Wk	Focus & Medium	Weekly Topic & Assignment

6

Oct 3 to 8

(1 of 4)

git codebook

wk6.d2.lecture python pillars family

10/5 Class

-> do all the basic Python pillars

- **core objects**
- **conditionals**
- **iterators**
- **functions**
- **transposition**

10/6 Class

-> ~~create objects and functions for reporting~~

> all pillars except transformers and classes

10/8/22 – wrap-up

- We completed code on left for week6.
- This sets us up for making transaction generator and finally advaning our work to system design were we will pull project plans, manipulate them, and mock up reporting.

NOTES ARE BELOW TO HELP YOU WITH YOUR HOMEWORK

PLEASE BE CREATIVE

```
  """# -*- coding: utf-8 -*- Created Sep 15 07:58:23 2022
@author:17574 b.hogan@snhu.edu it.304.fall.22
Objective: import data and apply zipper to transform, iterate,
use conditionals, apply functions, leading to python classes work
Library homebase = Python package index:  https://pypi.org
"""
'''=====================
#==========================
#============================================================
#=>STEP 1 get pip library install path from
#============================================================
#==========================
#=====================''' 
import pandas as pd              #dataframe library
import numpy as np               #numeric library
import matplotlib.pyplot as plt  #visualization library
import os
os.getcwd()           #where am i? <get working directory>
#os.chdir('c:\\Users\BBE\DATA\') #some op.sys use one slash
os.chdir('c:\\Users\\17574\Desktop\\data') #microsoft uses 2 \\
os.getcwd()

df0 = pd.DataFrame() #explicitly set the data object
#df0 = pd.read_csv("shakes_corpus_v1.csv")   #ETL method 1
df0 = pd.read_excel("shakes_corpus_v1.xlsx") #ETL method 2
df0.info()
   #             RangeIndex: 37 entries, 0 to 36
   #             Data columns (total 3 columns):
   #              #   Column  Non-Null Count  Dtype
   #             ---  ------  --------------  -----
   #              0   title   37 non-null     object
   #              1   script  37 non-null     object
   #              2   type    37 non-null     object
   #              3   ID      37 non-null     int64
   #             dtypes: int64(1), object(3) memory usage: 1.3+ KB

print(type(df0))  #use type() to always see what an object is
df0.head()
    #                                  title  ...    type
    #             0  Alls Well That Ends Well  ...  Comedy
    #             1            As You Like It  ...  Comedy

#2.1 use pandas df.to_dict() to move data into dictionary object
mydict = df0.to_dict()
print(mydict.keys())     #['title', 'script', 'type', 'ID'])
type(mydict.keys())      # object itself is keys

#2.2 understand what a dictionary and zip is doing
mylist_keys = list(zip(mydict.keys()))
mylist_keys    # [('title',), ('script',), ('type',), ('ID',)]

#Inspect huge data and then break into smaller chunks
mylist_values = list(zip(mydict.values())) #WOW huge !
#point - zip helpful but continue to learn more functions

mylist_values   #======================> #MEGASAURUS
                     #  35: 'Tragedy',
                     # 36: 'Tragedy'},),
                     #{0: 1,
                     # 1: 2,
                     # 2: 3,
```

wk
6
Oct
3
to
8

week 6
(2 of 4)

PLEASE BE CREATIVE

```
'''=====================
#==========================
#============================================================
#=>STEP 2 - seperate Megasaurus into usuable object chunks
#============================================================
#==========================
#====================='''

'''2.1'''
type(mylist_values) #=> [({...})],

'''======> packed as [({...})], =>list, tuple, dictionary'''

type(type(mylist_values[1]) )#hmm doesn't unpack
len(mylist_values)  #=> 4 columns in spreadsheet, ie data objects

'''megasaurus - all plays and words'''
mylist_values
                        # => format is list[(tuple(dict))]
                        # [  ({id:title}),({id:script}),
                        #    ({id:type}), ({id:id}) ]
                        # zip added an key sequential value
'''==>2.2'''
'''use slicing [0:1], [2] to view next level down'''
type(mylist_values[0])  # tuple
mylist_values[0]        #=> [x] is called slicing

                  Out[23]:
                  ({ 0: 'Alls Well That Ends Well',
                     1: 'As You Like It',

'''now think data like in spreadsheet'''
#  columns
#    0      1     2      3
# |title |script| type |  id  |
#  hamlet,oh joy,tragedy, 29

mylist_values[1] #displays all the script text!

'''==>2.3'''
len(mylist_values[1])  # waits its '1' so need to unpack my data

mylist = []
for i in mydict['title'].values():
    mylist.append(i)
mylist
len(mylist)  #37 - does htat match spreadsheet? always know your bounds

title_total_characters = 0    #how many characters?
for i in mylist:
    title_total_characters = title_total_characters + len(i)
title_total_characters  #do you get 560 ?

'''============================================================
==>2.4 autoBOTs304 - repeat this for total script words
#============================================================
===> moved  this into the graded_assign_wk7'''

#=====================
#==========================
#============================================================
#=>STOP! : view 'Variable Explorer' window
#  use this feature to propel data transformation learning
#============================================================
```

