勞動部產業新尖兵計畫

人工智慧金融應用與實務培訓班

課程模組： AI 金融科技課程 - Python 程式設計

4. 容器的處理（一）

葉建華 (Yeh, Jian-hua)

tdi.jhyeh@tdi.edu.tw
au4290@gmail.com

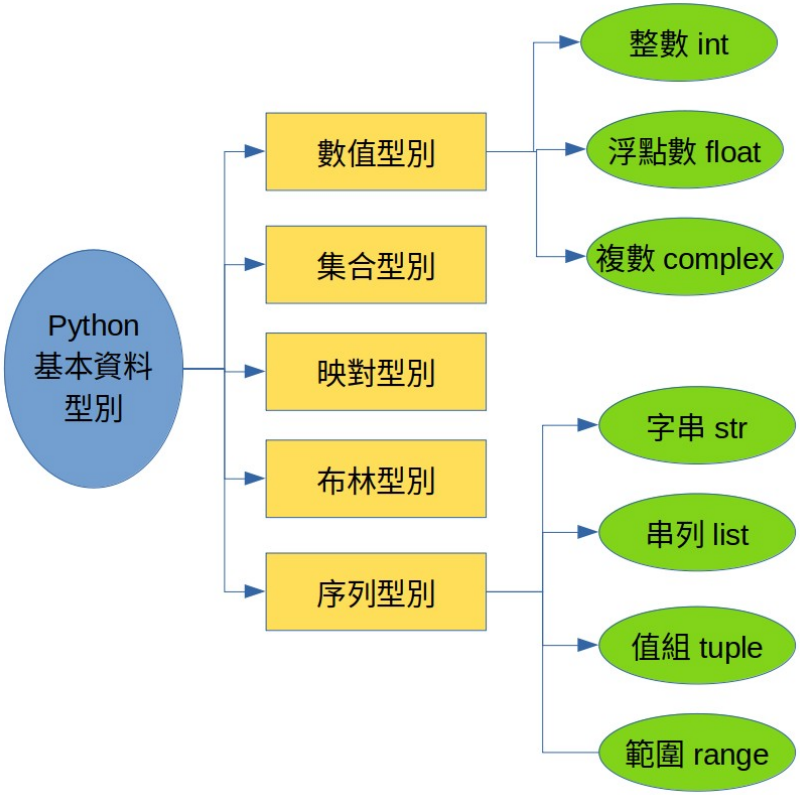
講次內容

- 序列型別
- 迭代協定
- 串列生成式

還記得資料型別嗎?

- 內建的基本資料型別
  - 數值型別： int, float, complex
  - 序列型別： list, tuple, range, str (文字型別)
  - 集合型別： set, frozenset
  - 映對型別： dict
  - 布林型別： bool

你可以使用 type() 函數來查驗資料的型別 !!



容器?

- 容器類型是指可儲存任意物件的資料型別
- 各容器之間差異是在：儲存結構（線性、對應、樹狀）、存取方式
  - 序列型別： list, tuple, str, range
  - 集合型別： set, frozenset
  - 映對型別： dict

序列型別

- 內容元素的存取方式：索引（ index ）
  - 使用方括號 [ ] 包住索引值，從 0 起算，超過容器的大小限制會出錯
  - 索引值可以是負數：負數索引被視為序列長度減回的索引位置

序列型別

- 內容元素的存取方式：索引（index）
  - 使用方
    過容器的
    大小限
  - 索引值
    度減回的
    索引位

```
# 串列資料
list1 = [3, 4, 5, 6, 7, 8]
print(list1[3])       # 會得到"5"

# 字串資料
str1 = 'abcdef'
print(str1[len(str1) - 1])       # 會得到"f"
print(str1[-1])                  # 和上一行一樣結果

# 值組資料
tup1 = (1, 3, 5, 7, 9)
print(tup1[99])       # 錯！IndexError, 索引值超出範圍
```

串列型別

- 可變的序列型別，使用指訂敘述修改容器元素

```
list1 = [3, 4, 5, 6, 7, 8]
list1[0] = 30
list1[-1] = list1[-1]*10
print(list1)                # [30, 4, 5, 6, 7, 80]

# del敘述用來刪除物件
del list1[-1]
print(list1)                # [30, 4, 5, 6, 7]
del list1[2]
print(list1)                # [30, 4, 6, 7]
```

