PYTHON程式設計

PANDAS資料處理套件

吳智鴻教授
Chih-Hung Wu
國立臺中教育大學 數位內容科技學系

[D C T]
數位內容科技學系
Department of Digital Content & Technology
chwu@mail.ntcu.edu.tw
Website: chwu.weebly.com
Department of Digital Content and Technology NTCU

PANDAS好處

- 可讀取網頁的表格資料
- 可以容易地對資料進行修改、排序等處理，以及繪製統計報表

在Colab安裝

```
!pip install pandas

import pandas as pd
```

安裝完成

```
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages (1.3.5
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (fro
```

使用

```
import pandas as pd
```

建立資料
第一種型態

「資料」可有多種型態：第一種資料型態是以擁有相同數目元素串列的字典建立 DataFrame 資料，例如建立一個4 位學生，每人有 5 科成績的 DataFrame，資料變數名稱為 df：

```
df = pd.DataFrame({"林大明":[65,92,78,83,70],"陳聰明":[90,72,76,93,56],\
                   "黃美麗":[81,85,91,89,77],"熊小娟":[79,53,47,94,80]})
```

建立資料
第二種型態: 自訂行及列標題

```
資料變數 = pd.DataFrame( 資料 [, columns= 行標題串列 , index= 列標題串列 ])
```

```
df = pd.DataFrame({"林大明":[65,92,78,83,70],"陳聰明":[90,72,76,93,56],\
                   "黃美麗":[81,85,91,89,77],"熊小娟":[79,53,47,94,80]})
```

二種型態的比較

```
import pandas as pd

#第一種型態
df1 = pd.DataFrame({"林大明":[65,92,78,83,70],"陳聰明":[90,72,76,93,56],\
                    "黃美麗":[81,85,91,89,77],"熊小娟":[79,53,47,94,80]})
print("第一種型態 \n", df1)
print("\n")

#第二種型態
datas = [[65,92,78,83,70],[90,72,76,93,56],[81,85,91,89,77],[79,53,47,94,80]]
indexs = ["林大明","陳聰明","黃美麗","熊小娟"]
columns = ["國文","數學","英文","自然","社會"]
df2 = pd.DataFrame(datas, columns=columns, index = indexs)

print("第二種型態 \n")
print(df2)
```

```
第一種型態
    林大明  陳聰明  黃美麗  熊小娟
0   65   90   81   79
1   92   72   85   53
2   78   76   91   47
3   83   93   89   94
4   70   56   77   80


第二種型態

     國文  數學  英文  自然  社會
林大明  65  92  78  83  70
陳聰明  90  72  76  93  56
黃美麗  81  85  91  89  77
熊小娟  79  53  47  94  80
```

[課堂練習#1] 建立一個學生的dataFrame

- 利用pandas建立學生的成績
- 第一個人為[你的名字]

	電腦	美術	體育
吳智鴻	70	85	90
王小花	90	72	76
李大呆	85	53	77

```
學生成績

      電腦  美術  體育
吳智鴻  70  90  85
王小花  85  72  53
李大呆  90  76  77
```

解答&結果

```python
#Exercise#1

datas = [[70,90,85],[85,72,53],[90,76,77]]
indexs = ["吳智鴻","王小花","李大呆"]
columns = ["電腦","美術","體育"]
df2 = pd.DataFrame(datas, columns=columns, index = indexs)

print("學生成績 \n")
print(df2)
```

```
學生成績

     電腦  美術  體育
吳智鴻   70   90   85
王小花   85   72   53
李大呆   90   76   77
```

df.values取得資料

「df.values」可取得全部資料，是一個二維串列，執行結果為：

(<datatake2.py>)

```
[ [65 92 78 83 70]
 [90 72 76 93 56]
 [81 85 91 89 77]
 [79 53 47 94 80] ]
```

