

多元表現 自主學習 PYTHON

OVERVIEW

課題名稱	自主學習 自學程式語言Python
課題學習規劃與期望	希望在一學期的時間內可以自學好Python的基礎語法
課題學習時間	每周兩天，一次110分鐘持續一個學期(扣除段考前兩周共約12周)
背景	之前在課程程式設計課上我學了C++，但目標是成為程式設計師的我當然想繼續往下點技能樹，於是就毅然決然地想開始學習新的程式語言Python
學習成果與心得	成功完成基礎語法的學習，另外還整理了一份紙本筆記。順利成功學習到新技能很開心，不過因為時間的關係沒有辦法再更深入研究學習，並且也沒有適合讓我施展拳腳的地方，希望未來能有機會更加專精並利用到實際的專案
反思與未來期許	希望能再更精進自己各方面的能力與知識，還有最重要的是持續使用或溫習過去有學習過的語言。
	<p>雖然對學習的結果感到滿意，但在之後重新檢視並反思我發現我有兩個很大的問題需要改善:</p> <ol style="list-style-type: none">1.筆記缺乏主題的分類 (見下一段落)2.在這次自學計畫後我缺乏持續的溫習與使用。 <p>與英文這類語言相同，程式語言一樣久沒用就很容易忘記，在打這份報告時我就突然發現自己忘了好多東西，之後申請的事弄完必須要再好好溫習，並且可以嘗試應用至有興趣的專案。</p>

筆記的檢討

圖一、圖二是我的筆記中的連續四頁，前面三頁是串列的介紹與其用法還有一些相關函式的介紹，但第四頁卻是字串的介紹與方法。這種行列式的筆記讓閱讀變得有些混亂，缺乏系統性的整理導致我再次複

習時總會需要花時間釐清這段筆記紀錄的是甚麼，希望之後的筆記能改善這一點。

1. 串列 (list): name_list = [元素1, 元素2, ..., 元素n] 可宣告空串列
要加逗號

Ex1. James = [23, 19, 22, 31, 18]
print("列印James得分=", James) \Rightarrow 列印James得分 = 23, 19, 22, 31, 18

Ex2. James = [23, 19, 22, 31, 18]
print("第1場得分", James[0]) \Rightarrow 列印第一個元素
 \uparrow James[0] \rightarrow 中間(無)空格(不)語法
(值值不可)

2. 串列切片 \rightarrow 讀取元素序列稱為子串列，使用方法如下：
name_list[start : end] : 讀取從 start 到 end 的串列元素
name_list[: n] : 取得串列前 n 個
name_list[:-n] : 取得串列前面不含最後 n 個
name_list[n:] : 取得串列索引 n 到最後 \rightarrow 同時第一個元素
name_list[-n:] : 取得串列後 n 個
name[:] : 取得所有元素

3. 串列統計資料：

最大值 max()] 可用在 unicode 碼值的最大、小值
最小值 min()
總和 sum()
Ex. James = [1, 2, 3]
print("max =", max(James))
print("min =", min(James))
print("sum =", sum(James))

4. 串列個數 len(): 可得串列元素個數

5. 更改串列元素內容：
Ex. James = [23, 24, 25]
James[0] = 22

串列可相加，乘以一個整數，相當於元素除重複次數

6. 刪除元素：

del name_list[i]
del name_list[start : end]
del name_list[start : end : step] * 每隔 step, 刪除從 start 到 end 索引
到 end 索引
EA 元素
Ex. x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
del x[0: 8: 2]
print(x) \Rightarrow 隔步區間再刪除 [1, 3, 5, 7, 8, 9, 10]

1. 補充多重指定串列

Ex1. a, b, c = 1, 2, 3, 4, 5
print(a, b, c)
 \Rightarrow 1, 2, [3, 4, 5]