串列型別

- 切取子序列（ slice ）
  - 索引範圍基本語法 [i:j] ， i 代表起始索引值， j 代表結束索引值 (不含)

```
list1 = [0, 1, 2, 3, 4, 5]
print(list1[0:3])       # [0, 1, 2]，不包含索引值3的元素！
print(list1[0:-1])      # [0, 1, 2, 3, 4]，沒有5哦！

tup1 = (list1[999:3], list1[3:999])     # 超過界限，視為無限遠

list1[-3:-1] = 33, 44   # 平行指定，猜猜是哪些位置？
print(list1)            # [0, 1, 2, 33, 44, 5]
```

串列型別

- 切取子序列（slice）
  - 索引範圍基本語法 [i:j]，省略 i 時預設為 0，省略 j 時預設為序列長度

```
list1 = [0, 1, 2, 3, 4, 5]
print(list1[:2])      # 從頭拿，到某處為止：[0, 1]
print(list1[2:])      # 從某處開始拿，直到最後：[2, 3, 4, 5]
print(list1[:-3])       # [0, 1, 2]
print(list1[:])         # 這樣會複製出新的list1
```

串列型別

- 切取子序列（ slice ）
  - 索引值語法 [i:j:k]， k 代表每次跳幾個元素

```
list1 = [0, 1, 2, 3, 4, 5]
print(list1[1:5:2])          # [1, 3]
print(list1[-1:1:-2])        # k是負數，由後往前數：[5, 3]
print(list1[::-1])           # 反轉整個串列：[5, 4, 3, 2, 1, 0]
a, b, c = list1[1:4]         # a=1, b=2, c=3
x, *y = list1[1::2]          # 還記得函數參數的星號嗎？ x=1, y=[3, 5]
list1[1:3] = 9, 8, 7         # 索引範圍指稱兩個，卻餵了三個，怎辦？！
print(list1)
```

串列型別

- 比較運算子
  - is、 is not： 判斷是否為同一物件
  - ==、 !=： 比較兩物件內容是否相同
  - in、 not in： 檢查是否包含元素

串列型別

- 比較運算子

```
list1 = [0, 1, 2, 3, 4]
list2 = list1[:]                # list2複製了list1

print(list1 == list2)           # 內容相同? 是的！
print(list1 is list2)           # 同一物件? 非也，是副本
print(3 in list1)               # 3這個元素在list1嗎？對！
print([3] in list1)             # [3]這個元素在list1嗎？錯！

tup1 = (0, 1, 2)
tup2 = tup1[:]                  # tup2複製了tup1，但要注意tuple不可變哦
tup3 = tuple(tup1)              # tup3複製了tup1，使用物件建構
print(tup2 is tup1)             # True!!
print(tup3 is tup1)             # True!!
```

串列型別

- 「＋」運算子代表連接

```
list1 = [0, 1, 2]
str1 = 'Hi'
print(list1 + [3, 4, 5])          # 產生新物件 [0, 1, 2, 3, 4, 5]
print(str1 + ', how are you, ' 'John?')    # 字串操作和串列相同
print(list1 + list((3, 4, 5)))  # tuple轉型為list，結果為 [0, 1, 2, 3, 4, 5]
print(list1 + 'abc')              # 牛頭不對馬嘴！TypeError！
```

串列型別

- 「*」運算子代表淺層複製 (?!)

```
list1 = ['a', [0, 1]]
list2 = list1 * 3
print(list2)          # ['a', [0, 1], 'a', [0, 1], 'a', [0, 1]]
list1[1][1] = 9       # 設定list1中元素的內容值!
print(list2)          # ['a', [0, 9], 'a', [0, 9], 'a', [0, 9]]
# 結果影響及於list2! 為什麼?
```

關鍵就在物件參考複製的層級!

淺層複製 vs. 深層複製

- 淺層複製：只及於第一層次物件參考的複製
- 深層複製：所有層次的物件參考都複製
  - 超級複雜！

遇到 import 了！ 之後再談

```
from copy import deepcopy

list1 = ['a', [0, 1]]
lisy2 = deepcopy(list1)        # list2「完整」複製了list1
list1[1][1] = 9                # list1變成是['a', [0, 9]]
print(list2)                   # list2仍舊是['a', [0, 1]]
```

