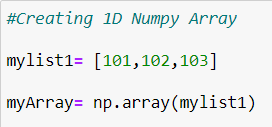
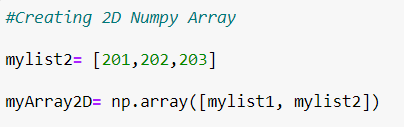
**NUMPY**

NumPy (Numerical Python) is an open source library of python which is used extensively in data science. It provides multidimensional array and matrix data structures. Numpy makes computation of numerical values easier and more structural.

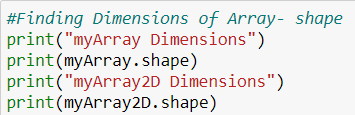
**Numpy Basics:**

Creating 2D array

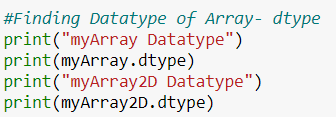




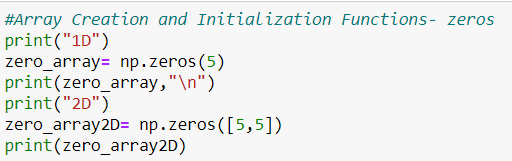
Finding dimensions of array using shape()

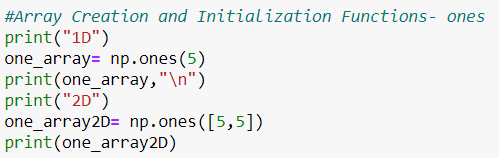


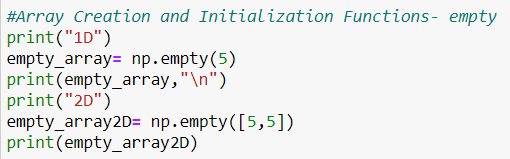
Finding the datatype of array using dtype()

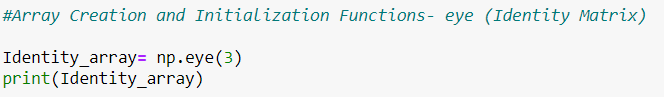


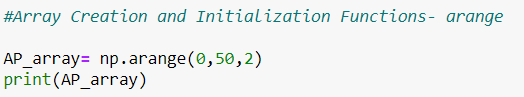
Creating and Initializing array using zeroes, ones, empty, identity and arange









