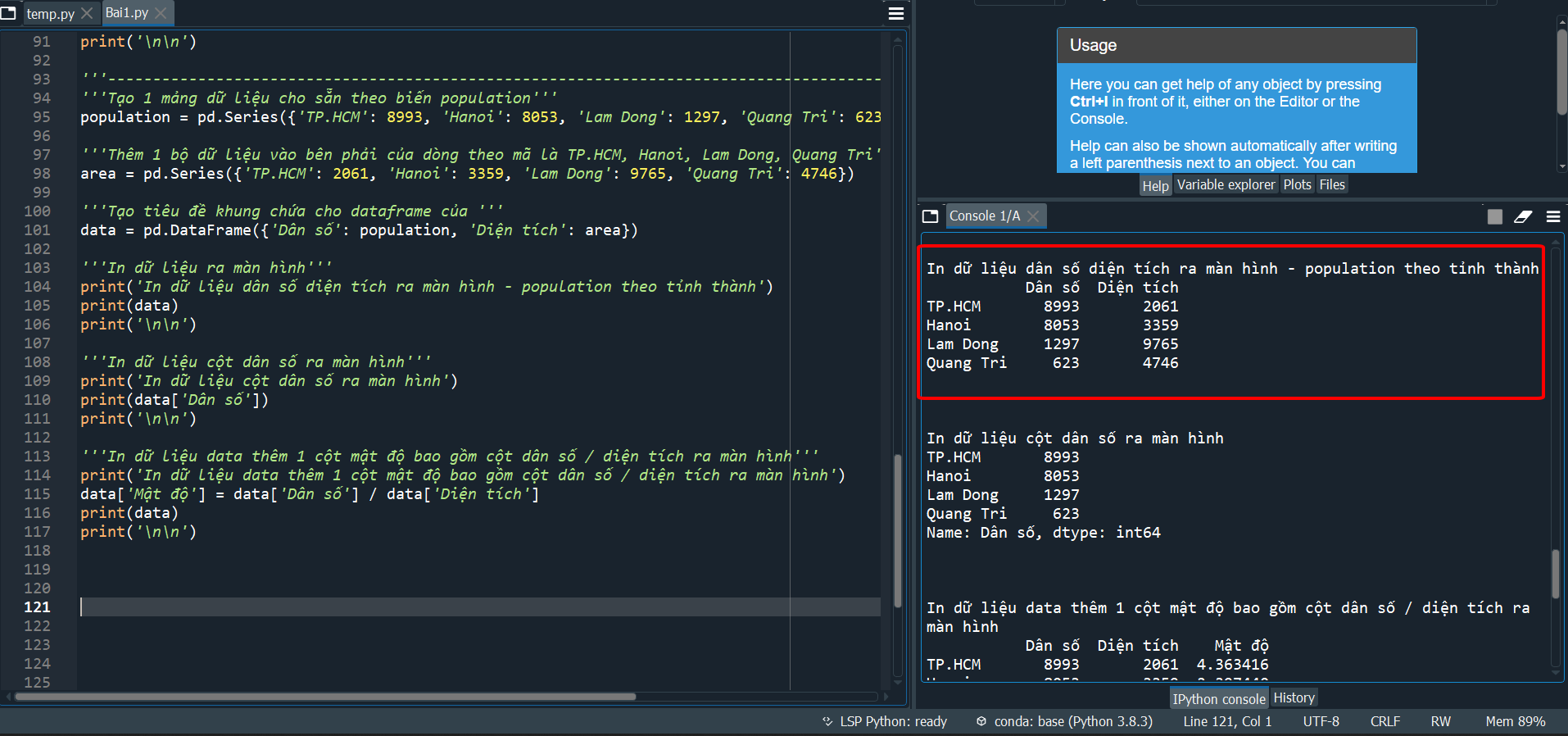
Dân số Diện tích

TP.HCM 8993 2061

Hanoi 8053 3359

Lam Dong 1297 9765

Quang Tri 623 4746



Từng Series trong DataFrame có thể được truy cập theo kiểu dictionary bằng cách index vào tên của cột tương ứng:

**data**['Dân số']

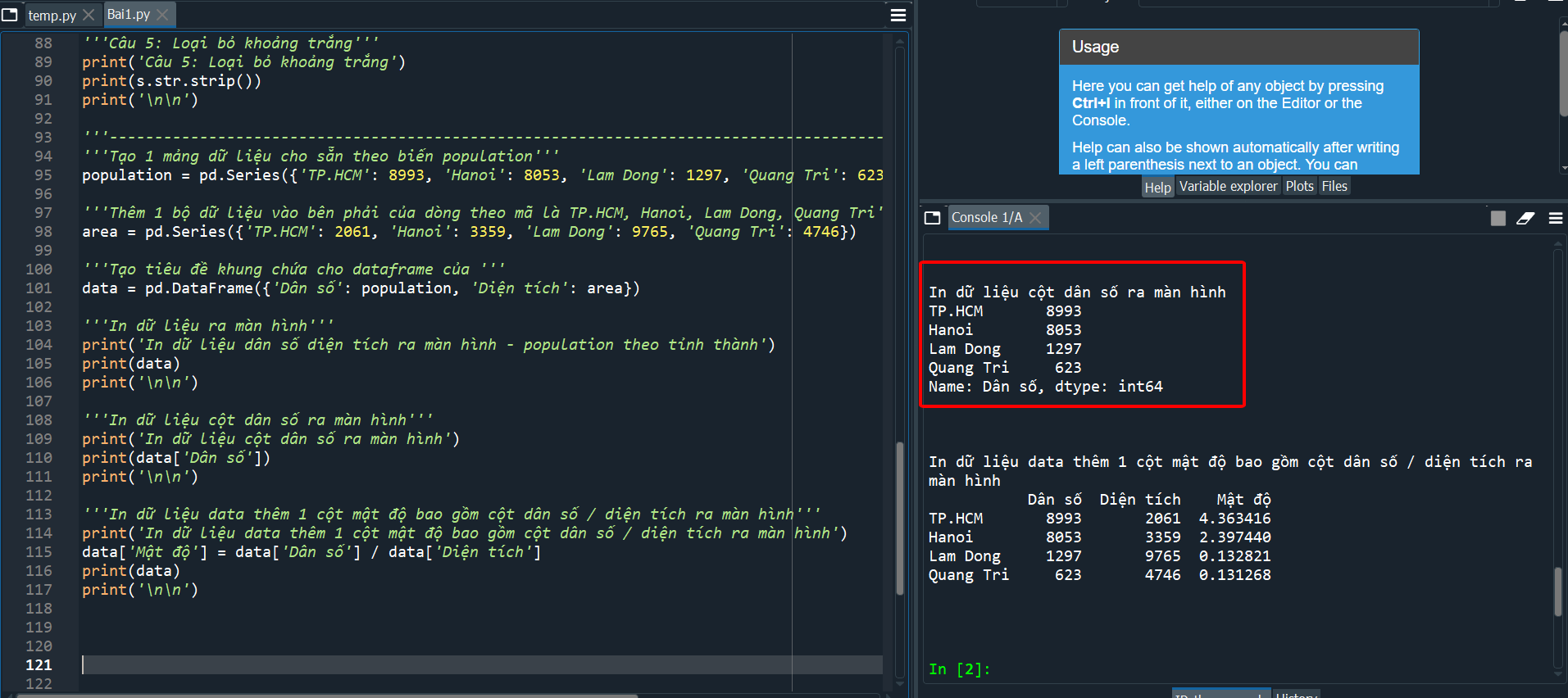
TP.HCM 8993

Hanoi 8053

Lam Dong 1297

Quang Tri 623

Name: Dân số, dtype: int64



Và ta cũng có thể thêm các cặp dữ liệu mới bằng cách tương tự như Series:

**data**['Mật độ'] = **data**['Dân số'] / **data**['Diện tích']

**data**

Kết quả :

