**Nguyễn Nguyên Hảo**

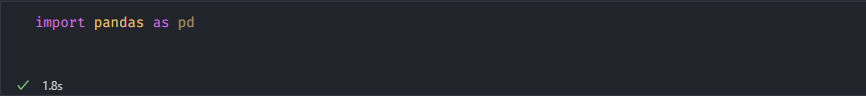
**Kiến thức về Pandas**

**Pandas là một công cụ rất mạnh cung cấp các tools để tính toán và phân tích dữ liệu. Hơn nữa, Pandas còn rất hiệu quả trong việc chuyển đổi định dạng dữ liệu và truy vấn dữ liệu từ các cơ sở dữ liệu. Hai cấu trúc dữ liệu chính của Pandas là Series và DataFrame.**

+ INSTALL PANDAS

Cú pháp *pip install pandas* nếu sử dụng anaconda setup môi trường thì chúng ta sử dụng cú pháp: *conda install pandas*

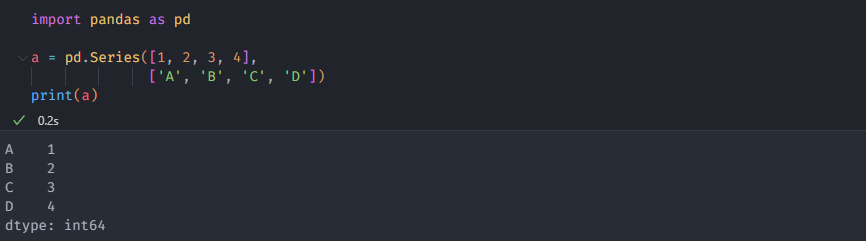
+ IMPORT MODULE



+ PANDAS SERIES

Một series trong padas là mảng 1 chiều được gán nhẵn. Bạn có thể tưởng tượng với cấu trúc dữ liệu dictionary trong python

Chúng ta sử dụng constructor của lớp Series. Tham số đầu tiên chúng ta truyền vào là 1 list các giá trị (value). Tham số thứ 2 là một list các key.

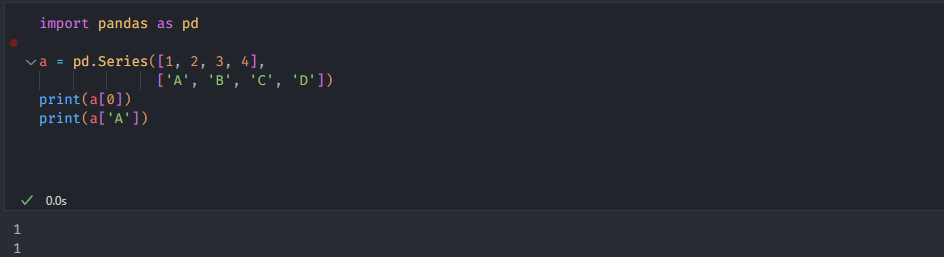


Cột đầu tiên là các key, cột thứ hai là các value tương ứng với các key.

+ACCESSING VALUES

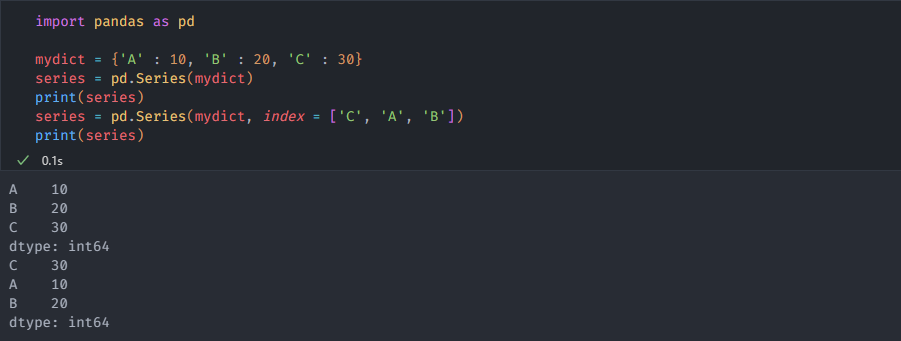
Việc truy cập vào các phần tử của series trong pandas giống với cách truy cập trong dictionary

Chúng ta cần sử dụng vị trí của các phần tử hoặc sử dụng key để truy cập vào các phần tử trong series



