巨集程式與資料分析應用
Macro programming and data analytics

Section 3
網路爬蟲與資料分析
Lecture 2

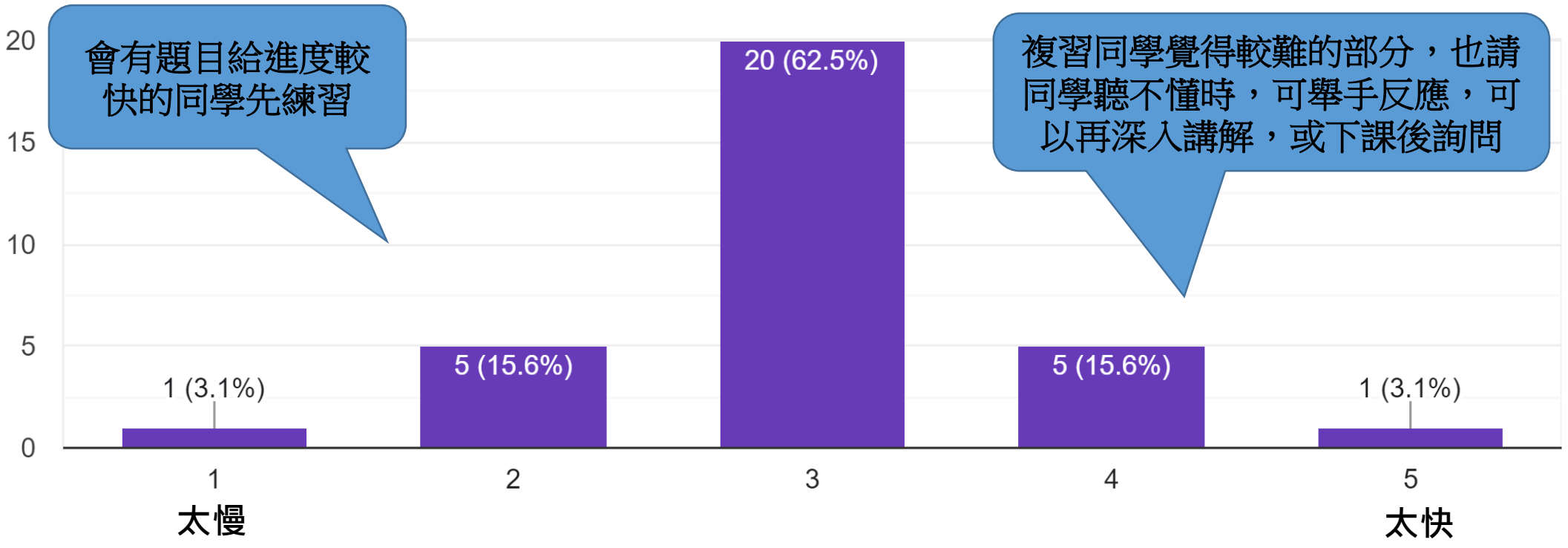
陳彥佑

課後問卷內容反饋

請問本日課程的講課速度是否恰當?

