COVID-19: The Latest Wave

ฟังก์ชัน **read_data** ข้างล่างนี้ให้บริการอ่านข้อมูลจากแฟ้มที่เก็บจำนวนผู้ติดเชื้อไวรัส COVID-19 รายใหม่รายวันในจังหวัดต่าง ๆ

```
def read_data(filename):
    d = np.loadtxt(filename, delimiter=",", encoding='utf-8', dtype=str)
    new_cases = np.array(d[1:,1:], dtype=int)
    norm = new_cases / np.max(new_cases, axis=1).reshape((new_cases.shape[0],1))
    return {'new_cases': new_cases,
            'norm_data': norm,
            'province_names': d[1:,0],
            'dates': d[0,1:]}
```

ข้างล่างนี้เป็นตัวอย่างแสดงข้อมูลภายในแฟ้ม (แฟ้มตัวอย่างนี้เก็บข้อมูลแค่ 4 จังหวัด ของวันที่ 14 ถึง 16 เมษายน 2564) ข้อมูลต่าง ๆ ในแฟ้มถูกคั่นด้วยเครื่องหมายจุลภาค บรรทัดแรกเป็นลำดับของวันที่ต่าง ๆ (ในรูปแบบปีเดือนวัน) บรรทัดถัด ๆ ไป เป็นข้อมูลของแต่ละจังหวัด บรรทัดละจังหวัด เริ่มด้วยชื่อจังหวัดตามด้วยรายการของจำนวนผู้ติดเชื้อใหม่ในแต่ละวัน

```
provinces,2021-04-14,2021-04-15,2021-04-16
กระบี่,5,5,5
กรุงเทพมหานคร,351,409,312
กาญจนบุรี,16,5,4
ขอนแก่น,18,18,8
```

สมมติว่าแฟ้มข้างบนนี้ชื่อ **sample.csv** การทำคำสั่ง **data = read_data('sample.csv')** จะได้ **data** เป็น dict เก็บข้อมูลข้างล่างนี้

```
{
   'new_cases':      array([[  5,        5,        5],
                            [351,     409,      312],
                            [ 16,        5,        4],
                            [ 18,       18,        8]]),
   'norm_data':      array([[1.        , 1.        , 1.        ],
                            [0.85819071, 1.        , 0.76283619],
                            [1.        , 0.3125    , 0.25      ],
                            [1.        , 1.        , 0.44444444]]),
   'province_names': array(['กระบี่', 'กรุงเทพมหานคร', 'กาญจนบุรี', 'ขอนแก่น']),
   'dates':          array(['2021-04-14', '2021-04-15', '2021-04-16']),
}
```

ผลที่ได้จากฟังก์ชันเป็น dict ที่มี key-value 4 คู่ดังนี้

- คีย์ **new_cases** เก็บอาเรย์สองมิติของจำนวนเต็ม แต่ละแถวคือข้อมูลของแต่ละจังหวัด แต่ละคอลัมน์คือข้อมูลของแต่ละวัน
- คีย์ **norm_data** เก็บอาเรย์สองมิติของจำนวนจริง ข้อมูลที่ตำแหน่งหนึ่ง ๆ ในอาเรย์นี้ มีค่าเท่ากับข้อมูลที่ตำแหน่งเดียวกันในอาเรย์ **new_cases** หารด้วยค่ามากสุดในแถวนั้น เช่น ข้อมูลที่แถว 1 คอลัมน์ 2 ของอาเรย์นี้ เท่ากับ 312 / 409 = 0.76283619
- คีย์ **province_names** เก็บอาเรยหนึ่งมิติของสตริง โดยช่องที่ **k** เก็บชื่อจังหวัดของแถวที่ **k** ของอาเรย์สองมิติสองอาเรย์ข้างต้น
- คีย์ **dates** เก็บอาเรยหนึ่งมิติของสตริง โดยช่องที่ **k** เก็บปีเดือนวันของคอลัมน์ที่ **k** ของอาเรย์สองมิติสองอาเรย์ข้างต้น