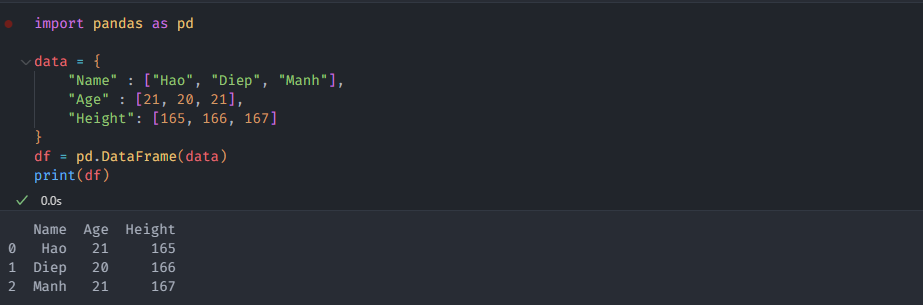
+ CONVERTING DICTIONARIES

Chúng ta có thể dễ dang convert dictionary trong Python sang Series trong Pandas



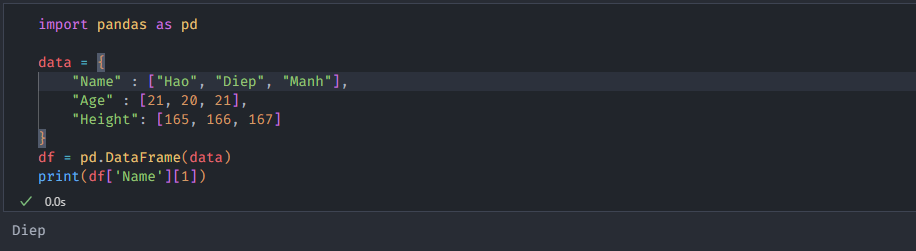
+ PANDAS DATA FRAME

Data Frame là một bảng nhiều. Chúng ta có thể tưởng tưởng với bảng trong Excel hay bảng trong database



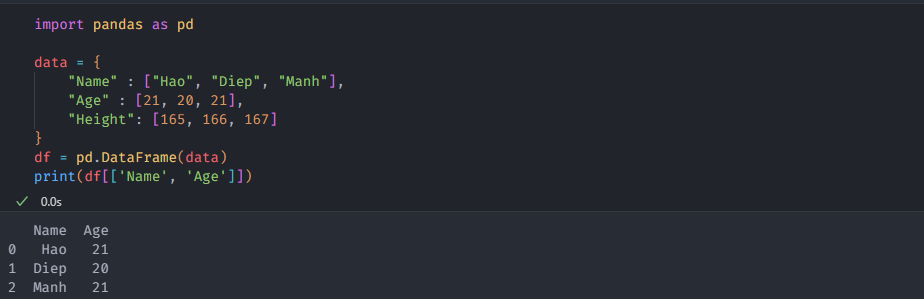
Để tạo data frame trong pandas, chúng ta sử dụng constructor của class. Trong ví dụ trên chúng ta tạo một dictionary với dữ liệu của 3 người sau đó đưa vào data frame.

Truy cập vào value:



Chúng ta cũng thể chọn in ra các cột khác nhau,

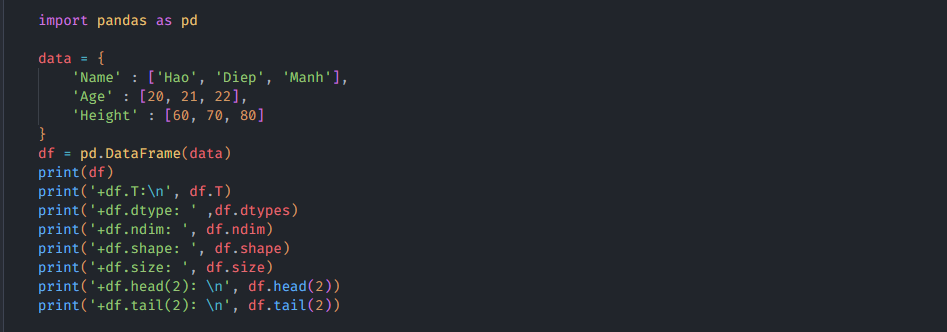
Tạo một list chứa “Name” và “Age” để truyền vào df