複製

32 則回應



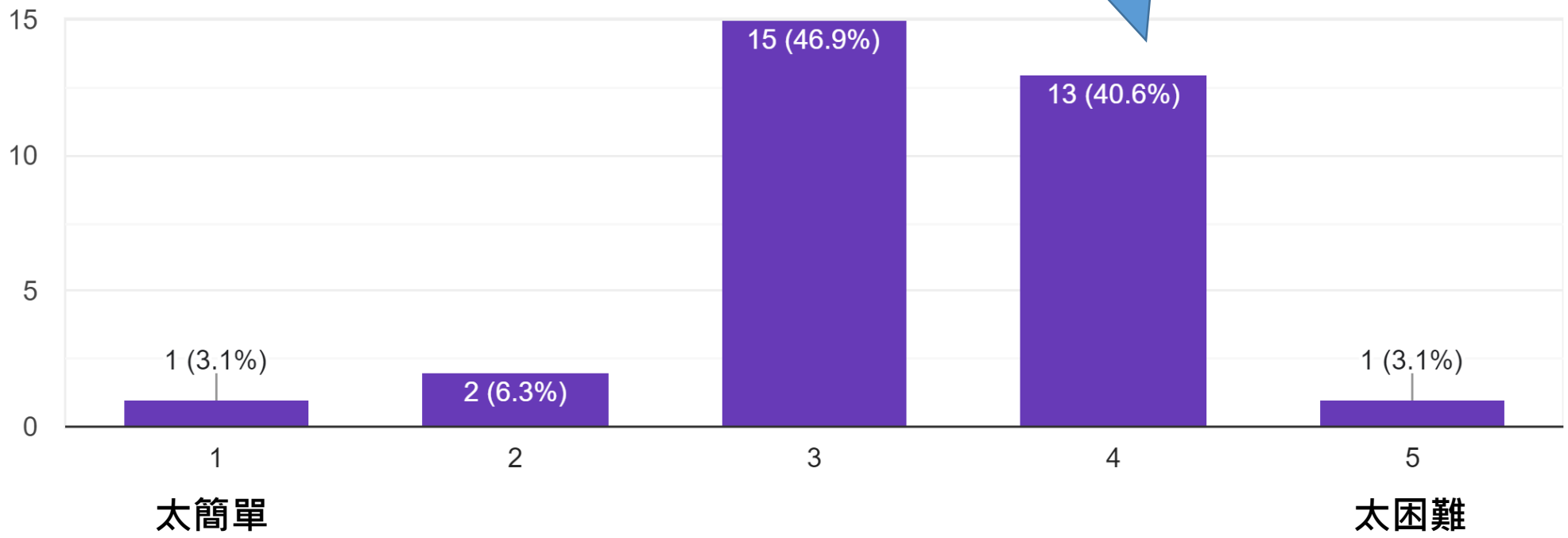
課後問卷內容反饋

會針對同學反應較難或講得比較快的部分，再講解一次

製

請問本日課程的難易程度?

32 則回應



HW1檢討

```
download_path = "https://tisvcloud.freeway.gov.tw/history/TDCS/M03A/" #for M03A
```

```
url = "https://tisvcloud.freeway.gov.tw/history/TDCS/M03A/" + year_str +
month_str + day_str + "/"+  hour_str +"/TDCS_M03A_" + year_str + month_str +
day_str + "_" + hour_str + min_str + "00.csv"
```

```
with open(file_name, "wb") as file:
    file.write(response.content)
```

VD資料下載– 多個檔案

- 利用for迴圈變動下載網頁連結與儲存位置

需變動之處

```
#指定下載的網頁連結
url = "https://tisvcloud.freeway.gov.tw/history/motc20/VD/20231024/VDLive_2357.xml.gz"
```

```
# 指定儲存檔案的路徑和檔名（包括附檔名）
file_name = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/VDLive_202310242357.xml.gz"
```

```
#指定gz文件檔名
gz_file_name = "VDLive_202310242357.xml.gz"
uncompressed_file_name = "VDLive_202310242357.xml"
```

VD資料下載– 多個檔案

For (變動日期、時間):

```
url = "https://tisvcloud.freeway.gov.tw/history/motc20/VD/" + year_str + month_str + day_str + "/VDLive_"+  hour_str + min_str + ".xml.gz"
file_name = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/VDLive_" + year_str + month_str + day_str + hour_str + min_str + ".xml.gz"

response = requests.get(url)


    with open(file_name, "wb") as file:  #這個 with 區塊打開名為 file_path 的檔案以供寫入，並將該檔案的參考指派給名為 file 的變數，當 with 區塊執行完畢時，Python 會自動關閉 f
        file.write(response.content)

        #==========================================解壓縮單個.ga檔案，並重新命名==========================================
        # 設定.gz文件的路徑
        gz_file_path = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/"  # 換成你的.rar文件的存放路徑

        # 設定解縮縮文件的路徑
        extracted_folder_path = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files unzip1/"  # 替換成你想要的解壓縮後，存放解壓縮文件的文件夾


        # 檢查文件夾是否存在，若無，則建立之
        if not os.path.exists(extracted_folder_path):
            os.makedirs(extracted_folder_path)


        #指定gz文件檔名
        gz_file_name = "VDLive_" + year_str + month_str + day_str + hour_str + min_str +".xml.gz"
        uncompressed_file_name = "VDLive_"+ year_str + month_str + day_str + hour_str + min_str +".xml"

        # 開啟.gz文件並解縮縮至目標文件夾，且更改名稱
        with gzip.open(os.path.join(gz_file_path, gz_file_name), 'rb') as gz_file, open(os.path.join(extracted_folder_path, uncompressed_file_name), 'wb') as uncompre
            shutil.copyfileobj(gz_file, uncompressed_file)
        #print(uncompressed_file_name + " 解壓縮並稱改名稱完成")
        #==========================================解壓縮單個.ga檔案，並重新命名==========================================
```