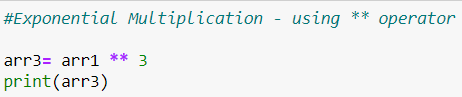


**Scalar Operations in Numpy:**

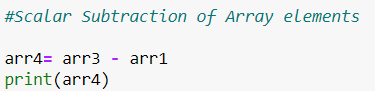
Scalar Array Multiplication



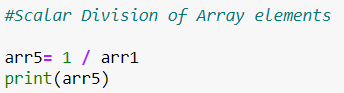
Exponential Multiplication



Scalar Subtraction

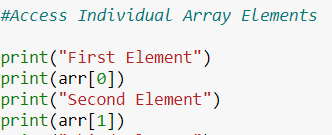


Scalar Division



**Numpy Array Indexes:**

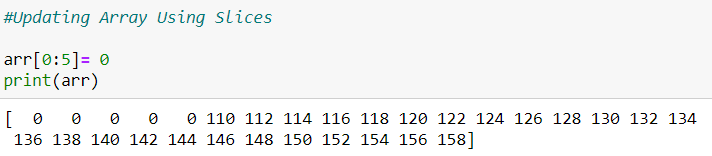
Accessing Individual Array Elements



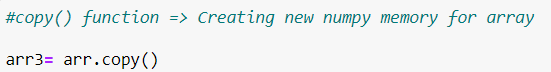
Slicing Array Indexes



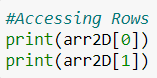
Updating Array using Slices



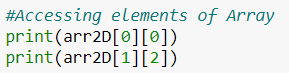
copy() function



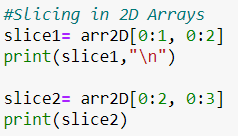
Accessing Rows



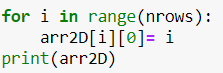
Accessing Elements of Array



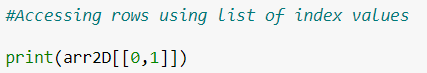
Slicing in 2D Array



Using Loops to Change Index

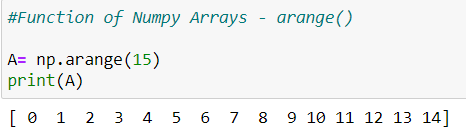


Accessing Rows using List of Index Values

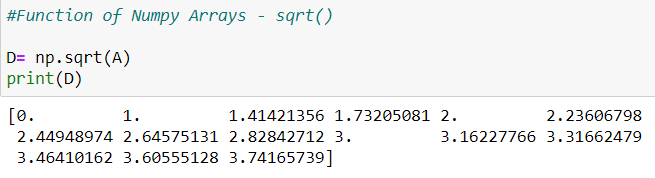


**Premium Array Operations:**

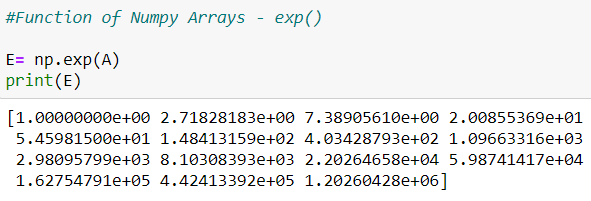
arange()



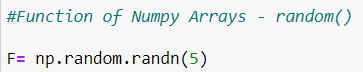
sqrt()



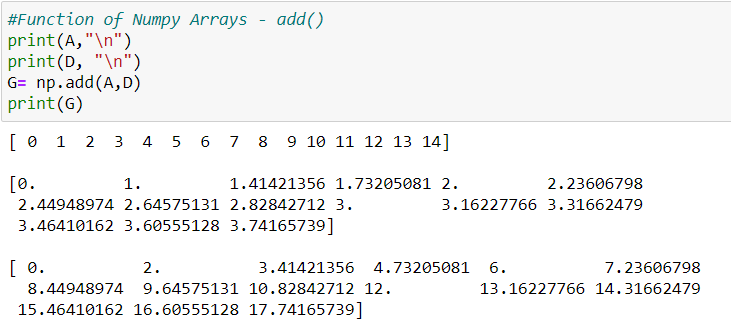
exp()



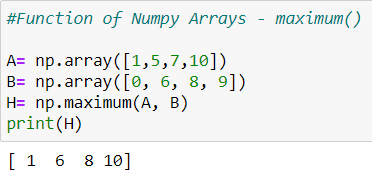
random()



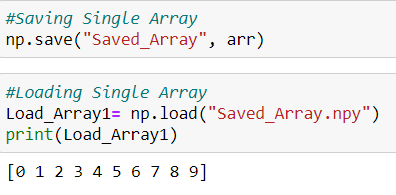
add()

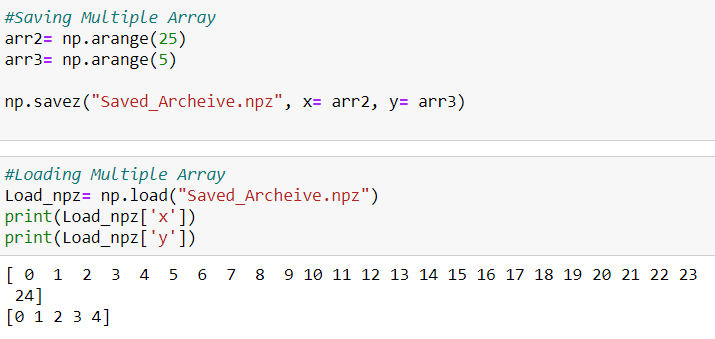


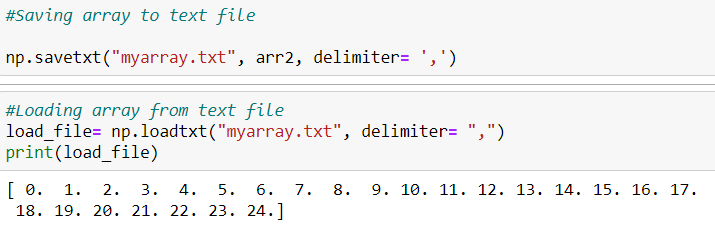
maximum()