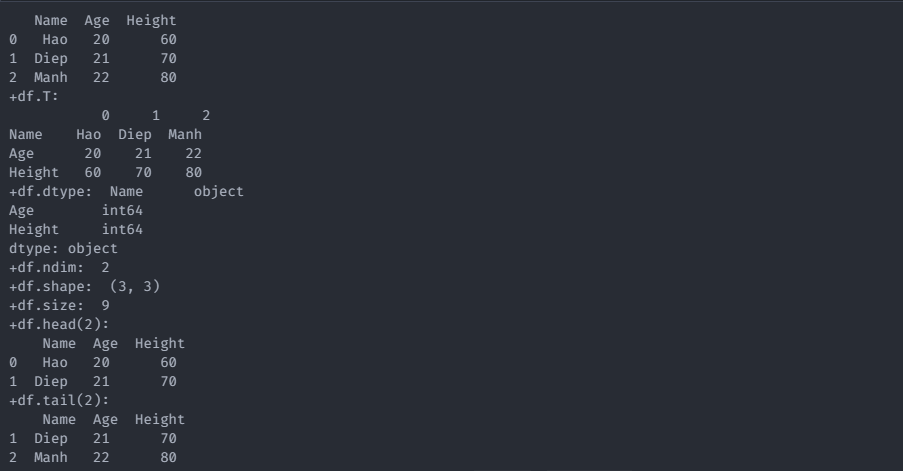


+ DATA FRAME FUCTION

BASIC FUNCTION AND ATTRIBUTE

|  |  |
| --- | --- |
| **Function** | **Description** |
| df.T | Đổi chỗ hàng và cột |
| Df.dtypes | Trả về kiểu dữ liệu của data frame |
| Df.ndim | Trả về số chiều của dataframe |
| Df.shape | Trả về hình dạng của data frame |
| Df.size | Trả về số lượng phần tử trong dataframe |
| Df.head(n) | Trả về n hàng đầu tiên của data frame  (mặc định là 5) |
| Df.tail(n) | Trả về n hàng cuối cùng của data frame (mặc định là 5 ) |





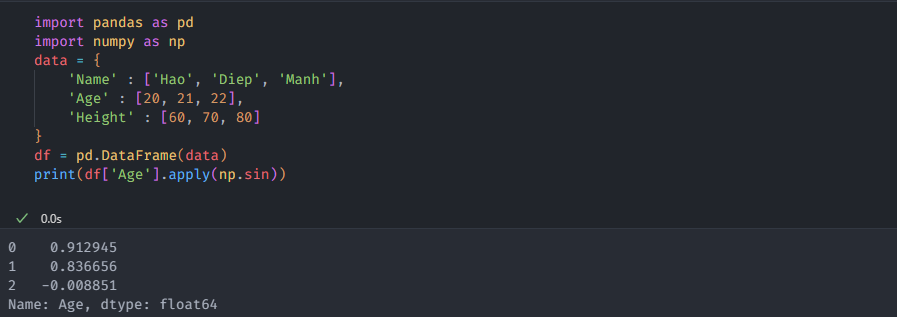
STATISTICAL FUNCTION

|  |  |
| --- | --- |
| **Function** | **Description** |
| Count() | Đếm số lượng phần tử không rỗng |
| Sum() | Trả về tổng của các giá trị của 1 cột được chọn |
| Mean() | Trả về giá trị trung bình cộng của 1 cột |
| Median() | Trả về giá trị trung vị của 1 cột |
| Mode() | Trả về giá trị xuất hiện nhiều nhất của 1 cột |
| Std() | Trả về đột lệch chuẩn của các giá trị |
| Min() | Trả về giá trị nhỏ nhất |
| Max() | Trả về giá trị lớn nhất |
| Abs() | Trả về trị tuyệt đối của các phần tử |
| Prod() | Trả về tích các phần tử được chọn |
| Describe() | Trả về dataframe với tất cả giá trị thống ke được tóm tắt |



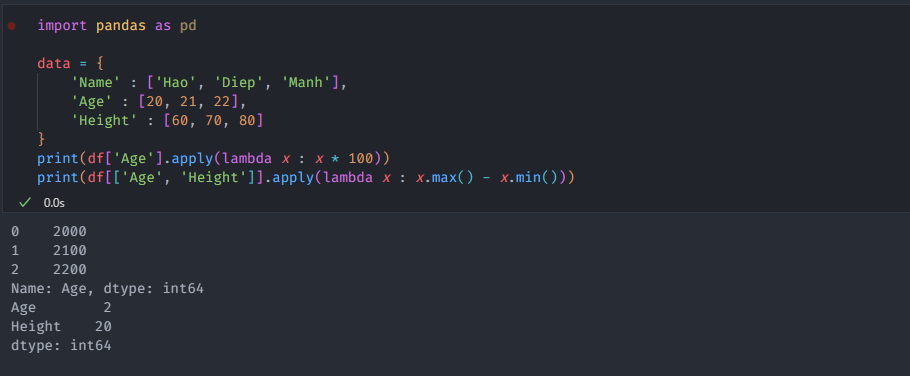
APPLYING NUMPY FUNCTION

Thay vì sử dụng những hàm built in trong Pandas, chúng ta cũng có thể sử dụng những phương thức mà chúng ta đã biết.