單一檔案的程式碼

加上for loop

置換檔案讀取路徑與儲存路徑

```
for year_index in range(year_start, year_end + 1):
    for month_index in range(start_month, end_month + 1):
        for day_index in range(start_day, end_day + 1):
            for hour_index in range(start_hour, end_hour + 1):
                for min_index in range(start_min, end_min + 1):
                    url = "https://tisvcloud.freeway.gov.tw/history/motc20/VD/" + year_str + month_str + day_str + "/VDLive_"+  hour_str + min_str + ".xml.gz"
                    file_name = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/VDLive_" + year_str + month_str + day_str + hour_str + min_str + ".xml.gz"

                    response = requests.get(url)

                    if response.status_code == 200:

                        with open(file_name, "wb") as file:  #這個 with 區塊打開名為 file_path 的檔案以供寫入，並將該檔案的參考指派給名為 file 的變數。當 with 區塊執行完畢時，Python
                            file.write(response.content)

                            #==========================================解壓縮單個.ga檔案，並重新命名==========================================
                            # 設定.gz文件的路徑
                            gz_file_path = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/"  # 換成你的.rar文件的存放路徑

                            # 設定解縮縮文件的路徑
                            extracted_folder_path = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files unzip1/"  # 替換成你想要的解壓縮後，存放解壓縮文件的文件夾


                            # 檢查文件夾是否存在，若無，則建立之
                            if not os.path.exists(extracted_folder_path):
                                os.makedirs(extracted_folder_path)
```

加上for loop

置換檔案讀取路徑與儲存路徑

```
for year_index in range(year_start, year_end + 1):

    year_str = str(year_index)

    for month_index in range(start_month, end_month + 1):
        if month_index <= 9:
            month_str = "0" + str(month_index)
        else:
            month_str = str(month_index)

        for day_index in range(start_day, end_day + 1):
            if day_index <= 9:
                day_str = "0" + str(day_index)
            else:
                day_str = str(day_index)

            for hour_index in range(start_hour, end_hour + 1):
                if hour_index <= 9:
                    hour_str = "0" + str(hour_index)
                else:
                    hour_str = str(hour_index)

                for min_index in range(start_min, end_min + 1):
                    if min_index <= 9:
```

VD資料下載– 多個檔案

加上for loop

置換檔案讀取路徑與儲存路徑

hour_str

- 202310242357

year_str day_str month_str min_str

```
#指定下載的網頁連結
url = "https://tisvcloud.freeway.gov.tw/history/motc20/VD/20231024/VDLive_2357.xml.gz"
```

```
# 指定儲存檔案的路徑和檔名（包括附檔名）
file_name = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/VDLive_202310242357.xml.gz"
```

```
url = "https://tisvcloud.freeway.gov.tw/history/motc20/VD/" + year_str + month_str + day_str + "/VDLive_"+ hour_str + min_str + ".xml.gz"
file_name = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files/VDLive_" + year_str + month_str + day_str + hour_str + min_str + ".xml.gz"
```

加上for loop

置換檔案讀取路徑與儲存路徑

```
#指定gz文件檔名
gz_file_name = "VDLive_202310242357.xml.gz"
uncompressed_file_name = "VDLive_202310242357.xml"
```

```
#指定gz文件檔名
gz_file_name = "VDLive_" + year_str + month_str + day_str + hour_str + min_str +".xml.gz"
uncompressed_file_name = "VDLive_"+ year_str + month_str + day_str + hour_str + min_str +".xml"
```

VD資料讀取– 單個檔案

VD資料讀取– 單個檔案(存在矩陣備用)

選擇要讀取的檔案

宣告儲存VD的矩陣

```
xml_file_path = "D:/2_Class/巨集程式/Program/PythonApplication1/VD files unzip/"
xml_file_name = "VDLive_202310242357.xml"
```