**Loading and Saving Array:**

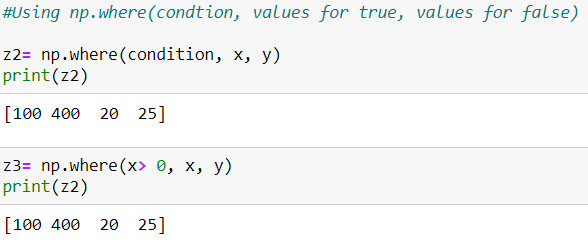




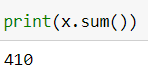


**Conditional Clauses and Boolean Operations:**

np.where()



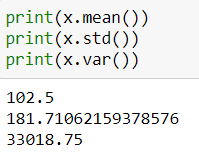
Sum



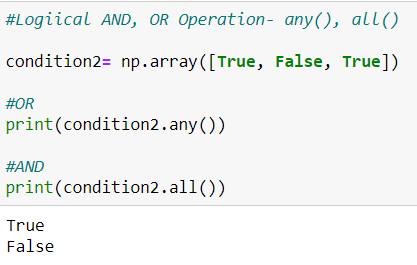
sum(0)



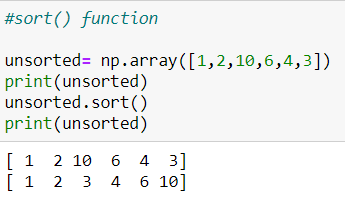
mean(), std(), var()



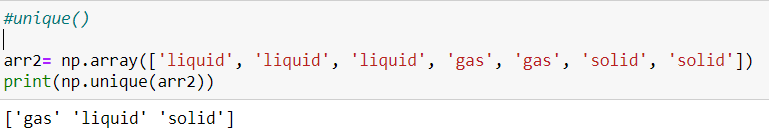
any(), all()



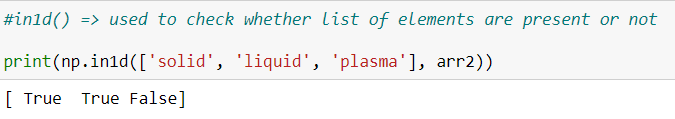
sort()



unique()

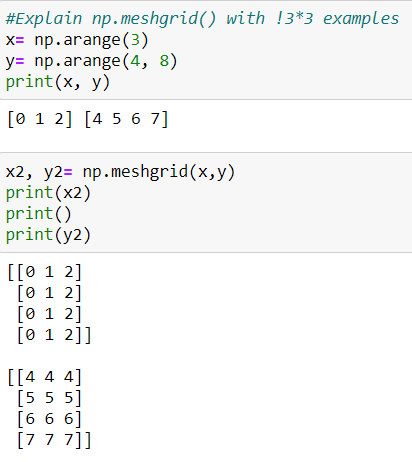


in1d()

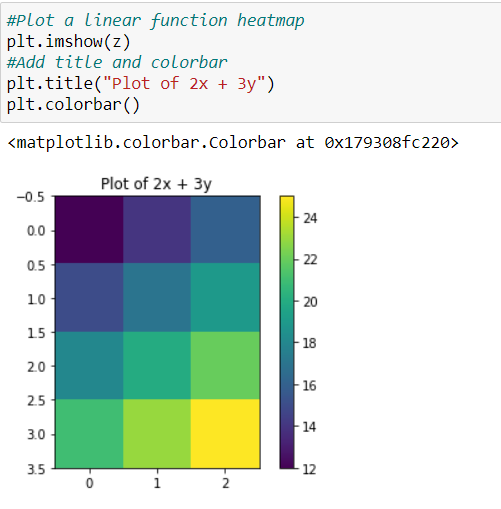


**Statistical Processing and Graphical Sketches:**

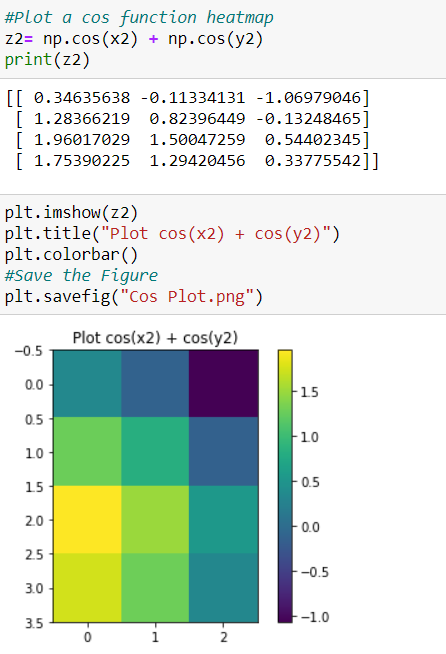
np.meshgrid()