取得第2 位學生陳聰明成績的語法為：

```
df.values[1]
```

執行結果：

```
[90 72 76 93 56]
```

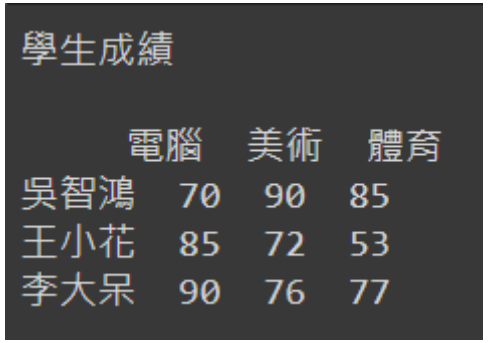
取得第2 位學生陳聰明的英文成績 (第3 個科目) 的語法為：

```
df.values[1][2]
```

執行結果為「76」。

[課堂練習#2]

- 取得你的所有成績
- 取得第二位同學的體育成績



[課堂活動2] 解答

```python
# Exercise2 取得成績

# 取得我的所有成績
print(df2.values[0])

# 取得第二位同學的體育成績
print(df2.values[1][2])
```

```
[70 90 85]
53
```

學生成績

	電腦	美術	體育
吳智鴻	70	90	85
王小花	85	72	53
李大呆	90	76	77

取得DataFrame資料

取得行資料

取得一個行資料的語法為：

```
df[ 行標題 ]
```

若要取得2 個以上行資料則需以2 個中括號包圍行標題，語法為：

```
df[[ 行標題 1, 行標題 2, ……]]
```

也可以使用行資料進行邏輯運算來取得資料，例如取得數學科成績80 分以上(含)的所有學生成績：

```
df[df.數學 >= 80]
```

df.loc 以行、列標題取得資料

使用 df.loc 的語法為：

```
df.loc[ 列標題 , 行標題 ]
```

例如取得學生陳聰明的所有成績：(<datatake3.py>)

```
df.loc[" 陳聰明 ", :]
```

取得學生陳聰明的數學科成績：

```
df.loc[" 陳聰明 "][" 數學 "]
```

取得學生陳聰明、熊小娟的所有成績：

```
df.loc[(" 陳聰明 ", " 熊小娟 "), :]
```

取得學生陳聰明、熊小娟的數學、自然科成績：

```
df.loc[(" 陳聰明 ", " 熊小娟 "), (" 數學 ", " 自然 ")]
```

取得學生陳聰明到熊小娟的數學科到社會科成績：

```
df.loc["陳聰明":"熊小娟", "數學":"社會"]
```

取得從頭到黃美麗的學生，他們的數學科到社會科成績：

```
df.loc[:"黃美麗", "數學":"社會"]
```

取得從陳聰明到最後的學生，他們的數學科到社會科成績：

```
df.loc["陳聰明":, "數學":"社會"]
```

取得最前或最後數列資料

如果要取得最前面幾列資料，可使用 head 方法，語法為：

```
df.head([n])
```

若要取得最後面幾列資料，則使用 tail 方法，語法為：

```
df.tail([n])
```

排序DataFrame資料

Pandas 提供2 種方法對DataFrame 資料排序。

第1 種是根據資料數值排序，語法為：

```
資料變數 = df.sort_values(by= 行標題 [, ascending= 布林值 ])
```

- **行標題**：做為排序值的行標題。
- **布林值**：可省略，True 表示遞增排序 (預設值)，False 表示遞減排序。

例如以數學成績做遞減排序，並將結果存於df1 中：(<datasort1.py>)

```
df1 = df.sort_values(by="數學", ascending=False)
```

第2 種是根據行、列標題排序，語法為：

```
資料變數 = df.sort_index(axis= 行列數值 [, ascending= 布林值 ])
```

■ **行列數值**：0表示依列標題排序，1 表示依行標題排序。

例如按照列標題遞增排序，並將結果存於df2 中：

```
df2 = df.sort_index(axis=0)
```

刪除DataFrame資料

Pandas 使用drop 刪除DataFrame 資料，語法為：

```
資料變數 = df.drop( 行標題或列標題 [, axis= 行列數值 ])
```