ฟังก์ชันที่ต้องเขียน

def max_new_cases_date(data):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- คืน tuple สองช่อง
  - ช่องที่ 0 เก็บสตริงปีเดือนวันที่มียอดจำนวนผู้ติดเชื้อรายใหม่รวมทุกจังหวัดมากที่สุด
  - ช่องที่ 1 เก็บยอดรวมมากสุดนั้น
  - ถ้ามีหลายวันที่ยอดรวมสูงสุดเท่ากัน ให้คืนของวันที่มาก่อน
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**
  max_new_cases_date(data) คืน **('2021-04-16', 1577)**

แฟ้ม **TH_20210401_20210416.csv** เก็บข้อมูลทั้ง 77 จังหวัดตั้งแต่วันที่ 1 ถึง 16 เมษายน พ.ศ. 2564)

def max_new_cases_province(data, beg_date, end_date):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- **beg_date** กับ **end_date** เป็นสตริงปีเดือนวันของวันเริ่มและวันสิ้นสุดที่สนใจที่มีอยู่ใน **data**
- คืน tuple สองช่อง

  ช่องที่ 0 เก็บชื่อจังหวัดที่มียอดจำนวนผู้ติดเชื้อรายใหม่รวมมากที่สุดตั้งแต่วัน **beg_date** ถึง **end_date**

  ช่องที่ 1 เก็บยอดรวมมากสุดนั้น

  ถ้ามีหลายจังหวัดที่ยอดรวมสูงสุดเท่ากัน ให้คืนของจังหวัดที่มาก่อนในแฟ้ม
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**

  max_new_cases_province(data, '2021-04-10', '2021-04-13') คืน **('เชียงใหม่', 854)**

def max_new_cases_province_by_dates(data):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- คืน numpy array ที่มี shape เท่ากับ (จำนวนวันใน **data**, 3)

  คอลัมน์ที่ 0 เก็บปีเดือนวัน แถวต่าง ๆ ในอาเรย์เรียงตามปีเดือนวันจากน้อยไปมาก (บนลงล่าง)

  คอลัมน์ที่ 1 เก็บชื่อจังหวัดที่มียอดผู้ติดเชื้อรายใหม่รวมมากสุดในวันนั้น

  คอลัมน์ที่ 2 เก็บยอดรวมของผู้ติดเชื้อรายใหม่นั้น

  ถ้ามีหลายจังหวัดที่ยอดรวมสูงสุดเท่ากันในหนึ่งวัน ให้คืนของจังหวัดที่มาก่อนในแฟ้ม
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**

  max_new_cases_province_by_dates(data) คืน

```
array([['2021-04-01', 'สมุทรสาคร', 11],
       ['2021-04-02', 'กรุงเทพมหานคร', 20],
       ['2021-04-03', 'กรุงเทพมหานคร', 32],
       ['2021-04-04', 'กรุงเทพมหานคร', 35],
       ['2021-04-05', 'นราธิวาส', 94],
       ['2021-04-06', 'กรุงเทพมหานคร', 156],
       ['2021-04-07', 'กรุงเทพมหานคร', 216],
       ['2021-04-08', 'นราธิวาส', 146],
       ['2021-04-09', 'กรุงเทพมหานคร', 268],
       ['2021-04-10', 'กรุงเทพมหานคร', 185],
       ['2021-04-11', 'กรุงเทพมหานคร', 236],
       ['2021-04-12', 'เชียงใหม่', 246],
       ['2021-04-13', 'เชียงใหม่', 251],
       ['2021-04-14', 'กรุงเทพมหานคร', 351],
       ['2021-04-15', 'กรุงเทพมหานคร', 409],
       ['2021-04-16', 'กรุงเทพมหานคร', 312]])
```

