基礎程式設計 - Python
Week06

Po-Chieh Yu
pcyu@saturn.yzu.edu.tw

練習

- 練習: 以下為某班成績，score=[33,100,99,45,77,35,100,12,100,99,45]
- (1) 將成績由高分排到低分，並且存到sort_score串列裡。(2) 99分的同學有幾位? (3) 將補考同學的分數 59, 34, 10, 88, 90新增到sort_score的串列裡並重新排序輸出。(4)將59分的同學分數調整為60。

```
score=[33,100,99,45,77,35,100,12,100,99,45]
#利用sorted將score複製並排序存到新串列sort_score
sort_score=sorted(score,reverse=True)
print(sort_score)
#利用count計算99分有幾位並輸出
print(sort_score.count(99))
#定義新成績的串列
new_score=[59, 34, 10, 88, 90]
#利用extend將new_score的值加到舊串列sort_score裡
#注意:如果是用append會發生什麼事?!
sort_score.extend(new_score)
#再排序一次，這次直接利用sort改變sort_score的排序
sort_score.sort(reverse=True)
print(sort_score)
#利用index找出59分在串列裡的索引值，並將索引值存到fail_index裡
fail_index=sort_score.index(59)
#此時fail_index存的是59分的索引值，利用此索引值來改變分數
sort_score[fail_index]=60
#再排序一次
sort_score.sort(reverse=True)
print(sort_score)
```

字組 (Tuple)

- Tuple跟list類似，但不能改變內容
- 用()來創造tuple
- my_tuple=(‘a’,’b’,’c’)
- 若只有一個元素，結尾需要加上逗號
- thisistuple = (3,) #一個字組，裡面含一個元素3
- thisisnumber = (3) #只是數字

字組 (Tuple)

- 可以一次指派多個變數 (數量需相同)
- student=(‘PC’,’Yu’,1999,’Taoyuan’)
- (firstname,lastname,birth,city)=student
- 可以換值: (b,a)=(a,b)

字典 (Dictionary)

- 資料是可變的
- 按照對應(mapping)的方式，並利用鍵值找到對應的值
- vs. 串列: 利用索引值找到對應的內容值
- For CSE students: remember hash map?

字典 (Dictionary)

- 建立字典: dict() or {}
- 例如: price = dict() or price = {}
- price = {‘apple’ : 30, ‘orange’ : 50, ‘egg’ : ’10’}
- 每個字典的項目是鍵(key)與值(value)的搭配
- 練習: 建立你的字典

讀取字典項目

- #利用key值取得apple價格
- price_apple = price[‘apple’]
- #利用get()取得
- price_apple = price.get(‘apple’)
- 如果沒有想讀取的項目會回傳None

加入或修改字典項目

- 可以直接修改: price[‘apple’] = 10
- #利用**update**來修改或新增
- price.update({‘apple’:10})
- #利用**update**來修改或新增多個項目
- price.update({‘apple’:10, ‘lemon’:15, ’banana’:100})
- #利用**update**來合併字典
- new_food = {‘lemon’ : 30, ‘banana’ : 50, ‘beaf’ : ’300’}
- price.update(new_food)

刪除字典項目

- #利用**del**
- del price[‘apple’]
- #利用**clear**會刪除所有項目
- price.clear()

指派 vs. 複製

- 跟串列一樣，如果用等號(=)來指派， 會指向一樣的內容

- new_price = price
- new_price[‘apple’]=0
- print(price[‘apple’])=?

指派 vs. 複製

- **用copy來複製，就是新的獨立的字典**
- new_price = price.copy()
- new_price[‘beaf’] = 600
- print(new_price[‘beaf’]) = ?
- print(price[‘beaf’]) = ?

練習

- 以下為五位同學的學號與成績: s1100001/77, s1100002/34, s1100003/55, s1100004/100, s1100005/85
- (1) 請建立一個字典名叫score儲存這些學號與成績; (2) 練習利用get讀取s1100001的成績; (3)利用update來修改s1100002的成績為50分; (4) 建立一個新字典new_student包含三位新同學的學號與成績(自行定義內容); (5)將new_student的內容增加到score裡

- score={“s1100001":77,"s1100002":34,"s1100003":55,"s1100004":100,"s1100005":85}
- print(score.get("s1100001"))
- score.update({"s1100002":50})
- print(score)
- new_student={"s874085":11,"s874086":22,"s874087":100}
- score.update(new_student)
- print(score)

集合set

- 類似數學集合
- set存不重複的值
- set(‘apple’)
- set(['a', 'p', 'e', 'l'])

集合運算

- a={‘apple’,’banana’,’orange’}
- b={‘lemon’,’apple’,’egg’}

- 交集: a&b 或是 a.intersection(b)
- 聯集: a|b 或是 a.union()
- 差集: a-b 或是 a.difference(b)

布林值 (Boolean value)

- True or False
- 關係或比較運算子
- x是否等於y: x==y
- 不等於: x!=y
- 大於: x > y
- 大於等於: x >= y
- 小於: x < y
- 小於等於: x < = y

```
>>> x=8
>>> x==5
False
```

邏輯運算

- and/or/not
- x > 0 and x < 20
- x%3 or x%5
- not (x%3 or x%5)