「+=」、「*=」

- 對內容不可變更的資料型別，會產生新的物件

```
tup1 = tup2 = (0, 1, 2)     # tup1和tup2指向同一個tuple物件
tup1 += (3, 4)              # tuple內容不能變更，所以產生新tuple
print(tup1 is tup2)         # 比較是否是同一物件? 當然不是
# tup1是(0, 1, 2, 3, 4)，而tup2是(0,1,2)

list1 = list2 = [0, 1, 2]   # list1和list2指向同一個list物件
list1 += [3, 4]             # list內容可以變更，所以物件內容變了
print(list1 is list2)

list3 = list4 = [0, 1, 2]   # list3和list4指向同一個list物件
list3 *= 3                  # 元素內容重複三次!
print(list4)                # [0, 1, 2, 0, 1, 2, 0, 1, 2]
```

串列的內建函數

- 常用的有 len、 min、 max、 sum、還有 slice 物件

```
list1 = [1, 2, 3, 4, 5, 6, 7]
# 找最小、最大元素
print(min(list1), max(list1))
# 串列長度、串列加總
print(len(list1), sum(list1))

s0 = slice(0, 2)                    # 切片物件: 定義切片範圍
s1 = slice(1, -1, 2)
print(list1[s0], list1[s1])         # list1直接帶入切片範圍
# 結果: ([1, 2], [2, 4, 6])
```

串列的物件方法

- 可以視為是專屬於串列的函數
  - 常用的有 index、 count

```
list1 = [3, 'a', 7, 3, 'b', (1, 2), 'c']
# 找出元素所在的索引值
print(list1.index(3), list1.index(3, 1))   # (0, 3)
# 計算元素出現個數
print(list1.count(3), list1.count(4))      # (2, 0)
print(4 in list1)                          # 使用in來判斷是否有此元素
print(list1.index(4))                      # 無此元素? ValueError
```

串列的物件方法

- 還有 append、 extend、 insert、 remove、 pop、 reverse

```
list1 = [0, 1, 2]
list1.append(3)         # 加入元素
print(list1)            # [0, 1, 2, 3]
list1.extend([2, 5])    # 加入另一個list
print(list1)            # [0, 1, 2, 3, 2, 5]
list1.insert(3, 'a')    # 指定位置插入元素
print(list1)            # [0, 1, 2, 'a', 3, 2, 5]
list1.remove(2)      # 移除指定元素（第一次出現的哦！）
print(list1)            # [0, 1, 'a', 3, 2, 5]
list1.pop()             # 取出最後一個元素
print(list1)            # [0, 1, 'a', 3, 2]
list1.reverse()         # 倒過來！！
print(list1)            # [2, 3, 'a', 1, 0]
```

再來一些串列函數 ...

- range、 zip、 enumerate、 sorted、 reversed

```
print(list(range(1, 10+1, 2)))
# [1, 3, 5, 7, 9], 請注意range的中止範圍不計

print(list(zip(('a', 'b', 'c'), (30, 41, 52))))
# [('a', 30), ('b', 41), ('c', 52)]

print(list(enumerate(['a', 'b', 'c'])))
# [(0, 'a'), (1, 'b'), (2, 'c')]

print(sorted([3, 2, 9, 5]))
# [2, 3, 5, 9]

print(list(reversed([3, 2, 9, 5])))
# [5, 9, 2, 3], 但這reversed是不是很多餘！！
```

函數有很多，用多了就會熟!

練習：串列內容「併」排

- 你手中有下列資料

  names = ['Amy', 'Bob', 'Cathy']

  scores = [70, 92, 85]

- 請用上列串列做以下的輸出

  0 Amy 70

  1 Bob 92

  2 Cathy 85

- 怎麼做?

練習：串列內容「併」排

- 你手中有下列資料

  names = ['Amy', 'Bob', 'Cathy']

  scores = [70, 92, 85]

- 請用上列串列做以下的輸出

  0 Amy 70

  1 Bob 92

  2 Cathy 85

- 怎麼做?

```
names = ['Amy', 'Bob', 'Cathy']
scores = [70, 92, 85]
list1 = list(enumerate(zip(names, scores)))
# [(0, ('Amy', 70)), (1, ('Bob', 92)), (2, ('Cathy', 85))]
for item in list1:
    print(item[0], item[1][0], item[1][1])
```

講次內容

- 序列型別
- 迭代協定
- 串列生成式

迭代概念： iteration, iterable, iterator

- iterator 稱為迭代器，迭代是指提供在容器中元素之間反覆移動的功能
  - 不管該容器是否具有順序性，也不管容器中的元素是否可以重複，迭代器都提供了相同的操作方式