打開XML文件

讀取XML檔案

VD資料讀取– 單個檔案

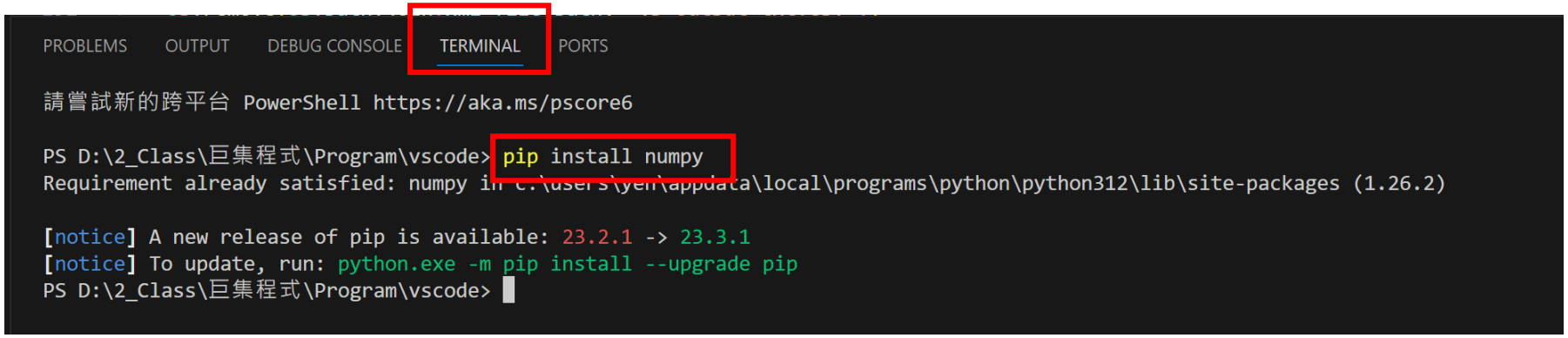
選擇要讀取的檔案

宣告儲存VD資料的矩陣

打開XML文件

讀取XML檔案

- 使用到的module：numpy (需要安裝，安裝後import)



- numpy.zeros() 是 NumPy 庫中的一個函數，用於建立一個指定大小的全零陣列

```
#宣告陣列來存放資料
speed_matrix = numpy.zeros(4)
occ_matrix =  numpy.zeros(4)
vol_matrix = numpy.zeros((4, 3)) #[lane_index][vehicle_type_index]
speed_matrix_by_veh_typ = numpy.zeros((4,3))
```

VD資料讀取– 單個檔案

選擇要讀取的檔案

宣告儲存VD資料的矩陣

- 讀取xml的module
  - xml.etree.ElementTree (不須安裝，可直接import)
    - Python中，用來處理 XML 的module，為一種簡單且有效，用來解析和操作XML檔案的module
  - 基本的用法

得到整個XML檔案的樹狀結構及其內容

```
import xml.etree.ElementTree as ET
tree = ET.parse("file.xml")
```

查找所有往後階層中<nodename>的元素

```
tree.findall("nodename",namespace)
```

查找第一個子階層中，第一次出現的 <nodename>元素

```
tree.find("nodename",namespace)
```

```
<?xml version="1.0" encoding="UTF-8"?>
<VDLiveList xsi:schemaLocation="http://traffic.transportdata.tw/standard/traffic/schema/" xmlns
  <UpdateTime>2023-09-11T23:56:00+08:00</UpdateTime>
  <UpdateInterval>60</UpdateInterval>
  <AuthorityCode>NFB</AuthorityCode>
  <LinkVersion>23.05.1</LinkVersion>
  <VDLives>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
```

```
  <VDLives>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
      <VDID>VD-N3-S-152.800-M-RS</VDID>
      <LinkFlows>
      <Status>0</Status>
      <DataCollectTime>2023-09-11T23:54:00+08:00</DataCollectTime>
    </VDLive>
    <VDLive>
    <VDLive>
    <VDLive>
```

選擇要讀取的檔案

宣告儲存VD資料的矩陣

打開XML文件

讀取XML檔案

```
namespace = {
    "ns": "http://traffic.transportdata.tw/standard/traffic/schema/"
}
```

```
os.path.join(xml_file_path, xml_file_name)
```

```
# 打開XML文件
tree = ET.parse(os.path.join(xml_file_path, xml_file_name))
vd_lives = tree.findall("./ns:VDLives/ns:VDLive", namespaces=namespace)
```

```
"./ns:VDLives/ns:VDLive"
```

- Xpath的表達式
  - 查找特定命名空間下的所有 VDLive 元素。
  - . 表示當前的node，一開始的node會在root node
  - / 表示只找下一層的子節點
  - // 表示從當前節點向下遞迴查找符合條件的元素
  - ns:VDLives 和 ns:VDLive 是指定的命名空間下的標籤名稱，ns 是命名空間首碼，在代碼的其他地方應該有對應的命名空間映射，將 ns 映射到命名空間的 URI

```
# 打開XML文件
tree = ET.parse(os.path.join(xml_file_path, xml_file_name))
vd_lives = tree.findall("./ns:VDLives/ns:VDLive", namespaces=namespace)

# 掃描所有VDLive元素
for vd_live in vd_lives:
```