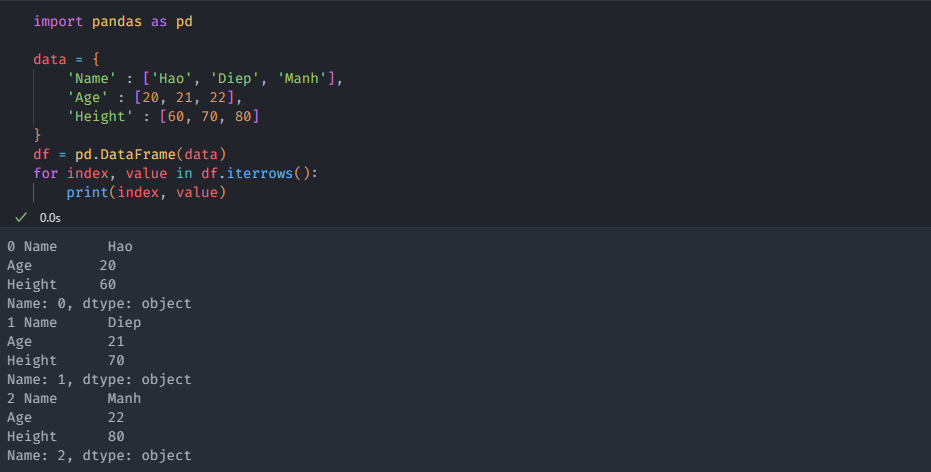
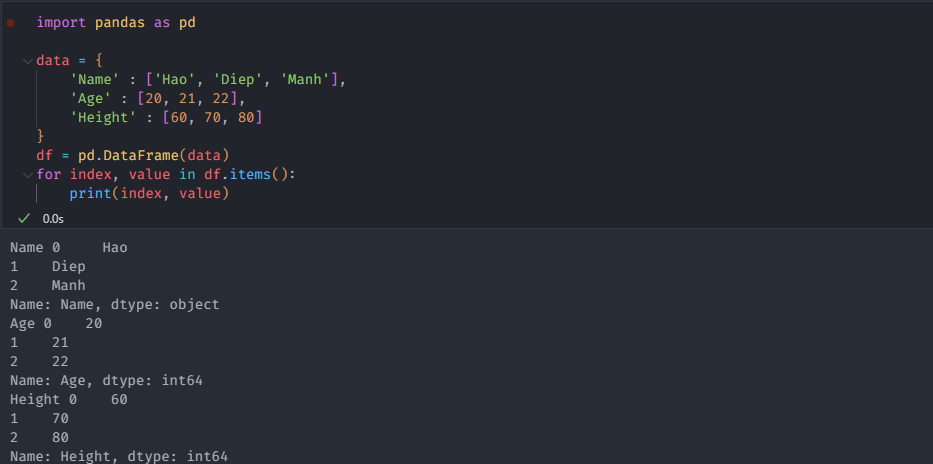
LAMBDA EXPRESSION

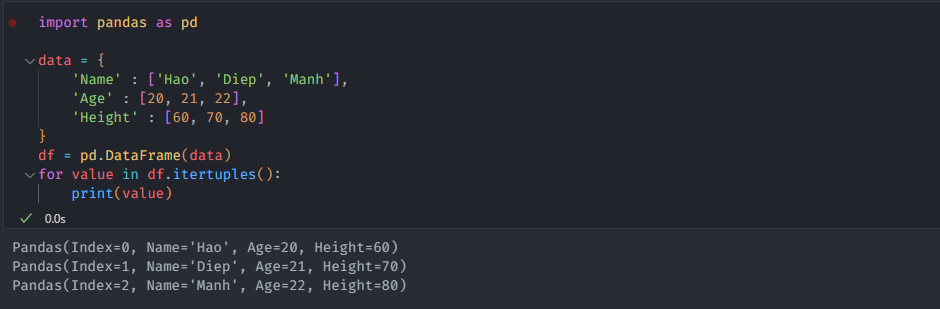
Một công cụ rất mạnh trong Python là lambda – hàm vô danh