Linear function heatmap along with title and colorbar



Cos function heatmap along with title and colorbar

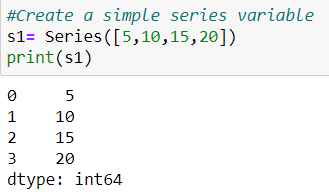


**PANDAS**

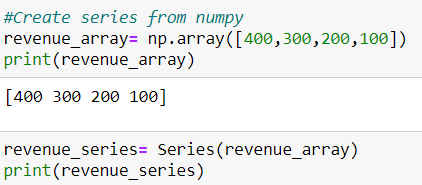
Pandas is helpful while working with tabular data. It is used for exploring, cleaning and processing given data where a data table is called a DataFrame. Pandas supports the integration with many file formats.

**Getting Started: Series**

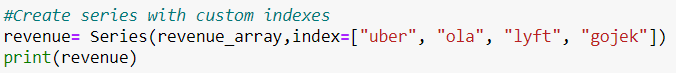
Create a simple series variable



Create series from numpy



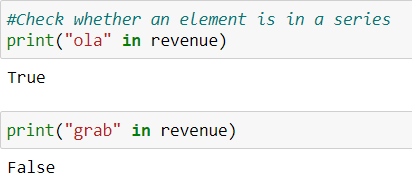
Create series with custom indexes



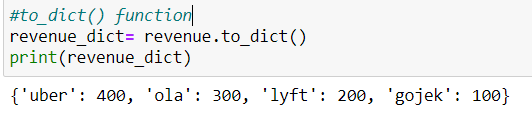
Filter a Series based on conditions



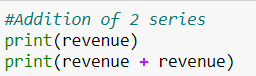
Check whether an element is in a series



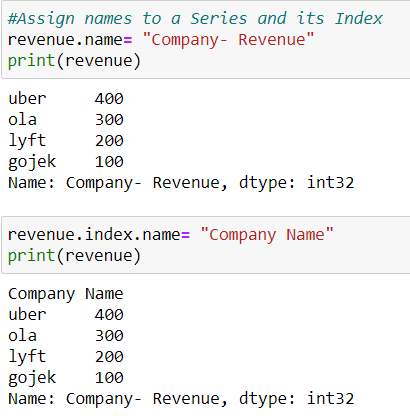
to\_dict() function



Addition of 2 series



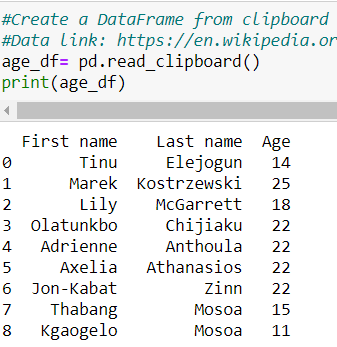
Assign names to a Series and its Index



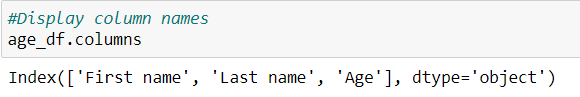
**Pandas DataFrame:**

* 2-Dimensional with 3 principal components- Rows, Columns and Values
* Similar to a table in excel sheet
* Labelled axes
* Size mutable

