**Master Python for Data Analysis.**

**Learn the tools of the trade: Pandas, NumPy, Matplotlib, and Seaborn**

[**Predictive Data Analysis with Python**](https://www.educative.io/courses/predictive-data-analysis-with-python)

**Pandas import convention**

import pandas as pd

Pandas is now accessible with the acronym pd. You can also install Pandas using the built-in Python tool pip and run the following command.

$ pip install pandas

**Create and name a Series**

Create one-dimensional array to hold any data type. Invoke the pd.Series() method and then pass a list of values. Pandas will default count index from 0.

series1 = pd.Series([1,2,3,4]), index=['a', 'b', 'c', 'd'])

Set the Series name

srs.name = "Insert name"

Set index name.

srs.index.name = "Index name"

**Create a DataFrame**

Create a two-dimensional data structure with columns. Create and print a df.

df = pd.DataFrame(  
         {"a" : [1 ,2, 3],  
          "b" : [7, 8, 9],  
          "c" : [10, 11, 12]},        index = [1, 2, 3])

**Specify values in DataFrame columns**

Specify how you want to organize your DataFrame by columns.

df = pd.DataFrame(   
     [[1, 2, 3],   
     [4, 6, 8],  
     [10, 11, 12]],  
     index=[1, 2, 3],   
     columns=['a', 'b', 'c'])

**Read and Write to CSV file**

Open the CSV file, copy the data, paste it in our Notepad, and save it in the same directory that houses your Python scripts. Use read\_csv function build into Pandas and index it the way we want.

import pandas as pd  
data = pd.read\_csv('file.csv')  
  
data = pd.read\_csv("data.csv", index\_col=0)

**Read and write to**[**Excel**](https://www.educative.io/blog/excel-python-data-analytics)**file**

Call the read\_excel function to access an Excel file. Pass the name of the Excel file as an argument.

pd.read\_excel('file.xlsx')  
  
df.to\_excel('dir/myDataFrame.xlsx',  sheet\_name='Sheet2')

**Read and write to**[**SQL Query**](https://www.educative.io/blog/what-is-database-query-sql-nosql)

from sqlalchemy import create\_engine  
engine = create\_engine('sqlite:///:memory:')  
pd.read\_sql("SELECT \* FROM my\_table;", engine)  
pd.read\_sql\_table('my\_table', engine)  
pd.read\_sql\_query("SELECT \* FROM my\_table;", engine)

(read\_sql() is a convenience wrapper around read\_sql\_table() and read\_sql\_query())

df.to\_sql('myDf', engine)

**Get the first element of a Series**

Since Pandas indexes at 0, call the first element with ser[0].

import pandas as pd    
      
df = pd.read\_csv    
    
df['Name'].head(10)   
    
# get the first element   
ser[0]

**Get the first 5 elements of a Series**

Use ser[:n] to get the first n*n* elements of a Series.

import pandas as pd    
      
df = pd.read\_csv    
    
df['Name'].head(10)   
    
ser[:5]

**Get the last 5 elements in a Series**

Use ser[-n:] to get the last n*n* elements of a Series.

import pandas as pd    
      
df = pd.read\_csv   
    
df['Name'].head(10)   
    
ser[-5:]

**Select a single value position**

df.iloc[[0],[0]] 'Name'  
df.iat([0],[0]) 'Name'

**Select a single value by label**

df.loc[[0], ['Label']] 'Name'  
df.at([0], ['Label']) 'Name'

**Access a DataFrame with a boolean index**

In boolean indexing, we filter data with a boolean vector.

import pandas as pd   
     
# dictionary of lists   
dict = {'name':["name1", "name2", "name3", "name4"],   
        'degree': ["degree1", "degree2", "degree3", "degree4"],   
        'score':[1, 2, 3, 4]}   
     
df = pd.DataFrame(dict, index = [True, False, True, False])   
     
print(df)

**Drop values from rows**

s.drop(['a', 'c'])

**Drop values from columns**

df.drop('Value', axis=1)

**Create a new column in a DataFrame**

df['New Column'] = 0

**Keep the learning going.**

Learn Pandas and Data Analysis without scrubbing through videos or documentation. Educative’s text-based courses are easy to skim and feature live coding environments, making learning quick and efficient.

[**Predictive Data Analysis with Python**](https://www.educative.io/courses/predictive-data-analysis-with-python)

**Rename columns in a DataFrame**

df.columns = ['Column 1', 'Column 2', 'Column 3']

**Sort Series by labels along an axis**

Sort Series by index labels and returns a new Series sorted by the label if inplace argument is False, otherwise it updates the original series and returns None.

Series.sort\_index(self, axis=0, level=None, ascending=True, inplace=False, kind='quicksort', na\_position='last', sort\_remaining=True)

**Sort values along an axis (ascending order)**

df.sort\_values(by='Values')  
  
# descending order  
df.sort\_values(ascending = False)

**Adding ranks to particular entries**

Specify how you want to rank a column and add ranks.

df.rank()

**Retrieve rows and columns description**

df.shape

**Describe columns of DataFrame**

df.columns

**Retrieve index description**

df.index

**Get information on DataFrame**

df.info()

**Retrieve number of non-NA values**

df.count()

**Get sum of values**

df.sum()  
  
# cumulative sum  
  
df.cumsum()

**Subtract/Add 2 from all values**

s.sub(2)  
  
s.add(2)

**Multiply/Divide all values by 2**

s.mul(2)  
  
s.div(2)

**Find min/max values of a DataFrame**

df.min()  
  
df.max()

**Get min/max index values**

df.idxmin()  
  
df.idxmax()

**Get median or mean of values**

df.mean()  
  
df.median()

**Describe a summary of data statistics**

df.describe()

**Apply a function to a dataset**

f = # write function here  
df.apply(f)  
  
# apply a function by an element   
  
f = # write function here  
df.applymap(f)

**Merge two DataFrames**

pd.merge(df1, df2, on='subject\_id')

**Combine DataFrames across columns or rows: concatenation**

print(pd.concat([df1, df2]))