+ ITERATING

|  |  |
| --- | --- |
| **Function** | **Description** |
| Iteritems() | Iterator for key-value pairs |
| Iterrows() | Iterator for the rows(index, series) |
| Itertuples() | Iterator for the row as named tuples |

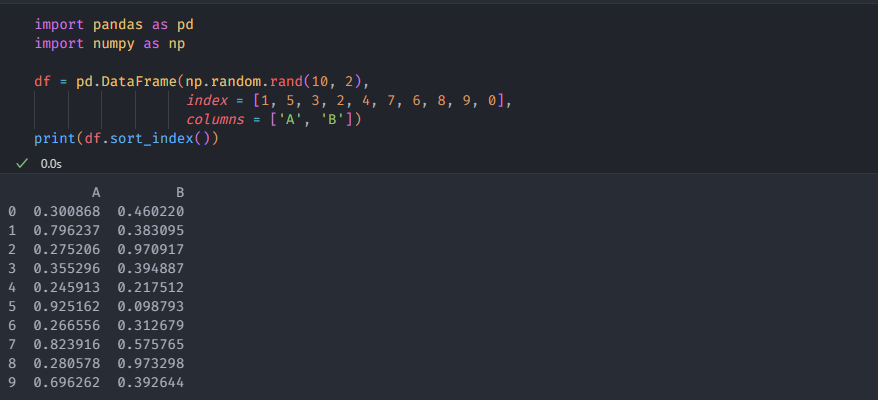




+ SORTING

Một điều rất mạnh của pandas dataframe là khả năng sắp xếp của chúng rất dễ dàng

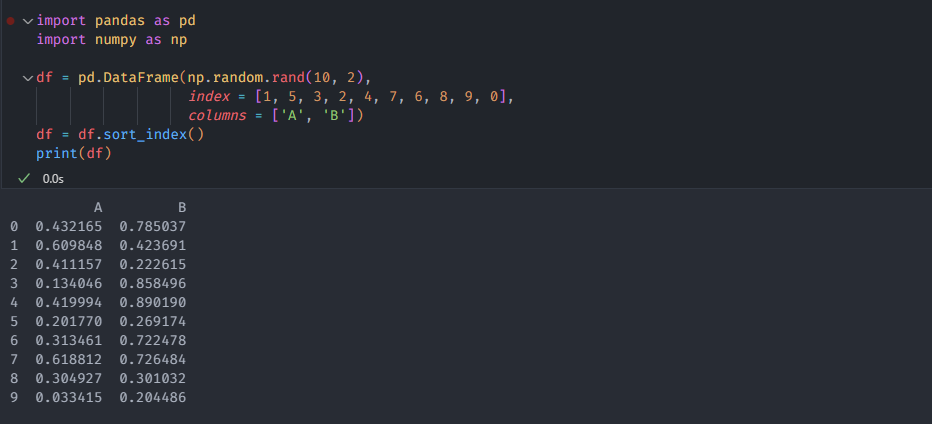
SORT BY INDEX



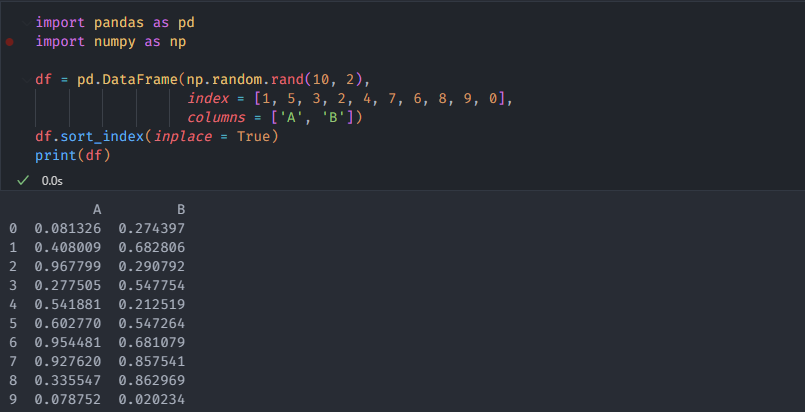
Phương thức sort\_index sẽ sắp xếp lại index của các cột