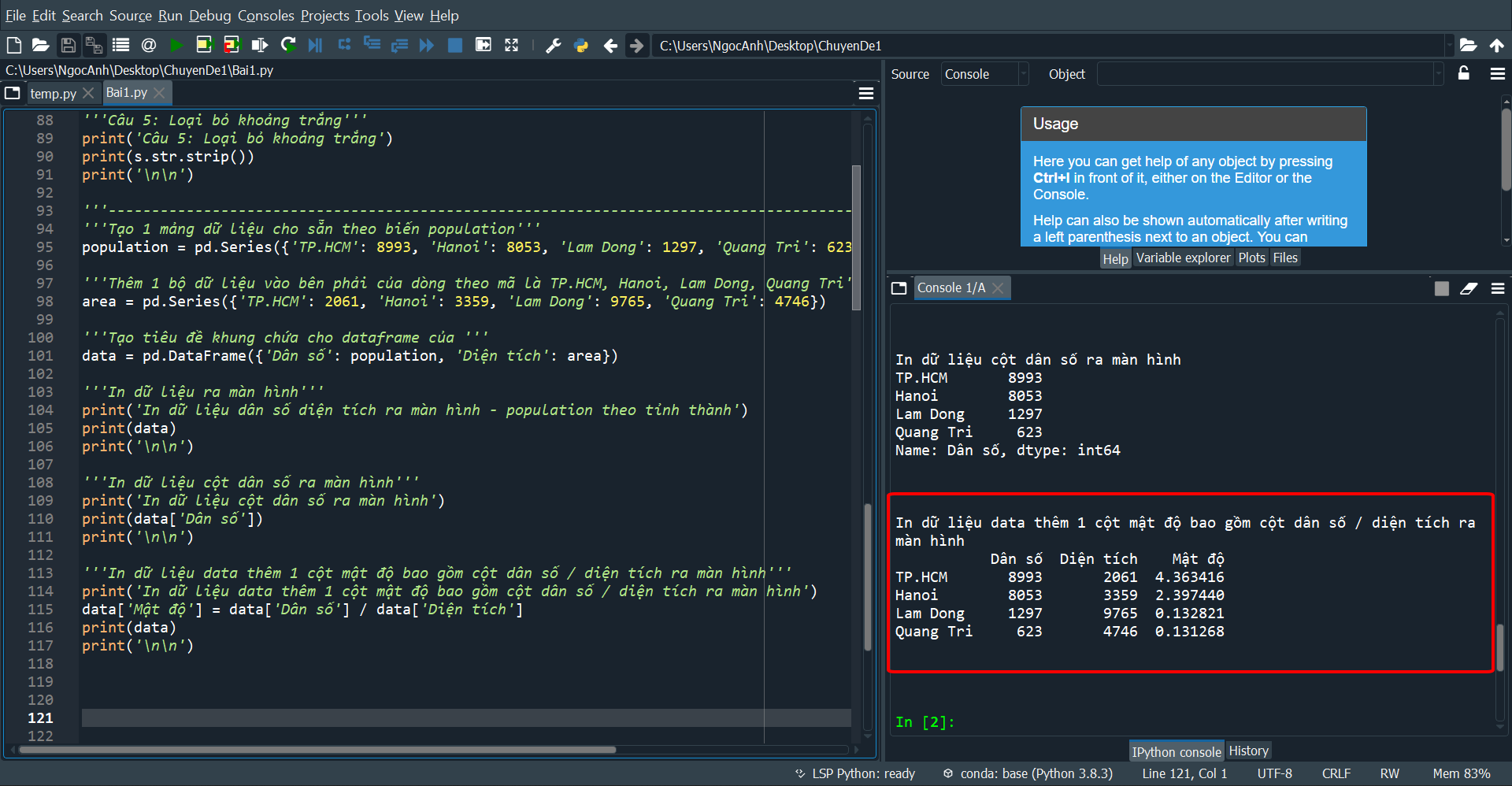
Dân số Diện tích Mật độ

TP.HCM 8993 2061 4.363416

Hanoi 8053 3359 2.397440

Lam Dong 1297 9765 0.132821

Quang Tri 623 4746 0.131268



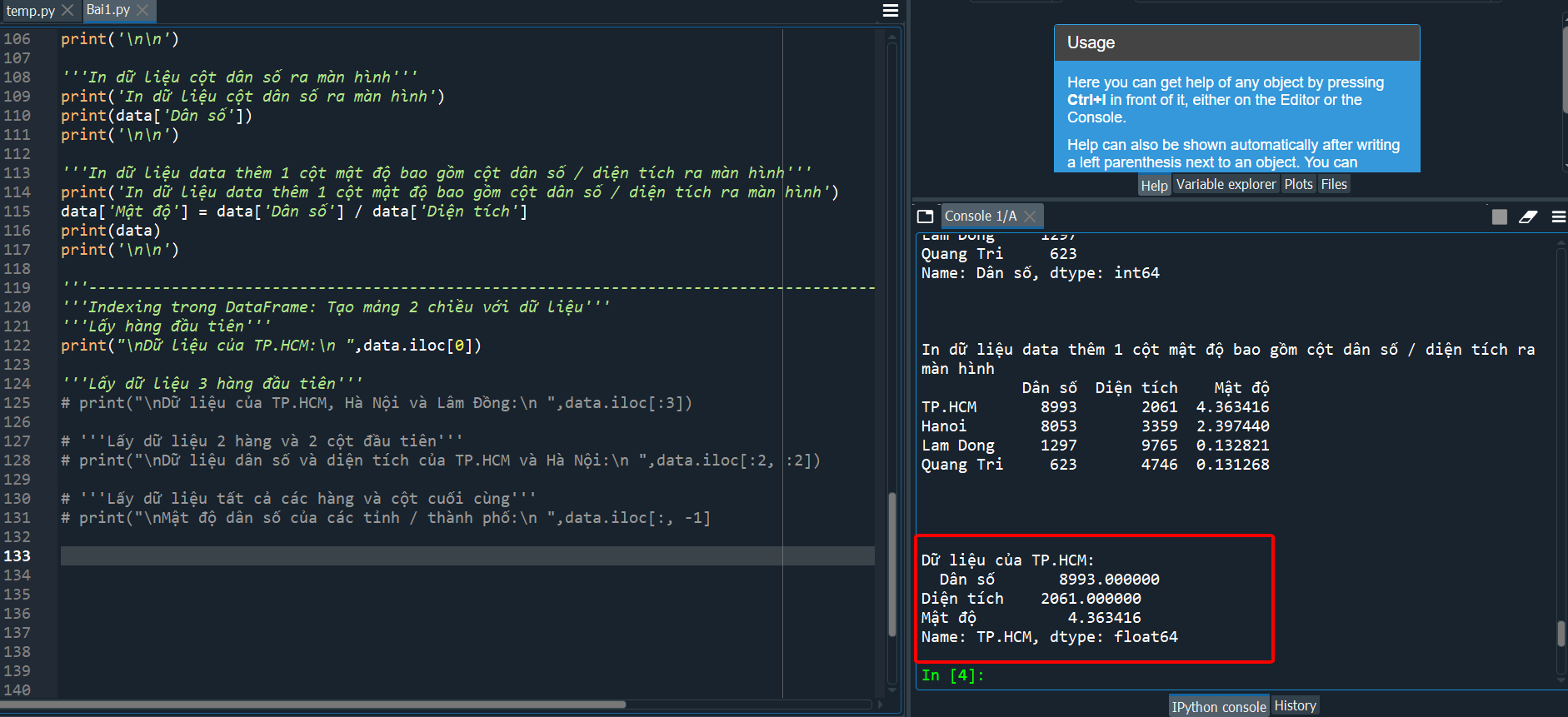
###### **c. Indexing trong DataFrame**

Giống với Series, ta sẽ sử dụng 2 thuộc tính là loc và iloc. Với iloc, ta sẽ thao tác không khác gì một mảng 2 chiều trong NumPy với các hàng và cột tương ứng vậy, chẳng hạn:

print(data)