■ **行列數值**：0表示依列標題排序(預設值)，1 表示依行標題排序。

例如刪除陳聰明(列標題) 的成績：(<datadrop1.py>)

```
df1 = df.drop(" 陳聰明 ")   #axis 參數可省略
```

刪除數學科(行標題) 成績：

```
df2 = df.drop(" 數學 ", axis=1)
```

若刪除的行或列超過1 個，需以串列做為參數，例如刪除數學科及自然科成績：

```
df3 = df.drop([" 數學 ", " 自然 "], axis=1)
```

如果刪除的行或列項目很多且連續，可使用刪除「範圍」方式處理。刪除連續列的語法為：

```
資料變數 = df.drop(df.index[開始數值：結束數值][, axis=行列數值])
```

執行結果會刪除「開始數值」到「結束數值 - 1」列，例如刪除第2 列到第4 列 (陳聰明、黃美麗、熊小娟) 成績：

```
df4 = df.drop(df.index[1:4])
```

刪除連續行的語法為：

```
資料變數 = df.drop(df.columns[開始數值：結束數值][, axis=行列數值])
```

例如刪除第2 行到第4 行 (數學、英文、自然) 成績：

```
df5 = df.drop(df.columns[1:4], axis=1)
```

計算平均

- 透過numpy模組計算平均
- numpy.mean()

```
# numpy計算平均
import numpy

data = [1,2,3,4,5,6]
mean = numpy.mean(data)
print(mean)
```

```
3.5
```

[課堂練習3]

- 計算出你的各科平均
- 分別計算出三個科目的平均

	A	B	C	D	E
1		電腦	美術	體育	
2	吳智鴻	70	90	85	81.66667
3	王小花	85	72	53	70
4	李大呆	90	76	77	81
5		81.66667	79.33333	71.66667	

[課堂練習#3] 解答

	A	B	C	D	E
1		電腦	美術	體育	
2	吳智鴻	70	90	85	81.66667
3	王小花	85	72	53	70
4	李大呆	90	76	77	81
5		81.66667	79.33333	71.66667	

- 計算我的平均

```
# Exercise3 平均
import numpy

myscore=df2.loc["吳智鴻",:]
print(myscore)
print('我的平均 : ', numpy.mean(myscore))
```

```
電腦    70
美術    90
體育    85
Name: 吳智鴻, dtype: int64
我的平均 :  81.66666666666667
```

計算各科平均

```
computer_score = df2.loc[:,"電腦"]
print(computer_score ,'\n')
print('電腦科平均 : ', numpy.mean(computer_score),'\n')

art_score = df2.loc[:,"美術"]
print(art_score ,'\n')
print('美術平均 : ', numpy.mean(art_score))
```

```
吳智鴻    70
王小花    85
李大呆    90
Name: 電腦, dtype: int64

電腦科平均 :  81.66666666666667

吳智鴻    90
王小花    72
李大呆    76
Name: 美術, dtype: int64

美術平均 :  79.33333333333333
```