+ INPLACE PARAMETER

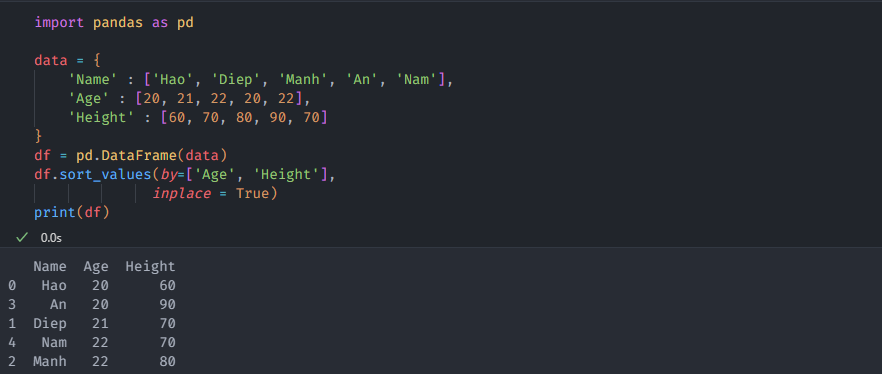
Khi sử dụng hàm sort\_index lên dataframe thì nó chỉ tạo một bản sao chứ không làm thay đổi mảng. Nếu chúng ta muốn thay đổi dataframe chúng ta có thể làm như sau:



Nhưng với pandas đưa cho chúng ta một sự lựa chọn tốt hơn, đó là sử dụng tham số inplace.Khi tham số này được đặt là True thì khi sử dụng hàm sort\_index sẽ thay đổi mảng ban đầu

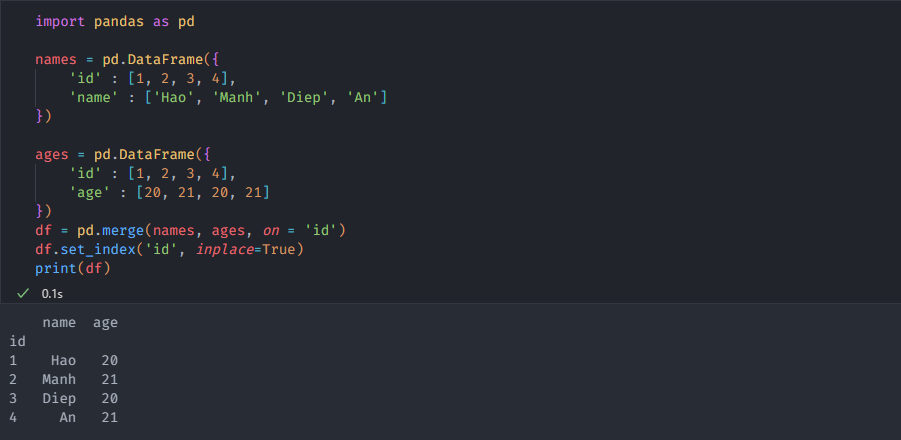


+ SORTING BY COLUMNS



Sử dụng hàm sort\_values để sắp xếp dataframe. Tham số by cho biết sẽ sắp xếp bởi cái gì. Trong trường hợp này, đầu tiên chúng ta sắp xếp theo tuổi, nếu 2 người có tuổi bằng nhau thì sẽ sắp xếp theo cân nặng

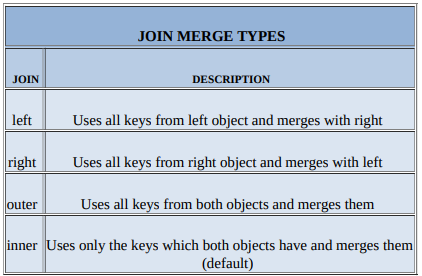
+JOINING AND MERGING

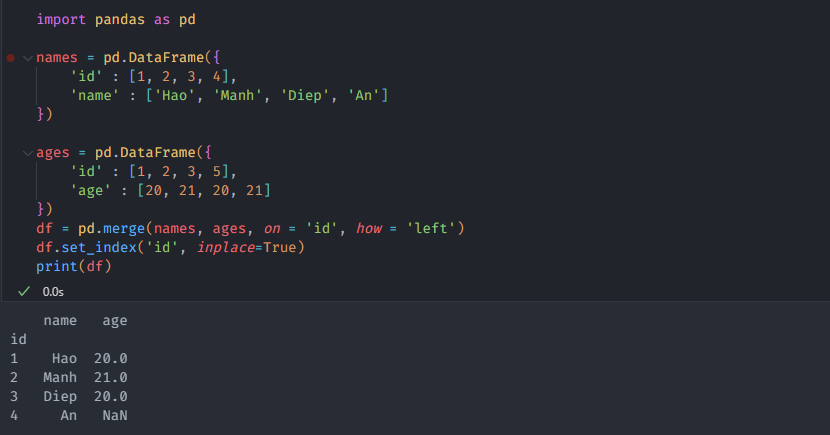


Trong ví dụ trên chúng ta có 2 dataframe. Để có thể merge 2 dataframe thành 1 thì chúng ta cần có một cột chung, trong trường hợp này là cột ‘id’.

