Python 101

-String
print(‘มา',end=100*'ก')
print('stat',end=',') #end= คือลงท้ายด้วยอะไร.... แบบไม่ต้องขึ้นบรรทัดใหม่
-Variables
modulo (%) #การหารเอาเศษ เช็คเลขคู่เลขคี่ได้ด้วยการหารด้วย2
ตรวจสอบชนิดของตัวแปรด้วยคำสั่ง type()
-Advanced Printing
print(f'ข้อความ{code}') ห้ามมีเว้นวรรคในชื่อตัวแปร
เช่น print(f'หนึ่ง บวก สอง (1+2) เท่ากับ สาม ({1+2})’)
print(f'{a}+{b} = {a+b}’) —> 5+2=7
-boolean (ตัวแปรที่มีค่า true หรือ false) ตรรกศาสตร์
print(b_T and b_F) —> False
-replace() แทนที่ string
st_test1.replace('e','อี') # แทนที่ e ด้วย อี
'Department of Statistics,'.replace('a','x') # แทนที่ตัว a ด้วย x
-Split() แยก string
'ab cd efg’.split() # ตัด string ที่ช่องว่าง —-> ['ab', 'cd', ‘efg']
'abcdefg'.split('c') #แยกด้วย c (เอาc ออกแล้วแยก) —-> ['ab', 'defg']

Data Structure

-list()
เก็บอะไรก็ได้ ลำดับมีความสำคัญ
list_b = ['ab',2,3.14 , list_a] # กำหนดตัวแปร จำนวนสมาชิกใน list นับตาม ,
list_a[3] # เอาสมาชิกตัวที่ 4 เริ่มนับจาก 0
list_a[-1] # index -1 คือตัวสุดท้ายของ list
-append() เพิ่มสมาชิกใน list
list_x1.append(2) # เก็บ 2 ไปไว้ใน list_x1
-pop() ลบสมาชิกใน list
list_x1.pop() # จะลบจากตัวท้ายสุดก่อน
-len() ตรวจสอบจำนวนสมาชิกของ list
print(list_b) # แสดง list_b
print(len(list_b)) # แสดงจำนวนสมาชิกของ list_b
-list slicing :
list_a[-3:] # : colon # ดึงสมาชิกสามตัวสุดท้ายออกมาได้
list_a[1:4] # จะเอาเลข 2 , 3 , 4
list_a[:] # คือเอาทุกตัว
list_a[1:-1:2] #เอาตั้งแต่ตัวแรกถึงตัวก่อนสุดท้ายโดยมี step = 2
-range()
range(21) # list ของเลข 0 จนถึง 20(จุดสุดท้าย)

list(range(21)) # เรียก ตัวเลข 0-20 ----> [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
คำนวณว่าช่วงเวลา 12:30:15-13:41:07 ห่างกัน กี่ชั่วโมง กี่นาที กี่วินาที (print ออกมาให้สวยงาม)

```
q1 = '12:30:15' #ข้อที่1
Q1 = q1.split(":")
hours1 = int(Q1[0])*3600
minutes1 = int(Q1[1])*60
seconds1 = int(Q1[2])
q2 = '13:41:07'
Q2 = q2.split(":")
hours2 = int(Q2[0])*3600
minutes2 = int(Q2[1])*60
seconds2 = int(Q2[2])
t1 =  hours1 + minutes1 + seconds1
t2 =  hours2 + minutes2 + seconds2
t3 = (t2-t1)
h = int(t3/3600)
m = int((t3%3600)/60)
s = int(t3%60)
print(f' จากเวลา {q1} ถึง {q2} เป็นเวลาห่างกัน {h}:{m}:{s}')
```

-Dictionary

```
dict_a = {1:'Happy','N':'New','2022':'Year'} # { } เรียกว่า curly brackets
dict_a['N'] # กำหนดว่าตัว N คือ New
dict_a.keys() # แสดงผลของ keys ขึ้นมา index ของ dictionary = key
dict_a.values() # แสดงผลของค่าผลลัพธ์
```

เพิ่มสมาชิกใน Dict

```
dict_a[7] = 3.14 # index ใหม่ = keys 7 มีค่าเท่ากับ 3.14
dict_a[7] = 2023
dict_a #มันจะไปแทนตัวเดิม ---> {1: 'Happy', 'N': 'New', '2022': 'Year', 7: 2023}
```