打開XML文件

讀取XML檔案

掃描所有的VDLive

找出我們要的VDID

讀取該VD的內容

```
vd_live.find("./ns:VDID", namespaces=namespace).text
```

讀取VDID的文字

```
# 打開XML文件
tree = ET.parse(os.path.join(xml_file_path, xml_file_name))
vd_lives = tree.findall("./ns:VDLives/ns:VDLive", namespaces=namespace)

# 掃描所有VDLive元素
for vd_live in vd_lives:
    vd_id = vd_live.find("./ns:VDID", namespaces=namespace).text
    if vd_id == "VD-N1-N-92.900-M-LOOP":
```

讀取XML檔案

掃描所有的VDLive

找出我們要的VDID

讀取該VD的內容

如何找出VDID?

```
  7  <VDLives>
  8    <VDLive>
 43    <VDLive>
 78    <VDLive>
205    <VDLive>
240    <VDLive>
344    <VDLive>
345      <VDID>VD-N3-S-152.800-M-RS</VDID>
346      <LinkFlows>
422      <Status>0</Status>
423      <DataCollectTime>2023-09-11T23:54:00+08:00</DataCollectTime>
424    </VDLive>
425    <VDLive>
506    <VDLive>
541    <VDLive>
```

```
344    <VDLive>
345      <VDID>VD-N3-S-152.800-M-RS</VDID>
346      <LinkFlows>
347        <LinkFlow>
348          <LinkID>0000300015200K</LinkID>
349          <Lanes>
420        </LinkFlow>
421      </LinkFlows>
422      <Status>0</Status>
423      <DataCollectTime>2023-09-11T23:54:00+08:0
```

```
344    <VDLive>
345      <VDID>VD-N3-S-152.800-M-RS</VDID>
346      <LinkFlows>
347        <LinkFlow>
348          <LinkID>0000300015200K</LinkID>
349          <Lanes>
350            <Lane>
373            <Lane>
396            <Lane>
419          </Lanes>
420        </LinkFlow>
421      </LinkFlows>
422      <Status>0</Status>
423      <DataCollectTime>2023-09-11T23:54:00+08:00</DataCollect
424    </VDLive>
425    <VDLive>
```

```
344     <VDLive>
345       <VDID>VD-N3-S-152.800-M-RS</VDID>
346       <LinkFlows>
347         <LinkFlow>
348           <LinkID>0000300015200K</LinkID>
349           <Lanes>
350             <Lane>
373             <Lane>
396             <Lane>
419           </Lanes>
420         </LinkFlow>
421       </LinkFlows>
422       <Status>0</Status>
423       <DataCollectTime>2023-09-11T23:54:00+08:00</DataCollectTime>
424     </VDLive>
425     <VDLive>
```

```
344     <VDLive>
345       <VDID>VD-N3-S-152.800-M-RS</VDID>
346       <LinkFlows>
347         <LinkFlow>
348           <LinkID>0000300015200K</LinkID>
349           <Lanes>
350             <Lane>
351               <LaneID>0</LaneID>
352               <LaneType>2</LaneType>
353               <Speed>128</Speed>
354               <Occupancy>1</Occupancy>
355               <Vehicles>
356                 <Vehicle>
361                 <Vehicle>
366                 <Vehicle>
371               </Vehicles>
372             </Lane>
373             <Lane>
396             <Lane>
419           </Lanes>
420         </LinkFlow>
421       </LinkFlows>
422       <Status>0</Status>
423       <DataCollectTime>2023-09-11T23:54:00+08:00</DataCollectTime>
```

轉換成int型態

讀取LaneID的文字

```
int(lane.find("ns:LaneID", namespaces=namespace).text)
```

```
# 打開XML文件
tree = ET.parse(os.path.join(xml_file_path, xml_file_name))
vd_lives = tree.findall("./ns:VDLives/ns:VDLive", namespaces=namespace)

# 掃描所有VDLive元素
for vd_live in vd_lives:
    vd_id = vd_live.find("./ns:VDID", namespaces=namespace).text
    if vd_id == "VD-N1-N-92.900-M-LOOP":
        lanes = vd_live.findall("./ns:LinkFlows/ns:LinkFlow/ns:Lanes/ns:Lane", namespac
        for lane in lanes:
            lane_id = int(lane.find("ns:LaneID", namespaces=namespace).text)
```

讀取XML檔案

VDLive

找出我們要的VDID

讀取該VD的內容

```
if vd_id == "VD-N1-N-92.900-M-LOOP":
        lanes = vd_live.findall("./ns:LinkFlows/ns:LinkFlow/ns:Lanes/ns:Lane", namespa
        for lane in lanes:
            lane_id = int(lane.find("ns:LaneID", namespaces=namespace).text)
            speed_by_lane = int(lane.find("ns:Speed", namespaces=namespace).text)
            occupancy = int(lane.find("ns:Occupancy", namespaces=namespace).text)
            speed_matrix[lane_id] = speed_by_lane #speed_matrix[車道]
            occ_matrix[lane_id] = occupancy #occ_matrix[車道]
```