Để set cột id thành cột index chúng ta sử dụng hàm set\_index(), tham số truyền vào là tên cột id và thêm tham số inplace = True để data frame thay đổi cả mảng ban đầu

+ JOIN

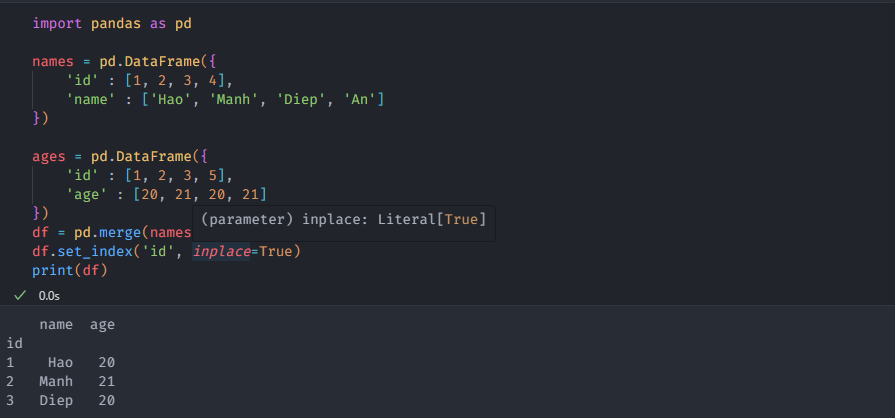




Đối với kiểu join left: Các hàng từ data frame ‘name’ sẽ được giữ lại, và các hàng phù hợp từ DataFrame ‘ages’ sẽ được thêm vào các vị trí có sẵn. Nếu không có vị trí phù hợp. NaN sẽ được chèn cho các cột từ DataFrame ‘ages’



Đối với kiểu join right: Tương tự như kiểu left tuy nhiên lần này DataFrame ‘ages’ sẽ được giữ lại



Đối với kiểu join inner: khi merge hai DataFrame chúng sẽ chọn ra nhưng id có ở cả hai DataFrame



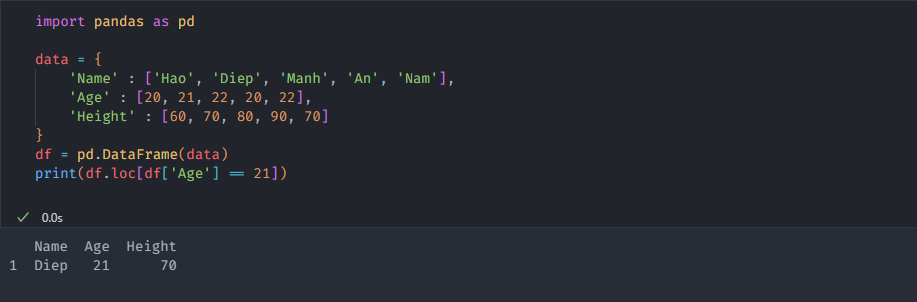
Đối với kiểu join outer: Tất cả các key từ cả 2 DataFrame sẽ được merge lại với nhau

+ QUERYING DATA

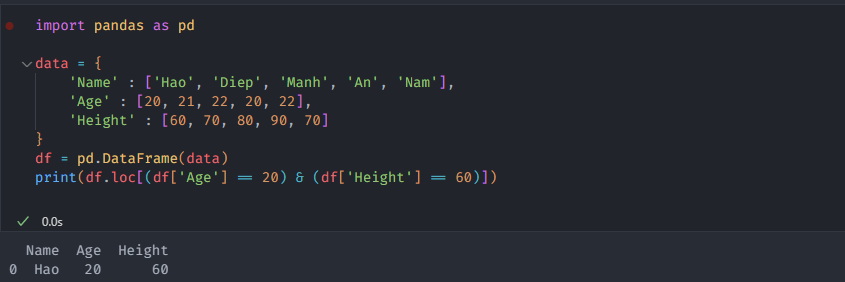
Như trong cơ sở dữ liệu của SQL, chúng ta có thể truy vấn dữ liệu từ các DataFrame trong Pandas. Để làm được việc này chúng ta sử dụng hàm ‘loc’, trong đó đặt biểu thức của mình

