Problem-Based Learning
Python 3
Lecture: 10-crawler

Online Python

- https://colab.research.google.com/
- https://www.onlinegdb.com/#

Lecture Notes

- https://web.phy.ntnu.edu.tw/~hongyi/?url=notes

Python list, Python Dict, Numpy array, Pandas DataFrame

✓ Python list:

```
[[96, 65, 73], [88, 76, 82], [92, 84, 89]
```

✓ Python dict:

```
{'a':[96, 65, 73], 'b':[88, 76, 82], 'c':[92, 84, 89]}
```

✓ Numpy array:

```
[[96 65 73]
 [88 76 82]
 [92 84 89]]
```

index →

✓ Pandas DataFrame:

	A	B	C ← columns
a	96	65	73
b	88	76	82
c	92	84	89

Python dict (column-like) → Pandas DataFrame

```
import pandas as pd
a = {
    'A': [1, 2, 3],
    'B': [6, 7, 8],
    'C': [2, 3, 4]
    }
print(a)
ind = ['a', 'b', 'c']
d = pd.DataFrame(a, index=ind)
print(d)
```

keys() values()

```
{'A': [1, 2, 3], 'B': [6,
7, 8], 'C': [2, 3, 4]}
```

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

Python dict (row-like) → Pandas DataFrame

keys() values()

```
{'a': [1, 6, 2], 'b': [2, 7, 3], 'c': [3, 8, 4]}
```

```
   A  B  C
a  1  6  2
b  2  7  3
c  3  8  4
```

```
import pandas as pd
a = {
     'a': [1, 6, 2],
     'b': [2, 7, 3],
     'c': [3, 8, 4]
     }
col = ['A', 'B', 'C']
d = pd.DataFrame(a.values(), index=a.keys(), columns=col)
print(d)
```

Pandas DataFrame：增加直行多筆資料

```
b = d
a2 = {
     'D':[7,8,9],
     'E':[3,4,5]
     }
ind = ['a', 'b', 'c']
d2 = pd.DataFrame(a2, index=ind)
b = pd.concat([b,d2], axis=1)
print(b)
```

	A	B	C	D	E
a	1	6	2	7	3
b	2	7	3	8	4
c	3	8	4	9	5

← axis=1

↑ axis=0

Pandas DataFrame：增加直行多筆資料

- b = d
- a2 = {
  'a':[7,3],
  'b':[8,4],
  'c':[9,5],
  }
- col2 = ['D', 'E’]
- d2 = pd.DataFrame(a2.values(), index=c.keys(), columns=col2)
- b = pd.concat([b,d2], axis=1)
- print(b)

	A	B	C	D	E
a	1	6	2	7	3
b	2	7	3	8	4
c	3	8	4	9	5

← axis=1

↑ axis=0

Pandas DataFrame：增加橫列多筆資料

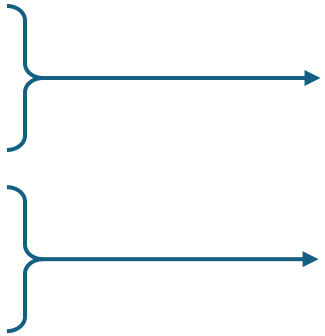
```
b = d
a2 = {
     'A':[4,5],
     'B':[9,1],
     'C':[5,6]
     }
ind2 = ['d', 'e']
d2 = pd.DataFrame(a2, index=ind2)
b = pd.concat([b,d2], axis=0)
print(b)
```

Pandas DataFrame：增加橫列多筆資料

```
b = d
a2 = {
     'd':[4,9,5],
     'e':[5,1,6],
     }
d2 = pd.DataFrame(a2.values(), index=c.keys(), columns=col)
b = pd.concat([b,d2], axis=0)
print(b)
```

爬蟲

- 發送HTTP請求
- 接收並解析響應
- 提取數據
- 儲存數據



爬蟲工具

- `requests`
- `Pandas`

- 數據格式：

JSON　　HTML　　CSV

Accurate Weather Forecasts

- https://open-meteo.com/

```
https://api.open-
meteo.com/v1/forecast?latitude=25.0330&longitude=121.5654&dai
ly=temperature_2m_max,temperature_2m_min
```