Create a DataFrame from clipboard



Display column names

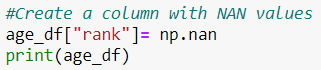


Access one or more columns





Create a column with NAN values

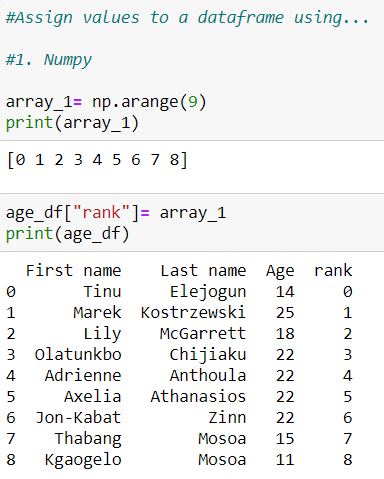


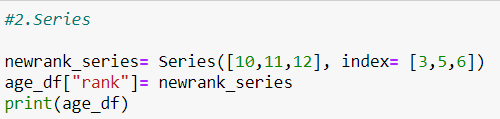
Head and Tail functions



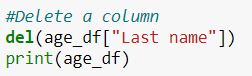


Assign values to a dataframe using 1. Numpy 2. Series

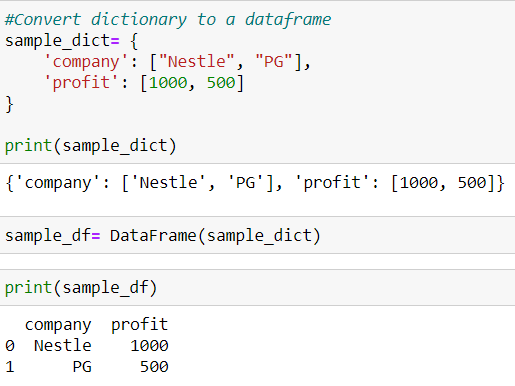




Delete a column

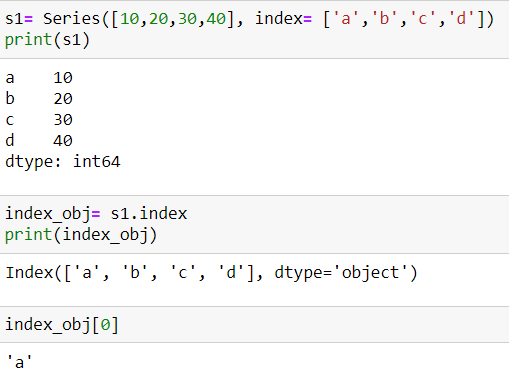


Convert dictionary to a dataframe

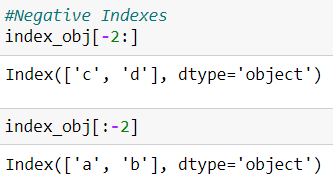


**Indexes in Pandas:**

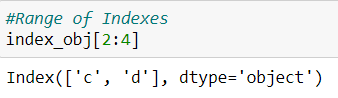
Index Array



Negative Indexes



Range of Indexes



**Reindexing in Pandas Series and DataFrames:**

Reindexing in Series- reindex() method



Reindexing in Series- reindex() method with fill\_value



Reindexing in Series- forwardfill



Reindexing in DataFrame

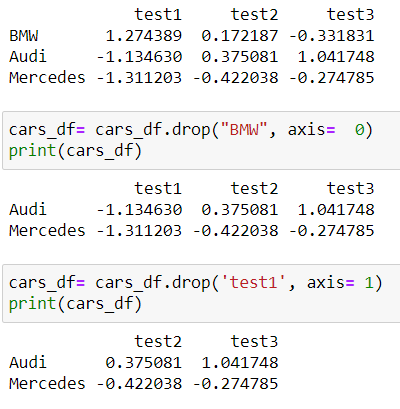


**Dropping Entries in Pandas Series and DataFrames:**

Drop values from Series

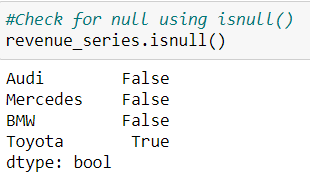


Drop Rows and Columns from DataFrame



**Handling Null or NAN values in Pandas:**

Check for null using isnull()



Series- dropna()



DataFrame- dropna()



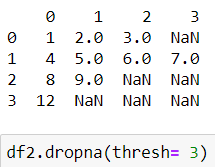
DataFrame- dropna() with how parameter



DataFrame- dropna() along column



Dropna with thresh parameter



fillna() function





**Selecting and Modifying Entries in Pandas:**

Access single element of series



Access multiple elements of series



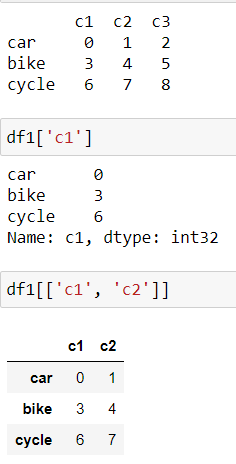
Using Numerical Indexes



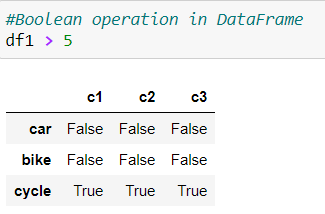
Conditional Indexes



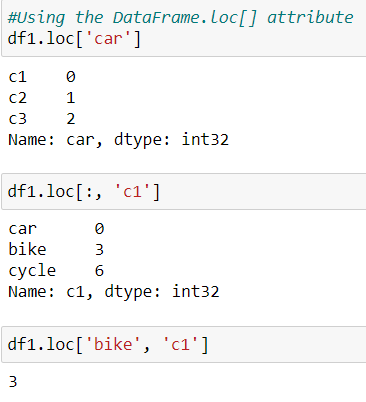
Accessing one or more column data from a DataFrame