***Hàm loc sẽ trả về một DataFrame***

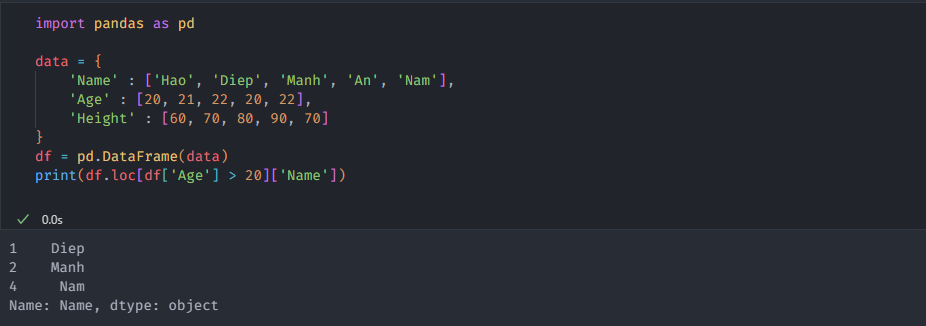
+ Liệt kê thông tin của những người có tuổi là 21



+ Liệt kê thông tin của người có tuổi là 20 và cân nặng 60 kg



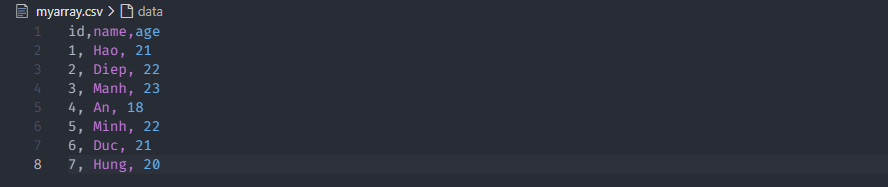
+Liệt kê TÊN của những người có tuổi lớn hơn 20



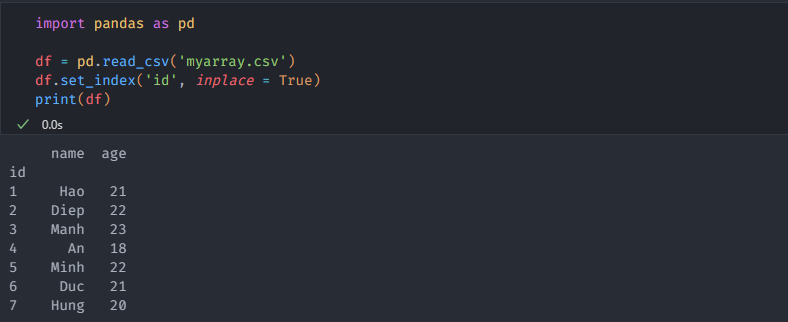
+ READING DATA FROM FILE

Giống với Numpy, Pandas cũng có thể đọc dữ liệu từ file

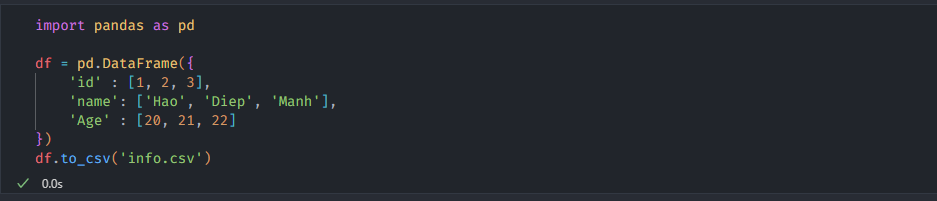
Giả sử có 1 file csv:

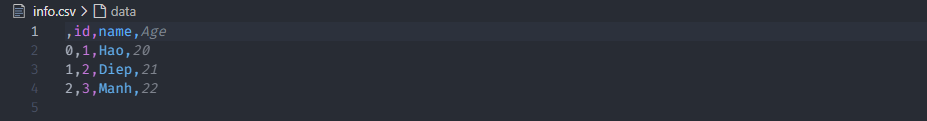


Đọc file .csv và in ra màn hình



+ Ghi vào file csv





+PLOTTING DATA

Vì pandas được xây dựng trên Matplotlib nên chúng ta có thể dễ dàng trực quan hóa dữ liệu từ DataFrame của chúng ta