迭代協定

- 可迭代者 (Iterable)
  - 提供符合迭代器介面的物件，內建函式 iter
- 迭代器 (Iterator)
  - 逐一給出元素的介面，內建函式 next
- 迭代協定
  - 提供方根據協定規則提供元素
  - 接收方經由協定的介面拿到那些元素

常用迭代樣式

- 提供方：串列、 tuple 或 range
- 接收方： for 迴圈

```
list1 = [0, 1, 2]
for x in list1:
    print(x)          # 迭代提供串列元素

tup1 = (0, 1, 2)
for x in tup1:
    print(x)          # 迭代提供值組元素

rng1 = range(3)
for x in rng1:
    print(x)          # 迭代提供範圍元素
```

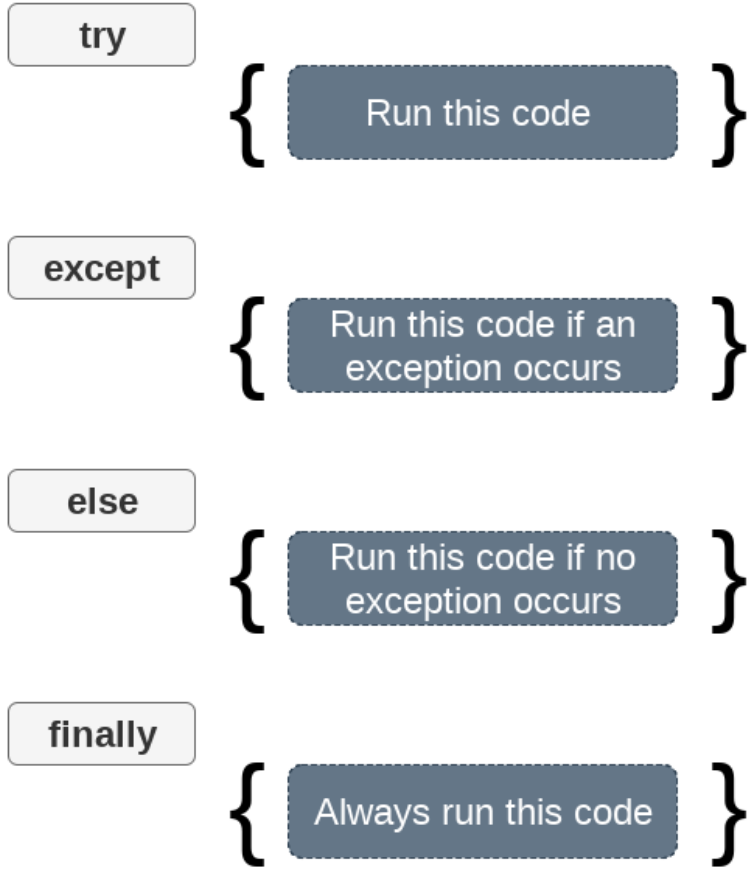
Iterable 介面

- iter： 可得到符合 Iterator 介面物件的內建函數
- next： 可從 Iterator 物件拿出元素的內建函數

```
list1 = [0, 1, 2]
it = iter(list1)
print(next(it))   # 0
print(next(it))   # 1
print(next(it))   # 2
print(next(it))   # 沒了啦! StopIteration
# 所以該怎麼寫才不會因為例外掛掉?
```

練習：例外處理運用在迭代上

- 上例中，如何使用例外處理框架避開程式中止?
- 怎麼做?

