

Weather Report (EP. 2)

เว็บไซต์ openweathermap.org มีข้อมูลสภาพอากาศปัจจุบันและข้อมูลพยากรณ์สภาพอากาศของนานาประเทศมากมาย การเข้าถึงข้อมูลมีทั้งแบบคิดและไม่คิดค่าใช้จ่ายซึ่งต้องลงทะเบียนกับระบบก่อนใช้งาน เพื่อความสะดวก ผู้ออกโจทย์ได้นำข้อมูลส่วนหนึ่งของจังหวัดต่าง ๆ ในประเทศไทย (จำนวน 39 จังหวัด) มารวม จัดรูปแบบ และเพิ่มข้อมูลอีกเล็กน้อย ได้เป็นแฟ้ม [th_weather_39.json](#) (ดาวน์โหลดแฟ้มนี้มาไว้ในโฟลเดอร์เดียวกับโปรแกรมที่จะเขียน) เมื่อใช้คำสั่งข้างล่างนี้

```
import json
data = json.load(open('th_weather_39.json'))
```

จะได้ dict (ชื่อ `data`) ที่เก็บข้อมูลสภาพอากาศ ขอให้สัปดาห์เปิดแฟ้ม `th_weather_39.json` (ด้วย notepad หรือโปรแกรมอื่น หรือใช้บริการที่ <https://codebeautify.org/jsonviewer> จะดูโครงสร้างข้อมูลได้ง่ายขึ้น) เพื่อศึกษาโครงสร้างข้อมูลภายในด้วยตนเอง สำหรับความหมายของคีย์ต่าง ๆ ในแฟ้มนี้ หาอ่านได้ใน <https://openweathermap.org/forecast5>)

ฟังก์ชันที่ต้องเขียน

def top_K_max_temp_by_region(data, K):

- o `data` เป็น dict เก็บข้อมูลสภาพอากาศ, `K` เป็นจำนวนเต็มบวก
- o คืน dict ที่มี key เป็นชื่อย่อภาค และ value เป็นลิสต์เก็บข้อมูลอุณหภูมิสูงสุด `K` ตัว (หรือเท่าที่มีถ้ามีน้อยกว่า `K`) ของภาค ข้อมูลอุณหภูมิเก็บใน tuple ประกอบด้วยอุณหภูมิ จังหวัด และวันเวลา เรียงตามอุณหภูมิสูงสุดจากมากไปน้อย ถ้าอุณหภูมิเท่ากันก็เรียงตามชื่อจังหวัดจากน้อยไปมากตามพจนานุกรม ถ้าเป็นจังหวัดเดียวกันก็เรียงตามวันเวลาจากน้อยไปมาก

เช่น ถ้า `data` ได้มาจากแฟ้ม `th_weather_39.json`

`top_K_max_temp_by_region(data, 2)` คืน

```
{
  'C': [(38.54, 'Sukhothai Thani', '2021-04-09 06:00:00'),
        (38.18, 'Phitsanulok', '2021-04-09 06:00:00')],
  'E': [(36.45, 'Prachin Buri', '2021-04-09 06:00:00'),
        (36.15, 'Chachoengsao', '2021-04-09 09:00:00')],
  'N': [(38.98, 'Mae Hong Son', '2021-04-10 06:00:00'),
        (38.73, 'Mae Hong Son', '2021-04-10 09:00:00')],
  'NE': [(37.31, 'Surin', '2021-04-06 09:00:00'),
          (37.29, 'Ubon Ratchathani', '2021-04-10 09:00:00')],
  'S': [(34.93, 'Yala', '2021-04-10 06:00:00'),
        (34.74, 'Yala', '2021-04-07 06:00:00')],
  'W': [(38.97, 'Kanchanaburi', '2021-04-09 09:00:00'),
        (38.11, 'Kanchanaburi', '2021-04-09 06:00:00')]
}
```

อุณหภูมิของทุกฟังก์ชัน
ให้ใช้ส่วนที่มีคีย์เป็น `temp`

def average_temp_by_date(data, region):