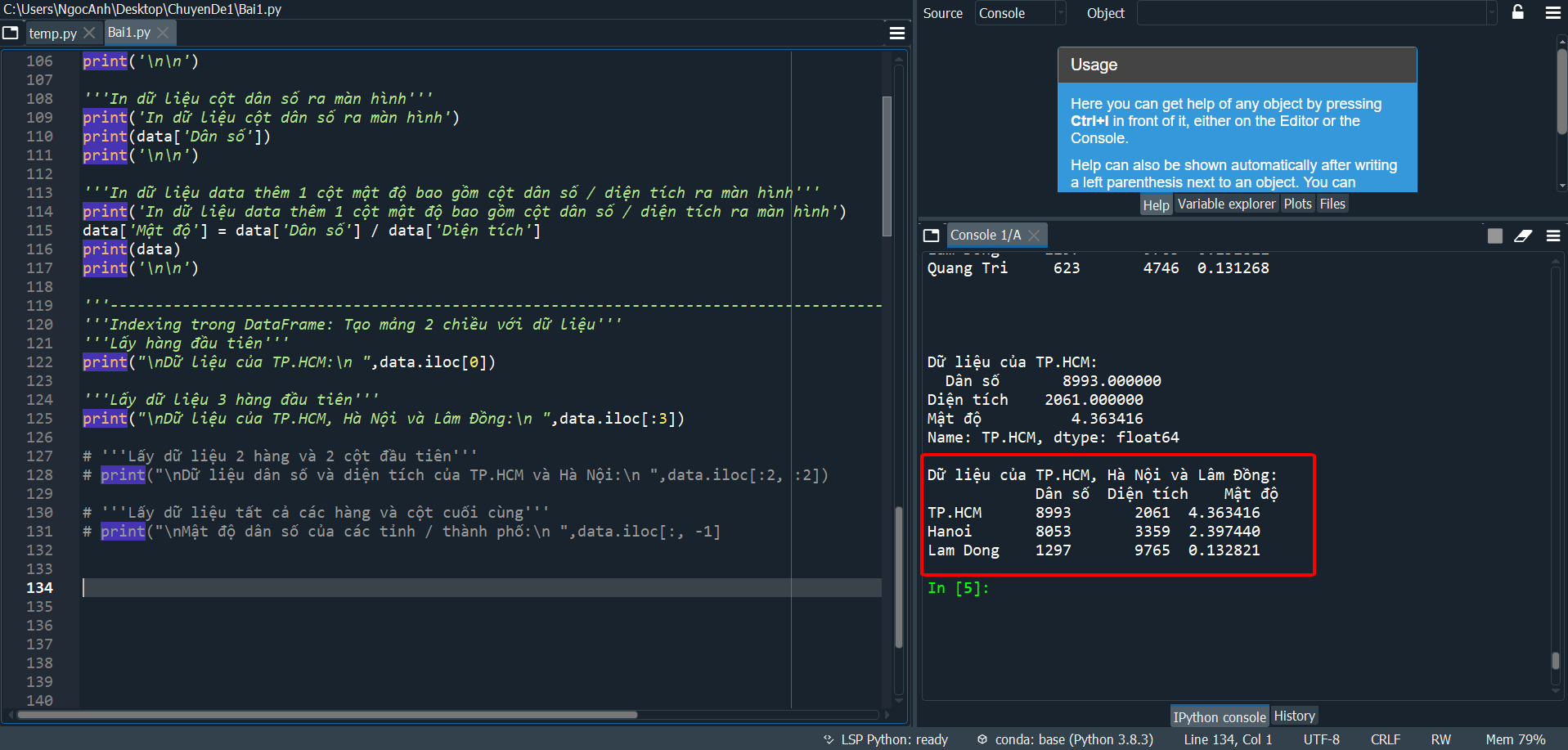
# Lấy hàng đầu tiên

print("\nDữ liệu của TP.HCM:\n ",data.iloc[0])