- หมายเหตุ: การสร้างอาเรย์ที่สามารถเก็บข้อมูลได้หลายแบบในอาเรย์เดียวกันทำได้โดยระบุ **dtype=object** ตอนจองอาเรย์ เช่น

```
a = np.ndarray( (6,), dtype=object ) # จองอาเรย์หนึ่งมิติ 6 ช่อง เก็บอะไรก็ได้
a[0] = 'A string'
a[1] = 12.4
a[2:] = 100
```

def most_similar(data, province):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- **province** เป็นสตริงเก็บชื่อจังหวัดที่มีอยู่ใน **data**
- คืน สตริงชื่อจังหวัด (ที่ไม่ใช่ **province**) ที่มีข้อมูล **norm_data** "คล้าย" กับ **norm_data** ของ **province** มากที่สุด

  ถ้ามีหลายจังหวัดที่คล้ายกับ **province** มากสุดเท่ากัน ให้คืนชื่อจังหวัดที่มาก่อนในแฟ้ม

  ความคล้ายกันของ 2 จังหวัดใดวัดจากค่า $\Sigma\delta^2$ ซึ่งคือผลรวมของกำลังสองของผลต่างของค่า **norm_data** ในแต่ละวันของ 2 จังหวัดนั้น เช่น แฟ้ม **sample.csv** ที่แสดงในหน้าแรกมี $\Sigma\delta^2$ ของกระบี่กับกาญจนบุรี คือ

  $(1-1)^2 + (1-0.3125)^2 + (1-0.25)^2 = 1.03515625$ $\Sigma\delta^2$ **ยิ่งน้อย หมายความว่า ยิ่งคล้ายมาก**
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**

  most_similar(data, 'กรุงเทพมหานคร') คืน **'ขอนแก่น'**

def most_similar_province_pair(data):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- คืน tuple สองช่อง ทั้งสองช่องเก็บชื่อจังหวัด (ที่ไม่ใช่จังหวัดเดียวกัน) ใน **data** ที่คล้ายกันมากสุด (คือมีค่า $\Sigma\delta^2$ น้อยสุด) ตามนิยามความคล้ายที่ได้อธิบายในฟังก์ชัน **most_similar**
  ถ้าหาคำตอบได้หลายคู่ คืนคู่ไหนก็ได้ที่คล้ายกันมากสุดหนึ่งคู่
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**
  most_similar_province_pair(data) คืน **('ระนอง', 'สตูล')**

def most_similar_in_period(data, province, beg_date, end_date):

- **data** เป็น dict เก็บข้อมูลที่อธิบายข้างต้น
- **province** เป็นสตริงเก็บชื่อจังหวัดที่มีอยู่ใน **data**
- **beg_date** กับ **end_date** เป็นสตริงปีเดือนวันของวันเริ่มและวันสิ้นสุดที่สนใจที่มีอยู่ใน **data**
- คืน tuple สามช่อง
  ช่องที่ 0 เก็บชื่อจังหวัด (ขอเรียกว่า **p** ซึ่งไม่ใช่ชื่อเดียวกับ **province**)
  ช่องที่ 1 เก็บปีเดือนวัน (ขอเรียกว่า **d1**) และช่องที่ 2 เก็บปีเดือนวัน (ขอเรียกว่า **d2**) **d1** มาก่อนหรือเป็นวันเดียวกับ **d2** โดยจำนวนวันตั้งแต่ **d1** ถึง **d2** เท่ากับจำนวนวันตั้งแต่ **beg_date** ถึง **end_date**
  โดยเมื่อนำค่า **norm_data** ของ **province** ตั้งแต่ **beg_date** ถึง **end_date** ไปคำนวณค่า $\Sigma\delta^2$ กับ **norm_data** ของ **p** ตั้งแต่ **d1** ถึง **d2** จะมีค่า $\Sigma\delta^2$ น้อยสุด ในบรรดาช่วงเวลาอื่นทุกช่วง (ที่จำนวนวันเท่ากัน) ของทุกจังหวัด
  ถ้ามีคำตอบอื่นที่มีค่า $\Sigma\delta^2$ น้อยสุดที่เท่ากัน ให้คืนชื่อจังหวัดที่มาก่อนในแฟ้ม และถ้าในจังหวัดเดียวกันมีหลายช่วงที่ค่า $\Sigma\delta^2$ น้อยสุดเท่ากัน ก็ให้คืนที่มีวันเริ่มต้นมาก่อน
- ตัวอย่าง: ให้ **data = read_data('TH_20210401_20210416.csv')**
  most_similar_in_period(data, 'กรุงเทพมหานคร', '2021-04-05', '2021-04-09')
  คืน **('ปทุมธานี', '2021-04-07', '2021-04-11')**
  most_similar_in_period(data, 'กรุงเทพมหานคร', '2021-04-01', '2021-04-16')
  คืน **('ขอนแก่น', '2021-04-01', '2021-04-16')**
  ซึ่งเหมือนกับผลของ **most_similar(data, 'กรุงเทพมหานคร')**