讀取XML檔案

掃描所有的VDLive

找出我們要的VDID

讀取該VD的內容

```
344    <VDLive>
345      <VDID>VD-N3-S-152.800-M-RS</VDID>
346      <LinkFlows>
347        <LinkFlow>
348          <LinkID>0000300015200K</LinkID>
349          <Lanes>
350            <Lane>
373            <Lane>
396            <Lane>
419          </Lanes>
420        </LinkFlow>
421      </LinkFlows>
422      <Status>0</Status>
423      <DataCollectTi
424    </VDLive>
425    <VDLive>
```

```
344    <VDLive>
345      <VDID>VD-N3-S-152.800-M-RS</VDID>
346      <LinkFlows>
347        <LinkFlow>
348          <LinkID>0000300015200K</LinkID>
349          <Lanes>
350            <Lane>
351              <LaneID>0</LaneID>
352              <LaneType>2</LaneType>
353              <Speed>128</Speed>
354              <Occupancy>1</Occupancy>
355              <Vehicles>
356                <Vehicle>
361                <Vehicle>
366                <Vehicle>
371              </Vehicles>
372            </Lane>
373            <Lane>
396            <Lane>
419          </Lanes>
420        </LinkFlow>
421      </LinkFlows>
422      <Status>0</Status>
423      <DataCollectTime>2023-09-11T23:54:00+08:
424    </VDLive>
```

```
<Lane>
  <LaneID>0</LaneID>
  <LaneType>2</LaneType>
  <Speed>128</Speed>
  <Occupancy>1</Occupancy>
  <Vehicles>
    <Vehicle>
      <VehicleType>S</VehicleType>
      <Volume>0</Volume>
      <Speed>0</Speed>
    </Vehicle>
    <Vehicle>
    <Vehicle>
```

為何要用雙斜線?

```
vehicles = lane.findall(".//ns:Vehicle", namespaces=namespace)

for vehicle in vehicles:
    vehicle_type = vehicle.find("ns:VehicleType", namespaces=namespace).text
    speed_by_veh = int(vehicle.find("ns:Speed", namespaces=namespace).text)
    volume = int(vehicle.find("ns:Volume", namespaces=namespace).text)

    #將vehicle type對應到矩陣
    if vehicle_type == "S":
        vehicle_type_index = 0
    elif vehicle_type == "L":
        vehicle_type_index = 1
    elif vehicle_type == "T":
        vehicle_type_index = 2

    speed_matrix_by_veh_typ[lane_id][vehicle_type_index] = speed_by_veh
    vol_matrix[lane_id][vehicle_type_index] = volume
```

讀取該VD的內容

輸出讀取之資料

選擇要讀取的檔案

宣告儲存VD資料的矩陣

VDID

讀取該VD的內容

- 列印出以逗點分隔之每個lane的speed資料
  VDID, lane_0, lane_1, lane_2, lane_3
  VD-N1-N-92.900-M-LOOP, 105.0, 100.0, 75.0, 0.0

```
#列印出讀取的資料(標頭)
print('VDID, ', end='')
for lane_index in range(0,4):
    if lane_index < 3:
        print('lane_'+str(lane_index)+', ', end='')
    else:
        print('lane_'+ str(lane_index))
```

輸出讀取之資料

選擇要讀取的檔案

宣告儲存VD資料的矩陣

打

讀

讀取該VD的內容

- 列印出以逗點分隔之每個lane的speed資料
  VDID, lane_0, lane_1, lane_2, lane_3
  VD-N1-N-92.900-M-LOOP, 105.0, 100.0, 75.0, 0.0

```
#列印出讀取的speed資料 (內容)
print('VD-N1-N-92.900-M-LOOP, ', end ='')
for lane_index in range(0,4):
    if lane_index < 3:
        print(str(speed_matrix[lane_index]) + ', ', end='')
    else:
        print(speed_matrix[lane_index])
```

輸出讀取之資料

- 練習1：
  - Print出每個lane、每個車種的Volume與Speed
  - VDID, Lane_0_S_spd, , Lane_0_S_vol, Lane_0_L_spd, Lane_0_L_vol, Lane_0_T_spd, Lane_0_T_vol, Lane_1_S_spd, , Lane_1_S_vol, Lane_1_L_spd, Lane_1_L_vol, Lane_1_T_spd, Lane_1_T_vol,.....

