**Libraries**

**Numpy: for mathemtatical operations**

* + import numpy as np

**Pandas: for data frames**

* + import pandas as pd

**Seaborn: unique set of graphing params built on top of matplotlib**

* + import Seaborn as sns

**Matplotlib: for graphing**

* + import matplotlib
  + import matplotlib.pyplot as plt : specific plots
  + import matplotlib.animation as animation

**import scipy.stats as stats:** for statistical analysis of kurtosis

**Functions**

* **Append**: Var.append(3) - add data to a list
* **Concatenate lists**: + [1,2] + [3,4]
* **Dates**: epoch time: January 1, 1970
  + dtnow = dt.datetime.fromtimestamp(tm.time()

dtnow

< datetime.datetime(2016,10,17,10,28,40,500572)

dtnow.year, dtnow.month, dtnow.day, dtnow.hour

* + delta = dt.timedelta(days = 100)

today = dt.date.today()

today - delta

* **Def: define - used to define a function**
  + def add\_numbers(x,y):

Return x + y

* + - def add\_numbers (x,y,z=none) :

if (z==none):

return x + y

else:

return x + y +z

\* used to define a function when don’t know if two or three vars to be added

* **float: converts to a decimal value; for example change a value in a dict to a float**
* **for**: used to create a loop
  + x = [1,’a’,y,’b’]

for item in x:

print(item)

* **in: gives a Boolean if an element in a list**
  + 1 in [1,2,3]
* **index: used to identify rows**
  + df.set\_index(): default numbers or give column name, or list of names
  + df.sort\_index()
  + df.reset\_index()
* **iterating:** 
  + for a list: for letter in x:

print(x[letter])

* + for key pair: for name in x.values():

print(name)

* + unpacking: for name, email in x.items():

print(name)

print(email)

* **isnull:** **can be used to count nan’s**
  + df.isnull().sum()
* **lambda:** **used to define a quick temporary function**
  + my\_function = lambda a, b, c, : a + b

my\_function(1,2,3)

* **len()**: **identifies the length of a variable or data set**
* **List**: **get column headers**
  + **List(df**.columns.values)
* **map():** **used to look at multiple lists to apply a function**
  + store1 = [5, 2, 7, 6]

store2 = [3,5,7,10]

cheapest = map(min, store1, store2) – must use a with to open map and see contents

* **merge: used to join two data frames using inner outer join on index or column**
* **missing values:**
  + df.fillna : fill holes with na???
  + df2=df.fillna(value=?, method =’bfill’ or ‘pad’, axis = 0 or 1): axis =0 means fill index, axis =1 means fill columns
  + df2=df.fillna(df.mean()): fills with mean of each column
  + df2=df.dropna(axis =1): to drop rows with na’s
* **read\_csv:** 
  + df=pd.read.csv(‘olympics.csv’, index\_col=0, skiprows=1) : ignore first row of data; but start with first column
* **query: to retieve by index name**
  + df.query(‘index\_name==”value”’)
* **repeat lists**: [1]\*3
* **set:** **used to create a set of keys**
  + cylinders= set (d[‘cyl’] for d in mpg)
* **slicing**: **cutting a list**
  + x = ‘this is a string’

print(x[0:2]) – prints only first and second item

* x[-1] – gives back end of string
* x[:3] – implicitly implies start from first
* **splitting: splitting a string by some key such as a space**
  + firstname= “William Thomas Draco Clarke”.split(‘ ‘)[0]

print(firstname)

* **time**:
  + import datetime as dt
  + import time as tm
    - tm.time() : gets current time since epoch
* **type(): to identify the type of variable**
* **unpacking: the act of retrieving a value from a list**
* **while: used in loops for something of undetermined length**
  + i=0

while (i != len(x)):

print(x[i])

i = i + 1

**types of data elements:**

* **tuple:** x=(1,’a’,2,’b’)
* **list:** x = [1,’a’,y,’b’]
* **string:** ‘this is the day of days’
* **dictionary:** x ={‘a’: ‘John’, ‘b’: ‘Mark’} to call x[‘a’]
  + **identify keys:** mpg[0].keys() : ideintifies columns in a csv converted to dict
* **list comprehension:**
  + my\_list = [for number in range(1, 1000) if number %2==0]

my\_list

* **python series**: cross between a list and a dictionary; uses an index and values; the values has a label
  + s1=pd.Series({‘a’: ‘John’, ‘b’: ‘Mark’} )
* **data frame**: like a spread sheet with column headers, and a row index
  + s1=pd.DateFrame(s1, index =[‘d1’, ‘d2’])