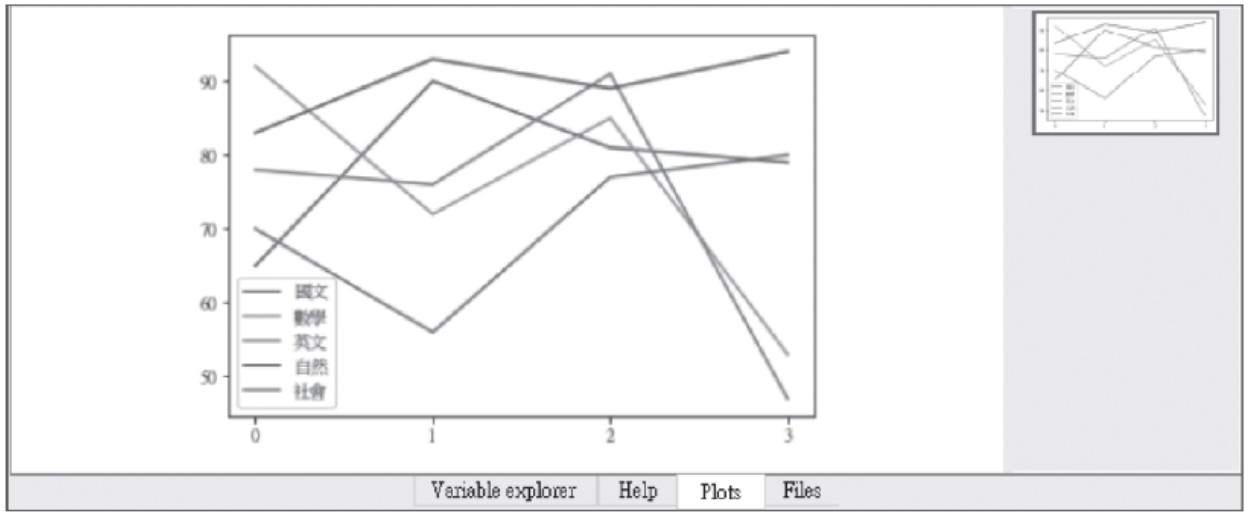
繪製線形圖

Pandas 繪製圖形功能的語法為：

```
df.plot()
```

範例：繪製學生成績線形圖

以DataFrame 資料繪製線形統計圖。

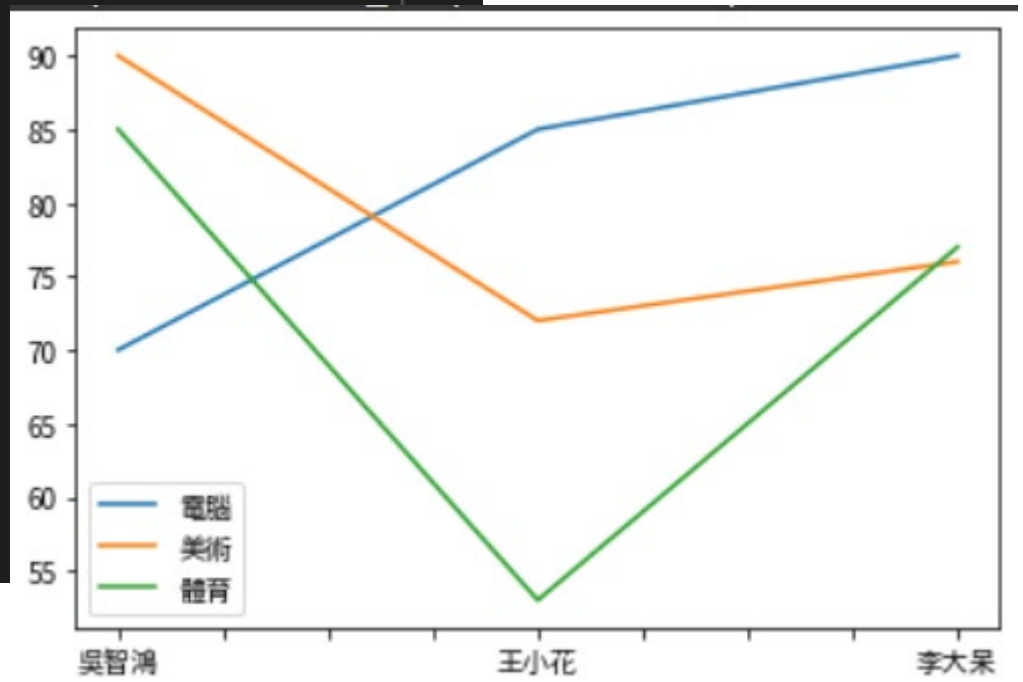


[課堂練習4] 繪製學生成績線形圖

- 繪製學生成績線形圖
- 能顯示中文

[課堂練習4]解答

```
# Colab 進行matplotlib繪圖時顯示繁體中文
# 下載台北思源黑體並命名taipei_sans_tc_beta.ttf，移至指定路徑
!wget -O TaipeiSansTCBeta-Regular.ttf https://drive.google.com/uc?id=1eGAsTN1HBpJAke

import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib.font_manager import fontManager

# 改style要在改font之前
# plt.style.use('seaborn')

fontManager.addfont('TaipeiSansTCBeta-Regular.ttf')
mpl.rc('font', family='Taipei Sans TC Beta')

df2.plot()
```