- o `data` เป็น dict เก็บข้อมูลสภาพอากาศ
- o `region` เป็นสตริงเก็บชื่อภาค ได้แก่ `N, E, W, S, C, NE` หรือ `ALL` (`ALL` หมายถึงทุกภาคทั้งประเทศ)
- o คืน ลิสต์ของ tuples แต่ละ tuple เก็บข้อมูลสองช่อง วันที่และอุณหภูมิเฉลี่ยของทั้งภาค `region` ในวันนั้น ข้อมูลวันที่มีตามที่ปรากฏใน `data` โดยข้อมูลในลิสต์นี้เรียงตามวันที่จากน้อยไปมาก

เช่น ถ้า `data` ได้มาจากแฟ้ม `th_weather_39.json`

`average_temp_by_date(data, 'C')` คืน

```
[
  ('2021-04-06', 28.395740740740735),
  ('2021-04-07', 28.821388888888887),
  ('2021-04-08', 30.702083333333334),
  ('2021-04-09', 31.940138888888896),
  ('2021-04-10', 31.774027777777786),
  ('2021-04-11', 30.828333333333333)
]
```

คำเตือน
ข้อมูลใน `data` ตอนตรวจให้คะแนน
อาจไม่ใช่ข้อมูลที่ได้จาก
`th_weather_39.json` ก็ได้
แต่มีโครงสร้างข้อมูลเหมือนกัน

```
def max_rain_in_3h_periods(data, region, date):
```

- o **data** เป็น dict เก็บข้อมูลสภาพอากาศ
- o **region** เป็นสตริงเก็บชื่อภาค ได้แก่ **N, E, W, S, C** หรือ **ALL** (**ALL** หมายถึงทุกภาคทั้งประเทศ)
- o **date** เป็นสตริงเก็บวันที่ในรูปแบบ ปี-เดือน-วัน เช่น **2021-04-12**
- o **คืน** ลิสต์ของ tuples แต่ละ tuple เก็บข้อมูลสองช่อง เลขเริ่มต้นชั่วโมง และปริมาณน้ำฝนมากที่สุดที่พบในภาค **region** ของวันที่ **date** เช่น ถ้า **data** ได้มาจากแฟ้ม **th_weather_39.json**

```
max_rain_in_3h_periods(data, 'ALL', '2021-04-07') คืน
```

```
[  
  (0, 3.52),  
  (3, 4.81),  
  (6, 5.66),  
  (9, 1.45),  
  (12, 10.68),  
  (15, 13.35),  
  (18, 3.66),  
  (21, 4.64)  
]
```

```
def AM_PM_weather_description_by_region(data, date):
```

- o **data** เป็น dict เก็บข้อมูลสภาพอากาศ
- o **date** เป็นสตริงเก็บวันที่ในรูปแบบ ปี-เดือน-วัน เช่น **2021-04-12**
- o **คืน** dict ที่มี key เป็นชื่อย่อภาค และ value เป็น dict ที่มี key '**AM**' กับ '**PM**' และ value เป็นข้อความบรรยายสภาพอากาศที่พบเป็นส่วนใหญ่ในช่วงก่อนเที่ยงวัน (AM) กับช่วงหลังเที่ยงวัน (PM) ของวันที่ **date** เช่น ถ้า **data** ได้มาจากแฟ้ม **th_weather_39.json**

```
AM_PM_weather_description_by_region(data, '2021-04-09') คืน
```

```
{  
  'C': {'AM': 'few clouds', 'PM': 'light rain'},  
  'E': {'AM': 'broken clouds', 'PM': 'light rain'},  
  'N': {'AM': 'clear sky', 'PM': 'clear sky'},  
  'NE': {'AM': 'scattered clouds', 'PM': 'broken clouds'},  
  'S': {'AM': 'light rain', 'PM': 'light rain'},  
  'W': {'AM': 'scattered clouds', 'PM': 'broken clouds'}  
}
```

