**General Purpose of NumPy**

* Numerical Python is a library for working with numerical data in Python.
* Contains multidimensional array and matrix data structures and a n-dimensional array
* Uses to perform a variety of mathematical operations on arrays

**NumPy Arrays**

* Central data structure that contains a grid of values and info about the raw data

<https://predictivehacks.com/tips-about-numpy-arrays/>

All other images from: <https://jakevdp.github.io/PythonDataScienceHandbook>

* Has a grid of elements that can be indexed in several ways:
  + by a tuple of nonnegative integers, by Booleans, by another array, or by integers
* Elements are all of the same type, (i.e., all integers, floats, text strings, etc.).

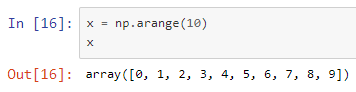
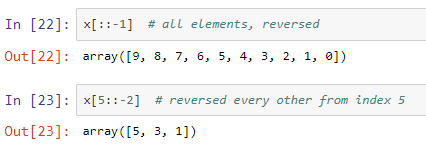
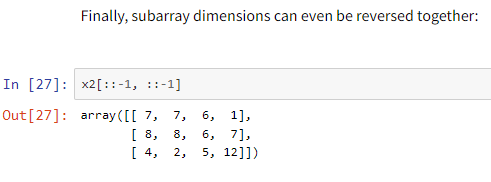
*Example 1D Array*: avg\_monthly\_precip = np.array([0.70, 0.75, 1.85])

*Example 2D Array*: precip\_2002\_2013 = np.array([[1.07, 0.44, 1.50], [0.27, 1.13, 1.72]])

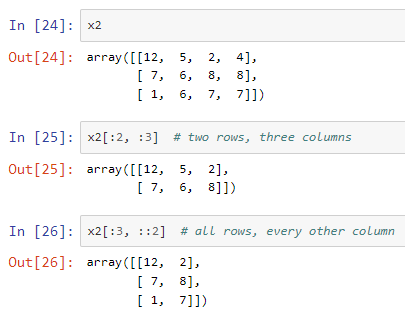
**Slicing NumPy Arrays**

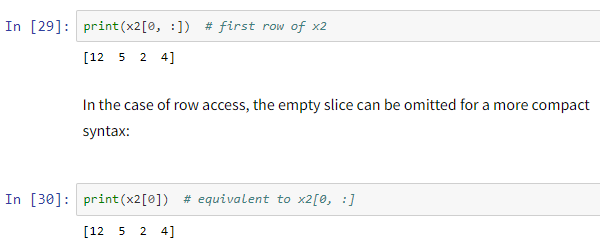
* For 1D arrays, you only need to specify **one index value**, which is the position of the element in the NumPy array (e.g. arrayname[index]).
  + *To get* ***third element****, use index value* ***2*** *(Python indexing begins with 0).*

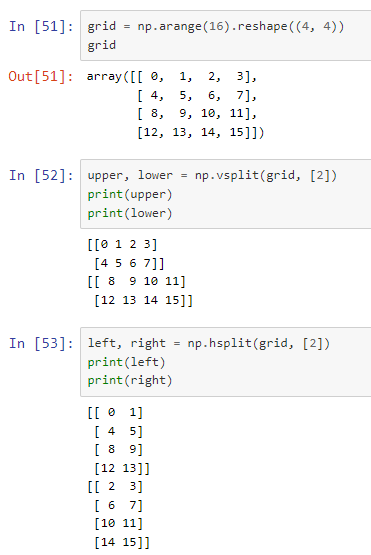
Basic 1D Array Examples

* + - *Example:* avg\_monthly\_precip[2]
  + Use .shape to reveal how many elements a 1D array has in it
    - avg\_monthly\_precip.shape returns (12,)
  + To select a range, specify using [starting\_value, ending\_value]
  + [:5] # first five elements
  + [::2] # every other element
  + [1::2] # every other element, starting at index 1
  + [::-1] # all elements, reversed
  + [5::-2] # reversed every other from index 5
* For 2D arrays, you need to specify both a **row index** and a **column index**
  + Rows are first columns are second! [row index, column index]
  + To select the element in the second row, third column, you can use: [1, 2]
  + To select a range, specify using [start\_row\_index:end\_row\_index, start\_column\_index:end\_column\_index]
    - To select the elements in first row, first two columns: array[0:1, 0:2]
  + To select all rows of a column (entire column), use a colon for the row index: array[:,0]
  + To select all column values of a row (entire row) use a colon for the column index: array[0,:]