匯入資料

自行製作Pandas 的DataFrame 資料是件非常繁瑣的工作，通常是將資料存於統計軟體Excel 或資料庫中，再將資料匯入Pandas。另一種情況是擷取網頁中成千上萬的表格資料，匯入Pandas 成為DataFrame。

Pandas 常用的匯入資料方法有：

方法	說明
read_csv	匯入表格式文字資料 (*.csv)
read_excel	匯入 Microsoft Excel 資料 (*.xlsx)
read_sql	匯入 SQLite 資料庫資料 (*.sqlite)
read_json	匯入 Json 格式文字資料 (*.json)
read_html	匯入網頁中表格資料 (*.html)

Pandas 的read_html 方法會使用html5lib 模組，在Anaconda Prompt 中以下列命令安裝：

```
conda install html5lib
```

[課堂練習#5]

- 製作一個學生成績EXCEL檔
- 將EXCEL檔上傳colab
- 讀入此EXCEL檔

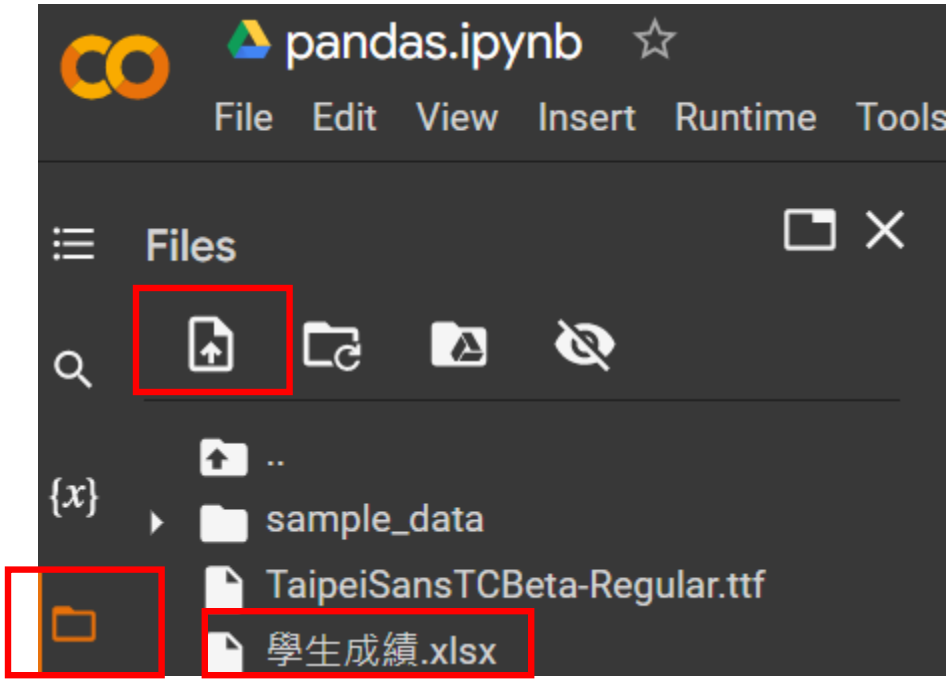
- 抓取王小花的所有成績

	A	B	C	D	E
1		電腦	美術	體育	
2	吳智鴻	70	90	85	
3	王小花	85	72	53	
4	李大呆	90	76	77	
5					

讀取EXCEL資料

	A	B	C	D	E
1		電腦	美術	體育	
2	吳智鴻	70	90	85	
3	王小花	85	72	53	
4	李大呆	90	76	77	
5					

- 先將打好的EXCEL上傳




```
data = pd.read_excel("學生成績.xlsx")

print(data)
```

```
  Unnamed: 0  電腦  美術  體育
0        吳智鴻  70  90  85
1        王小花  85  72  53
2        李大呆  90  76  77
```