จากตัวอย่างแปลว่า ในวันที่ '2021-04-09' พบว่า ส่วนใหญ่ของข้อความคำบรรยายสภาพอากาศ (weather description) ของจังหวัดต่าง ๆ ในภาคกลาง (C) ก่อนเที่ยง (AM) คือ **few clouds** (เพื่อให้่าย ในกรณีข้อความบรรยายสภาพอากาศที่พบเป็นจำนวนมากสุด มีจำนวนเท่ากันหลาย ๆ ข้อความ ให้เลือกข้อความที่มาก่อนเรียงตามพจนานุกรม)

```
def most_varied_weather_provinces(data):
```

- o **data** เป็น dict เก็บข้อมูลสภาพอากาศ
 - o **คืน** เซตของชื่อจังหวัดต่าง ๆ ที่มีสภาพอากาศหลากหลายเป็นจำนวนมากสุด
- พิจารณาจาก weather description ที่มีคำบรรยายแตกต่างกันเป็นจำนวนมากสุด เช่น สมมติว่า **data** มีแค่ 3 จังหวัด แต่ละจังหวัดเก็บข้อมูลแค่ 5 เวลา เมื่อสรุปคำบรรยายสภาพอากาศได้ดังนี้
- **Loei** มีสภาพอากาศทั้งหมดเป็น 'clear sky', 'clear sky', 'clear sky', 'clear sky', 'clear sky'
 - **Lampang** มีสภาพอากาศทั้งหมดเป็น 'few cloud', 'few cloud', 'light rain', 'few cloud', 'few cloud'
 - **Yala** มีสภาพอากาศทั้งหมดเป็น 'clear sky', 'light rain', 'light rain', 'scattered clouds', 'heavy intensity rain'
- ก็ต้องคืน {'Yala'} เป็นคำตอบ เพราะมีคำบรรยายสภาพอากาศที่ต่างกัน 4 แบบ เป็นจำนวนมากสุด
- ถ้า **data** ได้มาจากแฟ้ม **th_weather_39.json**

```
most_varied_weather_provinces(data) คืน {'Phichit', 'Ratchaburi'}
```

เหตุที่คืนสองจังหวัด เพราะสองจังหวัดนี้มีจำนวนคำบรรยายสภาพอากาศที่ต่างกันจำนวนเท่ากัน และเป็นจำนวนมากสุด

โปรแกรมต้นฉบับ

```
# Prog-11: Weather report (EP.2)
```

```
# 6?3????21 Name ?
```

ใส่เลขประจำตัว ชื่อ นามสกุล

```
import json
import math
```

```
def top_K_max_temp_by_region(data, K):
```

```
def average_temp_by_date(data, region):
```

```
def max_rain_in_3h_periods(data, region, date):
```

```
def AM_PM_weather_description_by_region(data, date):
```

```
def most_varied_weather_provinces(data):
```

```
def main():
```

```
    # put your own testing codes in this function
```

```
main()
```

โปรแกรมที่ส่ง ห้ามเปลี่ยนอะไรใด ๆ ในส่วนที่มีพื้น
หลังเป็นสีแดง และที่เป็นตัวสีแดงโดยเด็ดขาด
เปลี่ยนได้เฉพาะส่วนที่มีพื้นหลังเป็นสีเขียวเท่านั้น

ห้าม import คลังคำสั่งใด ๆ เพิ่มเติม

คำสั่งในทุกฟังก์ชันต้องไม่ใช่
ตัวแปรใด ๆ ที่อยู่นอกฟังก์ชัน

ข้อแนะนำ

ชมวิดีโอเพิ่มเติมเกี่ยวกับโครงสร้างข้อมูลสภาพอากาศ และคำอธิบายเพิ่มเติมของฟังก์ชันทั้งสี่ได้ที่

- <https://youtu.be/svLYt7RJpEw>
- <https://youtu.be/HGRT0FQPYpk>