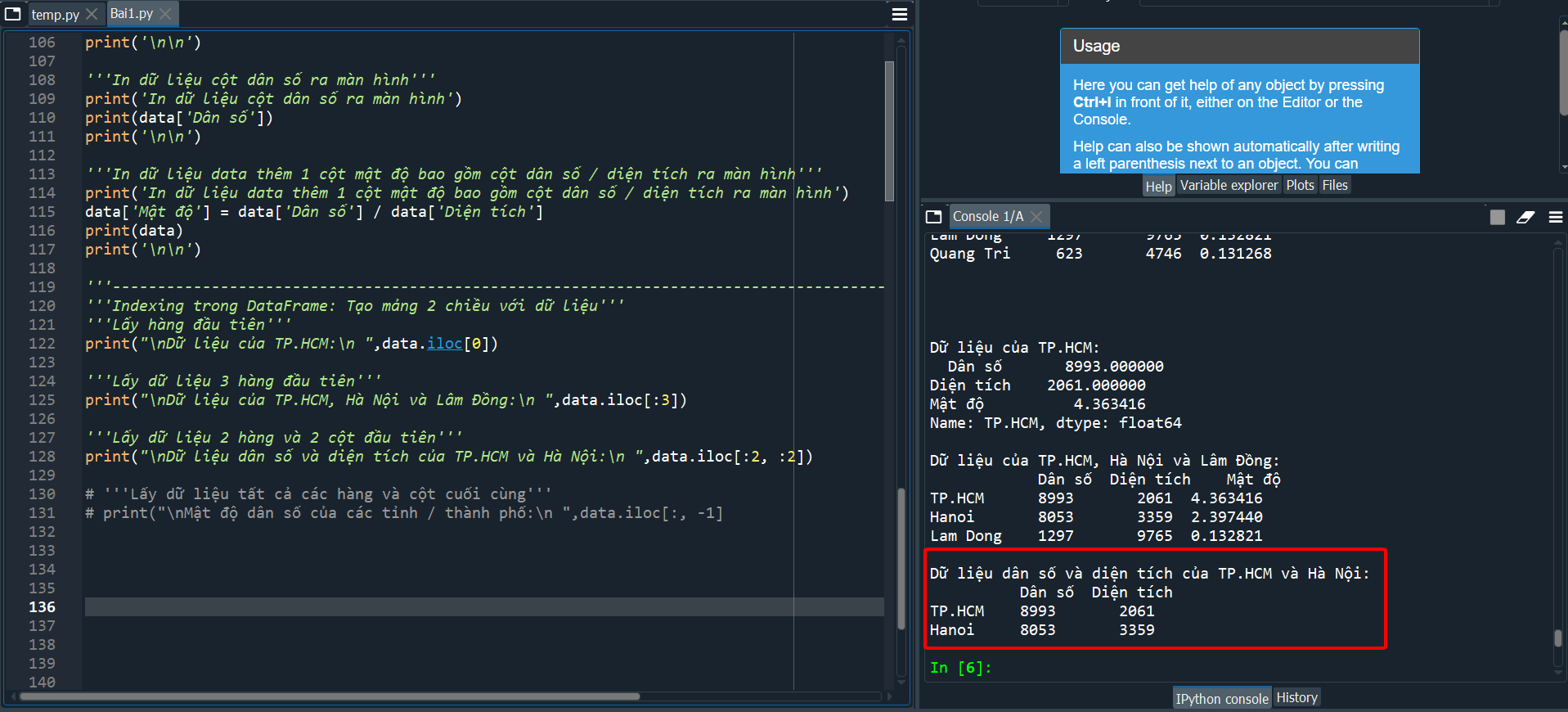
# Lấy dữ liệu 3 hàng đầu tiên

print("\nDữ liệu của TP.HCM, Hà Nội và Lâm Đồng:\n ",data.iloc[:3])



# Lấy dữ liệu 2 hàng và 2 cột đầu tiên

print("\nDữ liệu dân số và diện tích của TP.HCM và Hà Nội:\n ",data.iloc[:2, :2])



# Lấy dữ liệu tất cả các hàng và cột cuối cùng

print("\nMật độ dân số của các tỉnh / thành phố:\n ",data.iloc[:, -1]