Boolean operation in DataFrame

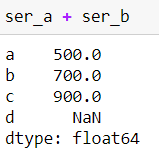


Using the DataFrame.loc[] attribute

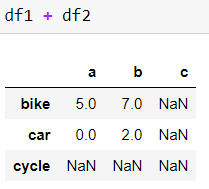


**Coordinate and Regulate Data in Pandas:**

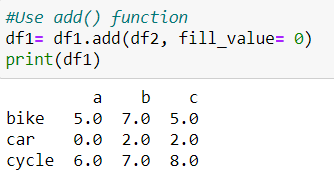
Add 2 Series



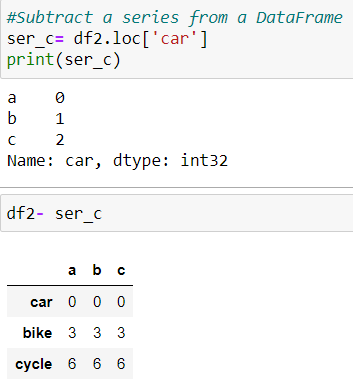
Add 2 DataFrames



Use add() function



Subtract a series from a DataFrame



**Ranking and Sorting in Pandas Series:**

Sorting by Index



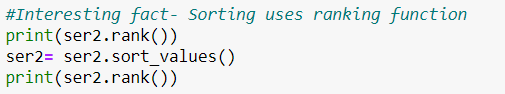
Sorting by Values



Ranking of Series

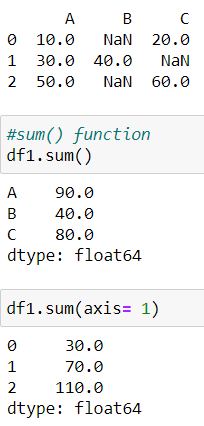


Interesting fact- Sorting uses ranking function

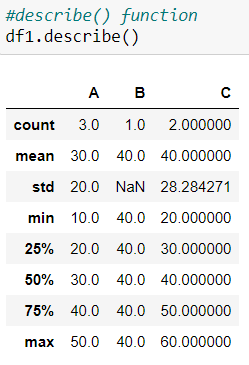


**Statistics and Graphs in Pandas DataFrame:**

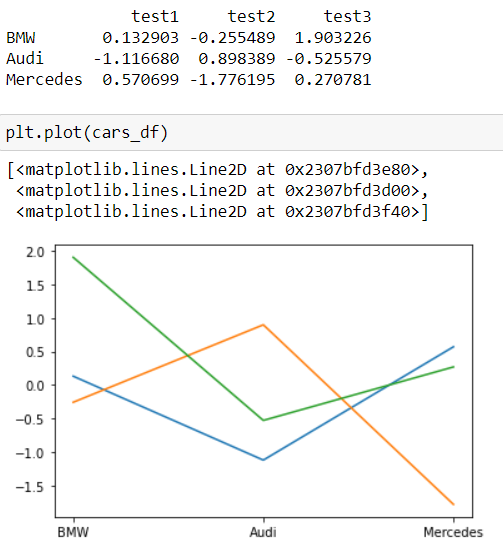
sum() function



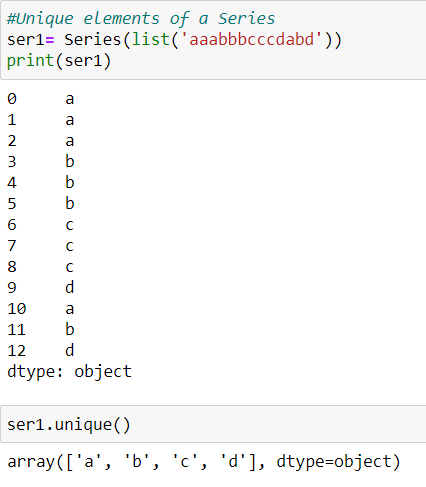
describe() function



Graph of DataFrame



Unique elements of a Series



Frequency of elements in a Series