-Numpy Array

```
import numpy as np # การเรียกใช้งาน package (นัมไพ)
list_A = [1,2,3,4,5] # สร้าง list
arr_a = np.array(list_A) # สร้าง list ให้แปลงเป็น array คำสั่งสร้าง array = numpy.array
```

np.zeros()

```
arr2_3_5_0 = np.zeros((3,5)) # สร้าง array 0 ที่มี 3 แถว 5 หลัก ที่มีค่าข้างในเป็น 0 ทั้งหมด
```

np.ones()

```
arr2_4_5_1 = np.ones((4,5)) # มี 4 แถว 5 หลัก
Tree = np.ones((15,9))
Tree[1::,4] = 66
Tree[3:6,3:6] = 66
```

Tree[5:7,5:7] = 66
Tree[5:7,2:7] = 66
Tree[7:9,1:8] = 66
Tree[9:12,0:9] = 66
Tree[4:5,4:6:2] = 0
Tree[6:7,3:7:2] = 0
Tree[8:9,2:8:2] = 0
Tree[10:11,1:9:2] = 0
Tree[12:15,4] = 55
print(Tree)
identity matrix ได้ด้วย np.eye(ขนาด)
np.eye(5)

Basic Programming Concepts

-Functions

```
def function_f1(x): #ชื่อ function = function_f1
   a = x**2 # ยกกำลัง
   y = a + 75
   return y #code ที่เยื้องเป็นส่วนหนึ่งของบรรทัดบนที่ไม่เยื้อง code
   # code ที่อยู่ลำดับเดียวกันมีความสำคัญเท่ากัน

print('done!') # โค้ด level เดียวกันจะไม่ได้เป็นส่วนหนึ่งของกันและกัน
#รันออกมาแล้วมันจะไปเก็บอยู่ในfunction_f1(x)
```

———————————————————————————-

```
def function_f1(x): #ชื่อ #x= input
   a = x**2 # ยกกำลัง
   y = a + 75 #function
   return
```

———————————————————————————-

```
def pi_v(): # ค่า pi
   y = 3.14159265359
   return y
```

———————————————————————————-

```
def print_name_id(name,id):
    print(f'ชื่อ {name} รหัสประจำตัว {id}') # แบบไม่มี return ค่า
```

———————————————————————————-

***Function ไม่มี process ไม่ได้

-Looping(for) สำหรับทำงานซ้ำๆ

```
for member in [1,2,3,4,5,6]: # member จะแทนด้วยสมาชิกใน list
    print(member)
```

——>1
2
3
4
5
6

————————————————————————————————————-

```
def print_name(name): #ตั้งสมาชิกใน list คือ name
    print(f'ฉันชื่อ {name}')
for name in ['วัชราภรณ์','จุฑากาญจน์','ชลธิชา','ญาดา','กิตติคุณ']: # ดึงสมาชิกแต่ละตัวมาเพื่อทำกับ function print_name
    print_name(name)
```

ฉันชื่อ วัชราภรณ์
ฉันชื่อ จุฑากาญจน์
ฉันชื่อ ชลธิชา
ฉันชื่อ ญาดา
ฉันชื่อ กิตติคุณ

————————————————————————————————————-

```
for mem1 in range(2,5): # [2,3,4]
    print(f'now mem1 = {mem1}')
    for mem2 in range(1,13): # [1,2,3,4,5,6,7,8,9,10,11,12]
        print(f'{mem1} x {mem2} = {mem1*mem2}') # อย่าลืมเว้นย่อหน้าเข้ามาด้วย
    print(f'end inner for mem1 = {mem1}') # จุดจบของ loop ด้านใน now mem1 = 2
```

2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20
2 x 11 = 22
2 x 12 = 24
end inner for mem1 = 2
now mem1 = 3
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12

```
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
3 x 11 = 33
3 x 12 = 36
end inner for mem1 = 3
now mem1 = 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
4 x 11 = 44
4 x 12 = 48
end inner for mem1 = 4
```

———————————————————————————-

```
for i in range(100): # range(มีสมาชิก 100 ตัว)
    print('งง', end = ' ') # end = ' ' เว้นแต่ละตัว
```

งง งง
งง งง
งง งง
งง งง งง งง งง งง งง

-operator ที่ใช้ตรวจสอบ condition

```
a = 10 == 20 # ถามว่า 10 เท่ากับ 20 มั้ย?
print(a) —-> False
```