Ex2. a, b, c = 1, 2, 3, 4, 5
print(a, b, c)
 \Rightarrow 1, [2, 3, 4], 5

2. 簡單物件導向觀念：有一些「方法」可供物件使用 \Rightarrow 物件.方法()

lower() : 字串 \rightarrow 小寫字
upper() : " \rightarrow 大寫字
title() : " \rightarrow 第字母大寫，其餘小寫
swapcase() : " \rightarrow 大變小，小變大
lstrip() : 刪去字串尾端多餘的空白
lstrip() : 刪去字串開始端 " " "
strip() : 刪去字串前後端 " " "
center() : 字串在指定寬度置中對齊
rjust() : " 靠右對齊
ljust() : " 靠左對齊
zfill() : 可設定字串長，原字串靠右對齊，左邊多餘空間補 0

islower() / isupper() / isdigit() / isalpha()

(列出字串是否全部大、小寫，或字、英文字組成 \rightarrow true or false)

3. 增加串列與刪除串列元素

增加 name_list.append('新增元素') \rightarrow 其它用法 = 串列A.append(串列B)

插入 insert(索引, 元素內容) * 索引是位置(索引X處插入)
刪除元素 name_list.pop(i) * i 同上，但無 i 則刪除最後一項
刪除指定值 name_list.remove(元素內容) * 有相同元素時，刪第一個出現的元素

4. 串列排序(顛倒) \rightarrow 讀取顛倒串列

name_list.reverse
Ex. num = [1, 2, 3] \rightarrow 從後向前取值(n開始)
print(num[::-1]) \rightarrow 每次向前值為 |

5. sort() 排序

name_list.sort() \rightarrow 由小到大排序
欲由大排到小 \Rightarrow 在 () 加上 reverse = true

(圖一)

1. sorted()

`new_list = sorted(name_list)` 用新串列储存变化, 用串列不动
 $\rightarrow \text{由大} \rightarrow \text{小} (+ \text{reverse} = \text{true})$

2. index()

索引值 = 串列名. index(搜寻元素名)

3. count()

次数 = 串列名. count(元素名)

4. 串中串列

Ex. `num = [1, 2, 3, [4, 5]]`
`print(num[3][0])` $\Rightarrow 4$

5. extend(): 连接两串列, 会先分解成元素再一一插入串列

6. 地址的觀念

`id()` 得到地址
 Ex. `x = 10`
`y = 10`


7. `friendsports = mysports[::]`

连接且获得两个地址不同的两个不同的串列

8. 赋值: 两串列地址相同, 变更将会影响

浅拷贝: 若 `b = a.copy()`, a, b 是独立物件, 但子物件元素指向同一物件, 也就是物件子物件会变动
 (物件会联动!) example 贝 6-37页

深拷贝: 若 `b = deepcopy(a)`, a 和 b 及其子物件皆是独立物件, 使
 用前需加入 "import copy" 模组

$\rightarrow b = \text{copy.deepcopy}(a)$

1. 字串切片

Ex. `ser = "Hey you"`

`print(ser[0:2])`

$\Rightarrow \text{Hey}$

2. 字串索引

字符串索引

`str = "Hi"`

`print(str[0])` $\Rightarrow \text{H}$

`print(str[1])` $\Rightarrow \text{i}$

3. list(): 字串转成串列

`X = list('Hey you')`

`print(X)`

$\Rightarrow ['H', 'e', 'y', '.', ' ', 'o', 'u']$

4. 以 split() 分割字串

`str1.split()` * 以空格做分隔字元将字串拆成串列

`str2.split(ch)` * 以 ch 字元 ..

5. 串列元素组合 join()

连接字串: `join(串列)`

Ex. `path = ['D:', 'ch6', 'ch6-49.py']`

`connect = '\\'` ← 退出字元

`print(connect.join(path))`

$\Rightarrow \text{D:}\backslash\text{ch6}\backslash\text{ch6-49.py}$

6. 字串其它方法

`startswith()`: 可列出字串 [开始] 文字是否是特定子串串
`endswith()`:

`replace(ch1, ch2)`: 将 ch1 字串以 ch2 取代

7. "in" and "not in" (字元是否在字串内)

`boolean_value = obj1 in obj2` \Rightarrow 会回 true or false

`boolean_value = obj1 not in obj2`

8. "is" and "is not" (两物件地址是否相同)

`boolean_value = obj1 is obj2` \Rightarrow 同上

`boolean_value = obj1 is not obj2`

(圖二)