wk
6
Oct
3
to
8

week 6
(3 of 4)

CREATIVITY ENCOURAGED

git codebook

```
#==========================
#=====================

'''#=====================
#=> WRAP - UP  Housekeeping
#  delete variables not using; help avoid unnecssary mistakes
#==========
# be mindful how you stage both variable and data names
#   df0 = baseline import
#       df1 = analysis 1
#           df2 = analysis 2
#===================='''

'''==>2.5'''
del mylist_keys    # del removes a variable

'''
mylist2 = []
for i in [mydict.get('title')]:
    mylist.append(i)   #so what happended here a. wrote name list wrong
print(len(mylist2), len(mylist))
#make a note here on what happended...............
mylist    #stacked a list on a dictionary bc meant to use list2
'''
#go back and rest data for part 2
mylist = []
for i in mydict['title'].values():
    mylist.append(i)

'''=====================
#==========================
#==================================================================
#=>STEP 3:  Use dir(object) to learn its methods to get work done
#==================================================================
#==========================
#====================='''
'''==>3.1'''
#===============> use dir() to get functions available for an object
myset = set()
print(type(myset))
dir(myset)
 # '__xor__',  ==> these are constructors, more later
 #    'add', 'clear',    ==> these are methods
 #   'copy','difference', 'difference_update', 'discard',
 #  'intersection','intersection_update', 'isdisjoint', 'issubset',
 #   'issuperset','pop', 'remove','symmetric_difference',
 #   'symmetric_difference_update','union',  'update']'''

'''==>3.1'''# ====> SETS
mylist2 = mylist
mylist2.append("Winters Tale")  #add one duplicate title
myset = set(mylist2)
print(len(mylist),len(myset))  #so got rid of duplicate
del mylist2

#============> ACTION learn what you need and go find it
mystring = ""
print(type(mystring))
dir(mystring)
#'''__subclasshook__',  'capitalize', 'casefold',, 'center',
#'count', 'encode', 'endswith', 'expandtabs', 'find', 'format',
#'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal',
#'isdigit', isidentifier', 'islower', 'isnumeric', 'isprintable',
#'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip',
```

wk
6
Oct
3
to
8

week 6
(4 of 4)

CREATIVITY ENCOURAGED

```
#'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
#'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines','startswith',
#'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']'''
'''====================
#==========================
#============================================================
#=>STEP 4: More dictionary: .keys(), .values(), .get(<key>)
#============================================================
#==========================
#=====================''' 

'''==>4.1'''
mydict.get('title')  #.get() views one series
play_names = [mydict.get('title')]
play_names
        [{0: 'Alls Well That Ends Well',
          1: 'As You Like It',
          2: 'The Comedy of Errors',
mylist
# Now add titles to a different object with an iterator
mylist2 = []
for i in [mydict.get('title')]: #method returns a dict obj
    mylist2.append(i)
mylist2
    [{0: 'Alls Well That Ends Well',
      1: 'As You Like It',
      2: 'The Comedy of Errors',

#3.2 =>  Learn dictionary key, value, items parameters
mylist_key =  []
mylist_values = []
for k,v in mydict.items():
    mylist_key.append(k)
    mylist_values.append(v)
mylist_key              #['title', 'script', 'type', 'ID']
mylist_values   #'''again megasaurus'''

'''==>4.2''' #=>  Understand and count items in a list
len(mylist_values) #hmm  why is this only four ?
mylist_values[0]
mylist_values[1]
mylist_values[2]
mylist_values[3]

#=====================
#==========================
#============================================================
#=>STEP 3:  Use Functions and get Meta Data
#============================================================
#==========================
#=====================
https://docs.python.org/3/library/functions.html#built-in-functions

sum(mylist_values[3])-1
sum(df0['ID'])-1
len(set(df0['ID']))
```

==========> END week 6