[課堂練習5] 解答

```python
# Exercise 5 讀取EXCEL檔
import pandas as pd
import numpy

data = pd.read_excel("學生成績.xlsx")

print(data)

print('\n')


print(data.iloc[1])
```

```
  Unnamed: 0  電腦  美術  體育
0         吳智鴻  70  90  85
1         王小花  85  72  53
2         李大呆  90  76  77


Unnamed: 0    王小花
電腦              85
美術              72
體育              53
Name: 1, dtype: object
```

[課堂練習#6] 計算平均

- 計算每個科目平均
- 計算每個學生成績平均

[課堂練習6解答] 科目平均

```python
# Exercise 6 計算平均
import pandas as pd
import numpy

data = pd.read_excel("學生成績.xlsx")
print(data)
print('\n')

# 計算科目平均
course_avg = numpy.mean(data.iloc[:])
print('科目平均')
print(course_avg)
```

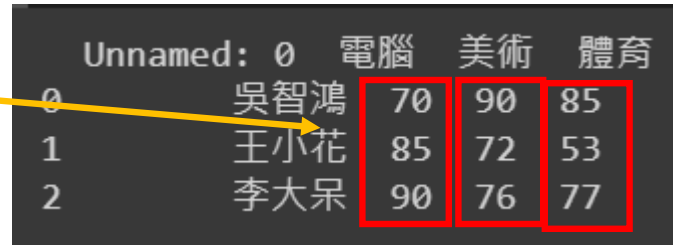
```
  Unnamed: 0  電腦  美術  體育
0        吳智鴻  70  90  85
1        王小花  85  72  53
2        李大呆  90  76  77


科目平均
電腦    81.666667
美術    79.333333
體育    71.666667
```

也可以直接用pandas的平均功能來求解

- mean（ ）裡面axis表示的是座標軸，
  - axis = 0表示對縱向求平均值，
  - axis = 1表示對橫向求平均值
- data.mean(axis = 1) # 對橫向

	Unnamed: 0	電腦	美術	體育
0	吳智鴻	70	90	85
1	王小花	85	72	53
2	李大呆	90	76	77

- data.mean(axis = 0) # 對縱向



	Unnamed: 0	電腦	美術	體育
0	吳智鴻	70	90	85
1	王小花	85	72	53
2	李大呆	90	76	77

利用data.mean求解

```
  Unnamed: 0  電腦  美術  體育
0        吳智鴻  70  90  85
1        王小花  85  72  53
2        李大呆  90  76  77
```

```
# axis=1 橫向
print('axis = 1 橫向')
print(data.mean(axis = 1))
print('\n')

print('axis = 1 縱向')
print(data.mean(axis=0))
print('\n')
```

```
axis = 1 橫向
0    81.666667
1    70.000000
2    81.000000
dtype: float64
```

個人平均

```
axis = 1 縱向
電腦    81.666667
美術    79.333333
體育    71.666667
dtype: float64
```

每科平均

[課堂練習6 簡單版解答] 利用data.mean(axis=?)

```python
# Exercise 6 計算平均
import pandas as pd
import numpy

data = pd.read_excel("學生成績.xlsx")
print(data)
print('\n')

#
#print('axis = 1 橫向')
#print(data.mean(axis=1))
#print('\n')

#
#print('axis = 0 縱向')
#print(data.mean(axis=0))
#print('\n')

# 個人平均 axis=1 橫向
print('[ 個人平均 ] axis = 1 橫向')
student_avg = data.mean(axis=1)
print(student_avg)
print('\n')

# 科目平均 axis=0 縱向
print('[ 科目平均 ] axis = 1 縱向')
course_avg = data.mean(axis=0)
print(course_avg)
print('\n')
```

```
  Unnamed: 0  電腦  美術  體育
0        吳智鴻  70  90  85
1        王小花  85  72  53
2        李大呆  90  76  77


[ 個人平均 ] axis = 1 橫向
0    81.666667
1    70.000000
2    81.000000
dtype: float64


[ 科目平均 ] axis = 1 縱向
電腦    81.666667
美術    79.333333
體育    71.666667
dtype: float64
```