———————————————————————————-

```
def grading(x):
    if x is  int():
        if x < 0:
            y = f'คะแนน {x} ไม่สามารถตัดเกรดได้'
        elif x < 50:
            y = f'คะแนน {x} ได้เกรด F'
        elif x < 55:
            y = f'คะแนน {x} ได้เกรด D'
        elif x < 60:
```

```
            y = f'คะแนน {x} ได้เกรด D+'
        elif x < 65:
            y = f'คะแนน {x} ได้เกรด C'
        elif x < 70:
            y = f'คะแนน {x} ได้เกรด C+'
        elif x < 75:
            y = f'คะแนน {x} ได้เกรด B'
        elif x < 80:
            y = f'คะแนน {x} ได้เกรด B+'
        elif x <= 100:
            y = f'คะแนน {x} ได้เกรด A'
        else:
            y = 'ไม่สามารถตัดเกรดได้'
    else:
        y=f'คะแนน {x} ไม่สามารถตัดเกรดได้'
    return y
```

Pandas101

```
import pandas as pd

from google.colab import drive
drive.mount('/content/drive')  # mount =  เชื่อมไดร์ฟ

data0 = pd.read_csv('/content/drive/MyDrive/DataViz23Data/BKK.csv') #เปิด Drive ในแถบ Files ฝั่งซ้ายแล้ว copy path มาได้เลย
data0

data0.shape # ตรวจสอบจำนวน (แถว,คอลัมน์)

data0.describe() # ดูภาพรวมทางสถิติของข้อมูล
```

-คำสั่ง .iloc

```
data0.iloc[0,6] #iloc เปลี่ยนมุมมองเป็น np.array ชี้เป็นแถว คอลัม = [row,column]
```

-Table Query

```
small_df = data0.iloc[-5:,:] # เอา 5 แถวสุดท้าย ของทุกคอลัม
small_df

small_df[[False,True,True,False,False]] #True = เอาแถวนั้น ,Fal

small_df['Approved_buget']<1000000 # งบน้อยกว่า1000000
```

อยากได้รายได้ที่ได้รับอนุมัติงบประมาณที่มากกว่า 1 ล้านบาท แต่ไม่เกิน 2 ล้าน
(data0[‘Approved_buget’]<=2000000)&(data0['Approved_buget']>1000000)

data0[(data0['Approved_buget']<=2000000)&(data0['Approved_buget']>1000000)]

-Missing Values
data0.isnull() # ตรวจสอบว่าค่านั้นเป็นค่า missing(ว่าง) หรือไม่ # True = ค่าว่าง

data0.isnull().any() #.any( )คือการเอาค่าความจริง ใน column มา or กัน ///False คือ ไม่มี missing เลย /// True มีค่าว่างต้องจัดการ

data0.isnull().any().any() # any รอบแรกสรุปแต่ละคอลัม ,any เอาค่าที่ได้จากตอนแรกมา or กัน เพื่อดูว่าทั้งตารางมีค่าว่างหรือเปล่า

รายการขุดลอกคูคลองของ'สำนักงานเขตคลองสาน' และ 'สำนักงานเขตคลองสามวา'
df = pd.DataFrame(data0,columns=['Department','Item_name'])
df.set_index('Department', inplace=True)

result = df.loc[["สำนักงานเขตคลองสามวา","สำนักงานเขตคลองสาน"],['Item_name']]
result

ลบ Missing
data0.dropna().shape

data0.dropna(subset=[‘Core_budget_code']).shape

Fillna()
set(data0['Core_budget_code']) # เลือกเฉพาะค่าที่ไม่ซ้ำกันออกมา

data0.fillna('-',inplace=True) # แทนค่า nan ด้วย -

unit.fillna(value={'count_room_utility':0,'area_total_min_wa':unit['area_total_min_wa'].mean()}) #dic {}

dropna()
unit.dropna(axis=1) # เอาแค่คอลัมที่ไม่มีmissing เลย

unit.dropna(how = 'all') # ตัดทุกแถวที่เป็น missing หมด

unit.dropna(axis=1, thresh=42000) # แนวตั้ง // int ใส่จำนวนที่มีค่า missing เกินกี่ค่าถึงจะลบ