- 練習1的輸出

```
VDID,lane_0_S_spd,lane_0_S_vol,lane_0_L_spd,lane_0_L_vol,lane_0_T_spd,lane_0_T_vol,lane_1_S_spd,lane_1_S_vol,lane_1_L_spd,lane_1_L_vol,lane_1_T_spd,lane_1_T_vol,lane_2_S_spd,lane_2_S_vol,lane_2_L_spd,lane_2_L_vol,lane_2_T_spd,lane_2_T_vol,lane_3_S_spd,lane_3_S_vol,lane_3_L_spd,lane_3_L_vol,lane_3_T_spd,lane_3_T_vol
VD-N1-N-92.900-M-LOOP,104.0,3.0,0.0,0.0,0.0,0.0,100.0,6.0,0.0,0.0,99.0,1.0,75.0,2.0,74.0,3.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
```

輸出讀取之資料

- 將每個lane的speed輸出至csv檔 (方法1)

```
import csv

# 指定要寫入的路徑與檔名
file_name = 'C:/output.csv'

# 寫入CSV檔案
with open(file_name, mode='w', newline='') as file:
    writer = csv.writer(file)

    # 使用 writer.writerow 寫入單行數據
    writer.writerow(data)

    # 使用 writer.writerows 寫入多行數據
    writer.writerows(data)
```

輸出讀取之資料

- 使用numpy.concatenate連接矩陣

```
連接後的矩陣：(4x3)
[[ 0  6  5]
 [ 3  0  3]
 [ 7  8  9]
 [10 11 12]]
```

```
# Example
array1 = numpy.array([[0, 6, 5], [3, 0, 3]])
array2 = numpy.array([[7, 8, 9], [10,11,12]])

# 使用 numpy.concatenate連接矩陣
result = numpy.concatenate((array1, array2), axis=0)

print("連接後的矩陣：")
print(result)
```

輸出讀取之資料

```
連接後的矩陣：(2x6)
[[ 0  6  5  7  8  9]
 [ 3  0  3 10 11 12]]
```

- 使用numpy.concatenate連接矩陣

```
# Example
array1 = numpy.array([[0, 6, 5], [3, 0, 3]])
array2 = numpy.array([[7, 8, 9], [10,11,12]])

# 使用 numpy.concatenate連接矩陣
result = numpy.concatenate((array1, array2), axis=1)

print("連接後的矩陣：")
print(result)
```

輸出讀取之資料

- 結合VDID與speed_matrix

```
speed_matrix=[105, 100, 75, 0]
```

```
[VD-N1-N-92.900-M-LOOP, 105, 100, 75, 0]
```

```
merged_row = numpy.concatenate((['VD-N1-N-92.900-M-LOOP'], speed_matrix), axis=0)
```