[課堂練習7] 計算平均後寫入EXCEL

- [插入欄]
- 利用df.insert(欄位號碼,欄位名稱,數值) #可插入資料在EXCEL
- 範例
  - df.insert(4, column= “個人平均", value=student_avg)

	Unnamed: 0	電腦	美術	體育
0	吳智鴻	70	90	85
1	王小花	85	72	53
2	李大呆	90	76	77

[課堂練習7] 計算平均後寫入EXCEL

- [指定位置更改dataframe資料]

- data.at[第幾列，欄位名稱] = Value

先透過 len(data)取得現在資料有幾列

```
# 寫入各科平均
data.at[end_rows, "Name"] = '各科平均'
data.at[end_rows, "電腦"] = course_avg.loc["電腦"]
data.at[end_rows, "美術"] = course_avg.loc["美術"]
data.at[end_rows, "體育"] = course_avg.loc["體育"]
```

	Name	電腦	美術	體育	個人平均
0	吳智鴻	70.000000	90.000000	85.000000	81.666667
1	王小花	85.000000	72.000000	53.000000	70.000000
2	李大呆	90.000000	76.000000	77.000000	81.000000
3	各科平均	81.666667	79.333333	71.666667	NaN

解答

Output.xlsx

```
# Exercise 7 寫入EXCEL
import pandas as pd
import numpy

data = pd.read_excel("學生成績.xlsx")
print(data)
print('\n')

# 個人平均 axis=1 橫向
print('[ 個人平均 ] axis = 1 橫向')
student_avg = data.mean(axis=1)
print(student_avg)
print('\n')

# 科目平均 axis=0 縱向
print('[ 科目平均 ] axis = 1 縱向')
course_avg = data.mean(axis=0)
print(course_avg)
print('\n')



#重新命名列名
data.columns = ['Name','電腦','美術','體育']

#取得資料共有幾列
end_rows = len(data)
print('資料共有幾列 ', end_rows, '\n')

# 寫入個人平均
data.insert(4, column="個人平均", value=student_avg)

# 寫入各科平均
data.at[end_rows, "Name"] = '各科平均'
data.at[end_rows, "電腦"] = course_avg.loc["電腦"]
data.at[end_rows, "美術"] = course_avg.loc["美術"]
data.at[end_rows, "體育"] = course_avg.loc["體育"]


print(data)

data.to_excel('output.xlsx')
```

```
  Unnamed: 0  電腦  美術  體育
0        吳智鴻  70  90  85
1        王小花  85  72  53
2        李大呆  90  76  77


[ 個人平均 ] axis = 1 橫向
0    81.666667
1    70.000000
2    81.000000
dtype: float64


[ 科目平均 ] axis = 1 縱向
電腦    81.666667
美術    79.333333
體育    71.666667
dtype: float64


資料共有幾列  3


   Name        電腦        美術        體育        個人平均
0   吳智鴻  70.000000  90.000000  85.000000  81.666667
1   王小花  85.000000  72.000000  53.000000  70.000000
2   李大呆  90.000000  76.000000  77.000000  81.000000
3  各科平均  81.666667  79.333333  71.666667        NaN
```

	A	B	C	D	E	F
1		Name	電腦	美術	體育	個人平均
2	0	吳智鴻	70	90	85	81.66667
3	1	王小花	85	72	53	70
4	2	李大呆	90	76	77	81
5	3	各科平均	81.66667	79.33333	71.66667	

資料來源

- 鄧文淵，Python初學特訓班，第四版，碁峰出版社