ตัดเฉพาะคอลัมที่กำหนด
unit.dropna(subset=['propertytype_name_en','price_min','area_usable_min']) # 3columnนี้จะไม่มี missingแน่นอน

-การสร้าง Columns เพิ่ม
df['ชื่อ column ใหม่'] = list of data #list of data มีจำนวนสมาชิก เท่ากับ จำนวน row ของตาราง

small_df['ผู้รับผิดชอบ'] = [‘ลุงตู่','ลุงตู่','ลุงป้อม','ชัชชาติ','พิธา']

-ลบ Columns
small_df.drop(axis=1,columns=['Core_budget_code']) # Axis = 1 แกนตั้ง, Axis = 0 แกนนอน (row)

ต่อตาราง pd.concat()
แกน Y
dataall = pd.concat([unitA,unitB])

แกน X
unit.merge(project, left_on='project_id', right_on='project_id') # จะต่อด้วยKey อะไร // ซ้ายของ unit

หาราคาบ้านที่แพงที่สุดในจังหวัดขอนแก่น
this = unit.merge(project, left_on='project_id', right_on='project_id')
this[this['province_name_th']=='ขอนแก่น'] #หาบ้านที่อยู่ในจังหวัดขอนแก่น

Pandas102

```
import pandas as pd
import os

from google.colab import drive # เชื่อมบัญชีกับ google drive
drive.mount('/content/drive') # เช็คว่าเชื่อมติดหรือไม่
path = '/content/drive/My Drive/dataviz_2021_data' # ชี้ว่าใน google drive โฟลเดอร์ที่เราอยากได้

data_df.head() # ดูหัวตารางแถวบน
```

ต่อตาราง

-ต่อในแนวแกน Y (เพิ่มจำนวนข้อมูล)

```
BKK = data_df[data_df['province_of_onset']=='กรุงเทพมหานคร'] # เลือกดู กรุงเทพ
BKK.head() # ต้องออกมา 5 แถว

BKK2 = data_df[data_df['province_of_onset']=='กรุงเทพมหานคร']
[['sex','age','nationality']] # เลือกดู กรุงเทพแต่เลือกเฉพาะคอลัมน์'sex','age','nationality'
BKK2.head()

pd.concat([BKK2,KKC2]) # ถ้าคอลัมน์เป็นชื่อเดียวกันจะ สลับที่ให้เลย
```

-ต่อในแนวแกน X (เพิ่มรายละเอียดของข้อมูล)

ตาราง1personal_data , ตาราง 2 covid_data

```
personalcovid_data = personal_data.merge(covid_data) # นำ no.1 มาแปะ จะเอาตารางสองมาแปะ คือ covid_data มาต่อ
personalcovid_data.head()
```

———————————————————————————————-

คอนโดแล้วเรียงจากราคาถูกที่สุด 3 อันดับแรก

```
unit_df[unit_df['propertytype_name_en']=='Condo'].sort_values(by=['price_min'
]).iloc[:3] #ทำให้เหลือแค่ข้อมูลคอนโดก่อนแล้วค่อย sort by

unit_df[unit_df['propertytype_name_en']=='Condo'].sort_values(by=['price_min'
]).iloc[:3,-1] #ดูจังหวัด
```

-Group by

```
unit_df.groupby('province_en').min().head() #จัดกลุ่มโดยใช้ Column 'province_en' และหา min column ที่เหลือ

unit_df.groupby(‘province_en')['price_min'].min().head()
```

หา 10 อันดับอสังหาริมทรัพย์ที่แพงที่สุดเป็นรายจังหวัด

```
unit_df.groupby('province_en')
['price_min'].max().reset_index().sort_values(by='price_min',ascending=
False).head(10)
```

———————————————————————————————-

หา 5 อันดับ condo ที่แพงที่สุดเป็นรายจังหวัด (2-6)

```
unit_df[unit_df[‘propertytype_name_en']=='Condo'].groupby('province_en')
['price_min'].max().reset_index().sort_values(by='price_min',ascending=False).he
ad(10)
```

-Create Pandas table

```
data_list = [{'account':'Jones LLC','Jan':150,'Feb':200,'Mar':140},
            {'account':'Alpha Co','Jan':200,'Feb':210,'Mar':215},
            {'account':'Blue Inc','Jan':50,'Feb':90,'Mar':95},
            ] # สร้าง list ขึ้นมา

df1 = pd.DataFrame(data_list) # สร้างตาราง
df1
```