URL解析

https://api.open-meteo.com/v1/forecast?latitude=25.0330&longitude=121.5654&daily=temperature_2m_max,temperature_2m_min

- url = "https://api.open-meteo.com/v1/forecast"
- params = {
    "latitude": 25.0330,
    "longitude": 121.5654,
    "daily": "temperature_2m_max,temperature_2m_min"
}

Request

- `response = requests.get(url, params=params)`
- `data = response.json()`
- `print(data)`

dict in Python


```
{"latitude":25,"longitude":121.625,"generationtime_ms":0.025987625122070
3,"utc_offset_seconds":28800,"timezone":"Asia/Taipei","timezone_abbreviati
on":"GMT+8","elevation":12,"daily_units":{"time":"iso8601","temperature_2
m_max":"°C","temperature_2m_min":"°C"},"daily":{"time":["2025-03-23","2025-
03-24","2025-03-25","2025-03-26","2025-03-27","2025-03-28","2025-03-
29"],"temperature_2m_max":[28.7,29.6,30.3,31.7,30.9,23.7,17.2],"temperatu
re_2m_min":[11.1,14.1,15.8,18.4,21.2,16.8,13.4]}}
```

JSON

- `response = requests.get(url, params=params)`
- `data = response.json()`


```
{
    "daily": {
        "time": ["2025-03-23","2025-03-24","2025-03-25","2025-03-
26","2025-03-27","2025-03-28","2025-03-29"],
        "temperature_2m_max": [28.7,29.6,30.3,31.7,30.9,23.7,17.2],
        "temperature_2m_min": [11.1,14.1,15.8,18.4,21.2,16.8,13.4]
    }
}
```

```
data["daily"]["time"][0]
```

Python程式

```
import requests
import pandas as pd
url = "https://api.open-meteo.com/v1/forecast"
params = {
    "latitude": 25.0330,
    "longitude": 121.5654,
    "daily": "temperature_2m_max,temperature_2m_min"
}
response = requests.get(url, params=params)
data = response.json()
```

Python程式

```
d = pd.DataFrame({
    "日期": data["daily"]["time"],
    "最高溫度 (°C)": data["daily"]["temperature_2m_max"],
    "最低溫度 (°C)": data["daily"]["temperature_2m_min"]
})
print("台北市一週天氣預報:")
print(d)
```

READ_HTML (1)

```
import pandas as pd
url = "https://www.tiobe.com/tiobe-index/"
d = pd.read_html(url, encoding="utf-8")
print(d[0])
```

```
# First table
```

READ_HTML (2)

- `import pandas as pd`
- `url = "https://rate.bot.com.tw/xrt?Lang=zh-TW"`
- `data = pd.read_html(url, encoding="utf-8")`
- `d = data[0]`
- `print(d)`
- `d = d[[d.columns[0], d.columns[2]]]`
- `d.columns = ["幣別", "本行賣出"]`
- `d.set_index("幣別", inplace=True)`
- `print(d)`

	幣別		現金匯率			Unnamed: 4_level_0
	幣別	Unnamed: 1_level_1	本行買入	本行賣出	本行買入	
0	美金 (USD) 美金 (USD)		27.27	27.94	27.595	27.745
1	港幣 (HKD) 港幣 (HKD)		3.394	3.598	3.515	3.585
2	英鎊 (GBP) 英鎊 (GBP)		36.69	38.81	37.585	38.215
3	澳幣 (AUD) 澳幣 (AUD)		19.74	20.52	19.955	20.3

READ_CSV

- `import pandas as pd`
- `url = "https://rate.bot.com.tw/xrt/flcsv/0/day"`
- `d = pd.read_csv(url)`
- `d.set_index("幣別", inplace=True)`
- `print(d)`

Problem

- 自己找一個網頁下載資料，利用Pandas整理資料，最後輸出結果

- 科學數據：
- 電影時刻表：
- 一週預報(XHR)：https://www.cwa.gov.tw/V8/C/W/week.html
- 空氣品質指標(AQI)：https://data.gov.tw/dataset/40448
- 統一發票：https://invoice.etax.nat.gov.tw/index.html
- 臺灣銀行牌告匯率網頁：https://rate.bot.com.tw/xrt