輸出讀取之資料

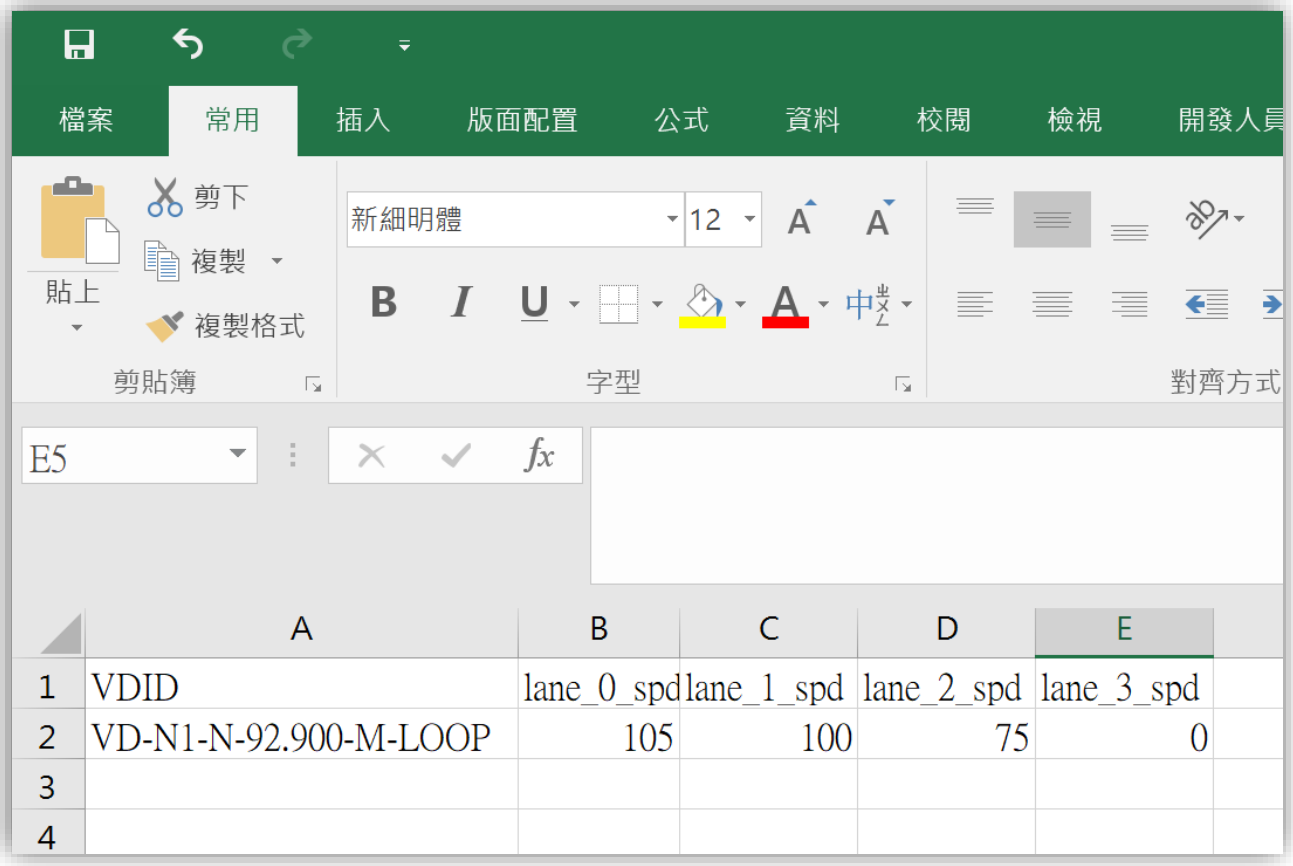
- 將每個lane的speed輸出至csv檔

```
#輸出.csv檔案
output_file_name = 'VD_output.csv'
merged_row = numpy.concatenate((['VD-N1-N-92.900-M-LOOP'], speed_matrix), axis=0)

with open(os.path.join(xml_file_path, output_file_name), mode = 'w', newline='') as outputfile:
    writer = csv.writer(outputfile)
    writer.writerow(['VDID', 'lane_0_spd', 'lane_1_spd', 'lane_2_spd', 'lane_3_spd'])
    writer.writerow(merged_row)
```

輸出讀取之資料

- 將每個lane的speed輸出至csv檔



	A	B	C	D	E
1	VDID	lane_0_spd	lane_1_spd	lane_2_spd	lane_3_spd
2	VD-N1-N-92.900-M-LOOP	105	100	75	0
3					
4					

輸出讀取之資料

- 將每個lane的speed輸出至csv檔(方法2)

'w'前不須加 mode =
不需要加newline=''

```
#輸出.txt檔，再轉csv檔
output_file_name = 'VD_output_txt.txt'

with open(os.path.join(xml_file_path, output_file_name), 'w') as outputfile:
    outputfile.write('VDID,')
    for lane_index in range(0,4):
        if lane_index < 3:
            outputfile.write('lane_'+str(lane_index)+'_spd,')
        else:
            outputfile.write('lane_'+str(lane_index) + '_spd\n')
```

輸出讀取之資料

- 將每個lane的speed輸出至csv檔(方法2)

```
    outputfile.write('VD-N1-N-92.900-M-LOOP,')
    for lane_index in range(0,4):
        if lane_index < 3:
            outputfile.write(str(speed_matrix[lane_index]) + ',')
        else:
            outputfile.write(str(speed_matrix[lane_index])+ '\n')
```

輸出讀取之資料

- 將每個lane的speed輸出至csv檔(方法2)

```
#檢查csv檔案是否存在，存在的話，先刪除該csv檔案
if os.path.exists(os.path.join(xml_file_path, 'VD_output_txt.csv')):
    os.remove(os.path.join(xml_file_path, 'VD_output_txt.csv'))
#重新命名檔案
os.rename(os.path.join(xml_file_path, output_file_name), os.path.join(xml_file_path,
'VD_output_txt.csv'))
```

os.rename(path/filename_1, path/filename_2)

path/filename_1: `os.path.join(xml_file_path, output_file_name)`

path/filename_2: `os.path.join(xml_file_path, 'VD_output_txt.csv')`

- 練習2：
  - 延續練習1，將練習1的結果輸出成.csv檔