2D Array Examples

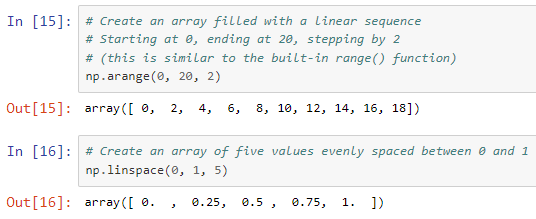


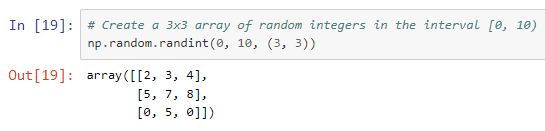


* **Creating NumPy Arrays**
  + Enter data directly for a 1D: np.array([1,2,3])
  + Enter data directly for a 2D: np.array([[1,2,3], [4,5,6]])
  + Make a set of random numbers: np.random.randint(10, size = (3,4)) #2d array 3 rows for columns, integers 0-10
  + Make a set of zeros or ones: np.zeros((dim1,dim2)) or np.ones((dim1,dim2,…))
  + Make a set of sequential values: np.arange(start, stop) (note: this is a 1-d array but you can use .reshape to change dimensions)

Splitting Examples

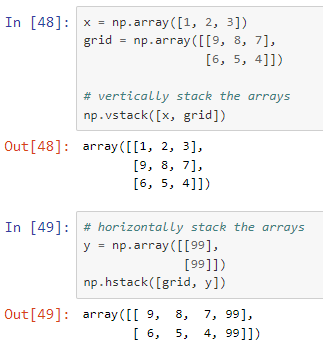
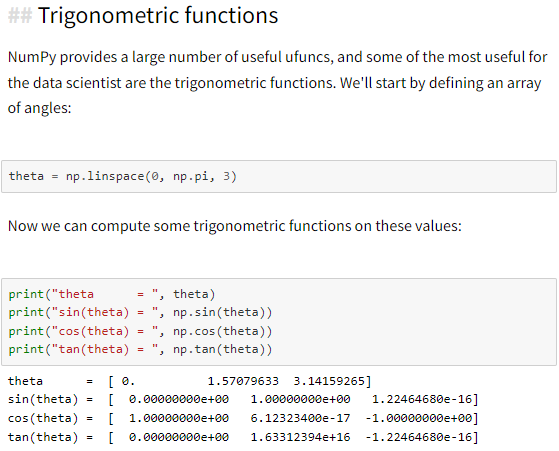
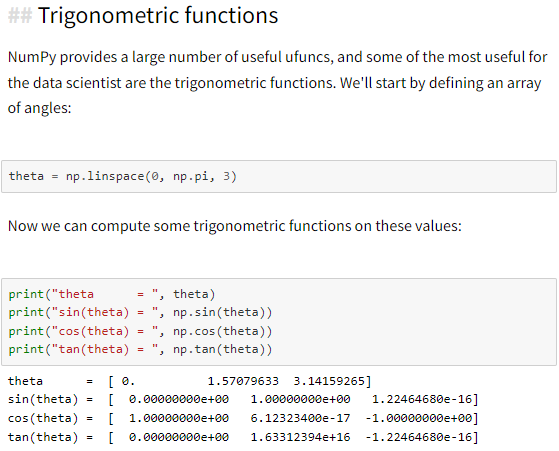
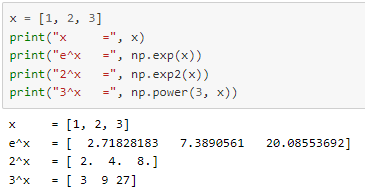
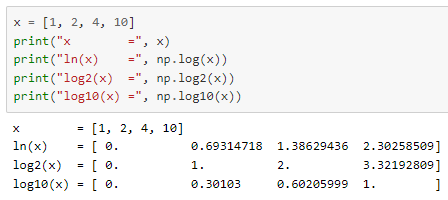
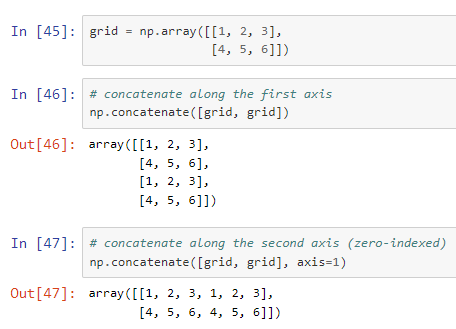
* + # Create a length-10 integer array filled with zeros: np.zeros(10, dtype=int)
  + # Create a 3x5 floating-point array filled with ones: np.ones((3, 5), dtype=float)





* **Helpful NumPy Functions**
  + Np.copy(array) 🡪 copies your array to a new
  + Statistics: np.mean, min, max, median
    - most np functions have an axis argument if you want to summarize on just one axis

Stacking Examples

* + - * axis=0 summary of each column across all rows
      * axis= 1 summary of each row across all columns
  + Combining arrays: concatenate, vstack, hstack:
    - np.concatenate((array1,array2),axis=0) 🡪 adds array2 as rows to end of array1
    - np.concatenate((array1,array2),axis=1) 🡪 adds array2 as columns to end of array1
  + Dividing arrays:
    - np.split(array,3) 🡪 splits array into 3 sub-arrays
    - np.hsplit(array,5) 🡪 splits array horizontally on the 5th index
  + Trig Functions:
    - Sin, Cos, Tan, Arcsin, Arccos, Arctan…
    - Logs and Exps…

Logarithms

*Note: np.log gives natural log; use log2 for base 2, log10 for base10, etc.*

Concatenate Examples

Trig Functions