ข้อห้าม

เนื่องจากโจทย์ข้อนี้ต้องการทดสอบความเข้าใจในเนื้อหาเรื่อง **numpy** จึงกำหนดข้อห้ามดังต่อไปนี้

- ห้าม **import package** ใด ๆ เพิ่มเติม
- ห้ามใช้คำสั่ง **for, while, list comprehension**
  (รวมถึงการใช้ **set/dict comprehension, map, reduce,** ฟังก์ชันแบบ **recursive** ถ้ารู้ว่าคืออะไร)
- ไม่เขียนคำสั่งใด ๆ นอกฟังก์ชัน แต่สามารถเขียนฟังก์ชันเพิ่มเติมได้
- ไม่ใช้ตัวแปรใด ๆ ที่อยู่นอกฟังก์ชัน

ข้อแนะนำ

- ศึกษาเพิ่มเติมเรื่อง **broadcast** ฟังก์ชัน **transpose** และการตั้งค่า **axis** ของอาเรย์สามมิติ
- ถ้าไม่เขียนการทำงานใด ๆ ในฟังก์ชันใด อย่าลืมเขียนคำสั่ง **return** สักหนึ่งบรรทัดในฟังก์ชันนั้น
- เขียนคำสั่งทดสอบต่าง ๆ ได้ในฟังก์ชัน **main**

โปรแกรมต้นฉบับ

```
# Prog-12: COVID-19: The Latest Wave
# 6?3?????21 Name ?
```

ใส่เลขประจำตัว ชื่อ นามสกุล

```
import numpy as np

def read_data(filename):
    d = np.loadtxt(filename, delimiter=",", encoding='utf-8', dtype=str)
    new_cases = np.array(d[1:,1:], dtype=int)
    norm = new_cases / np.max(new_cases, axis=1).reshape((new_cases.shape[0],1))
    return {'new_cases': new_cases,
            'norm_data': norm,
            'province_names': d[1:,0],
            'dates': d[0,1:]}

def max_new_cases_date(data):

def max_new_cases_province(data, beg_date, end_date):

def max_new_cases_province_by_dates(data):

def most_similar(data, province):

def most_similar_province_pair(data):

def most_similar_in_period(data, province, beg_date, end_date):

def main():
  # put your own testing codes in this function
  return
main()
```

โปรแกรมที่ส่ง ห้ามเปลี่ยนอะไรใด ๆ ในส่วนที่มีพื้นหลังเป็นสีแดง และที่เป็นตัวสีแดงโดยเด็ดขาด เปลี่ยนได้เฉพาะส่วนที่มีพื้นหลังเป็นสีเขียวเท่านั้น

อ่านรายละเอียดในหัวข้อ ข้อห้ามด้วย