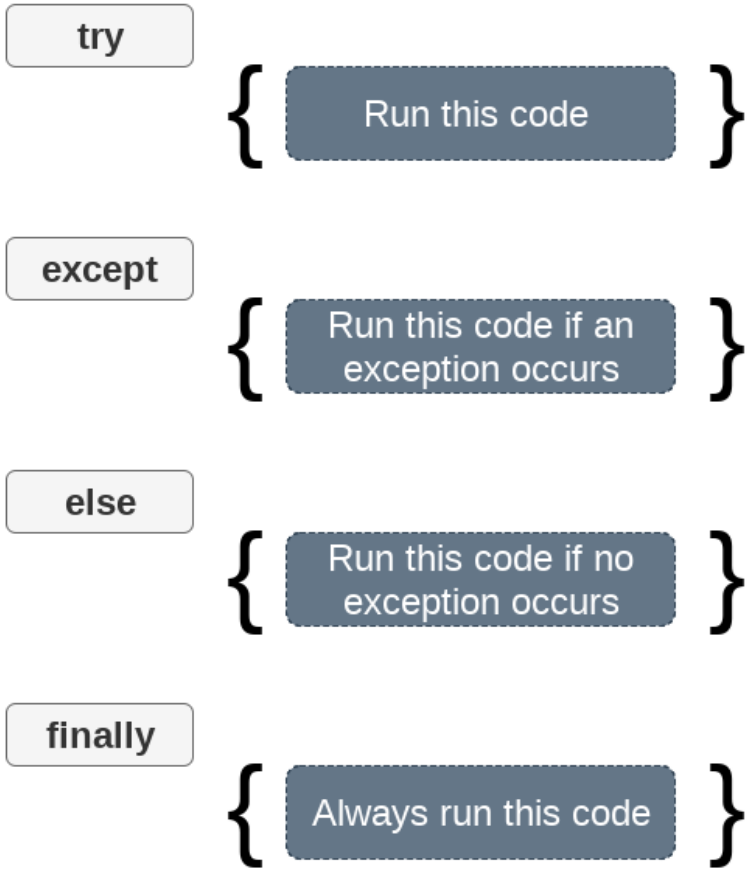


還記得這個嗎?

練習：例外處理運用在迭代上

- 上例中，如何使用例外處理框架避開程式中止?
-

```
list1 = [0, 1, 2]
it = iter(list1)
while True:
    try:
        x = next(it)      # 下一個
        print(x)
    except StopIteration: # 耗盡
        break
```



練習：運用迭代做數學運算

- 設計 prod() 函數，傳入含數字的 iterable，計算所有元素的乘積
  - 然後用它開發階乘函數 factorial()！
- 怎麼做?

練習：運用迭代做數學運算

- 設計 prod() 函數，傳入含數字的 iterable，計算所有元素的乘積
  - 然後用它開發階乘
- 怎麼做?

```
def product(itr):
    it = iter(itr)    # 取出iterable
    ret = 1
    while True:
        try:
            ret *= next(it)  # 連乘下一個!
        except:
            break
    return ret

n = 5
rng1 = range(1, n+1)
prod1 = product(rng1)
print(prod1)
```

練習：運用迭代做數學運算

- 設計 prod() 函數，傳入含數字的 iterable，計算所有元素的乘積
  - 然後用它開發階乘
- 怎麼做?

```
def product(itr):
    it = iter(itr)    # 取出iterable
    ret = 1
    while True:
        try:
            ret *= next(it)  # 連乘下一個!
        except:
            break
    return ret

def factorial(n):
    rng1 = range(1, n+1)
    prod1 = product(rng1)
    return prod1

print(factorial(5))
```

美觀又整潔!

講次內容

- 序列型別
- 迭代協定
- 串列生成式

串列生成式

- 稱為 List Comprehension
- Python 語言在設計上以 list 為主力
  - 就連簡單的迴圈都可以用生成式表達
- 生成式是一種需要經過運算的表示式
  - 最終產出是一個資料結構

串列轉換：元素轉換成階乘結果

- 將一個串列的數值內容，全部轉換成階乘的結果，以串列表示

```
list1 = [1, 3, 5, 7, 9]

# 標準作法
list2 = []
for num in list1:
    f = factorial(num)
    list2.append(f)

print(list2)   # [1, 6, 120, 5040, 362880]
```

串列轉換：元素轉換成階乘結果

- 將一個串列的數值內容，全部轉換成階乘的結果，以串列表示

```
list1 = [1, 3, 5, 7, 9]

# 標準作法
list2 = []
for num in list1:
    f = factorial(num)
    list2.append(f)

print(list2)   # [1, 6, 120,
```

```
list1 = [1, 3, 5, 7, 9]

# 寫成串列生成式，整齊美觀
list2 = [factorial(num) for num in list1]

print(list2)   # [1, 6, 120, 5040, 362880]
```

練習：處理低於 1000 元的消費

- 消費資料：[30, 45, 1024, 2500, 699, 126]
  - 計算低於 1000 元的消費總和以及平均
  - 加上條件判斷了！
- 怎麼做?

練習：處理低於 1000 元的消費

- 消費資料： [30, 45, 1024, 2500, 699, 126]
  - 計算低於 1000 元的消費總和以及平均
  - 加上條件判斷了！
- 怎麼做?

```
list1 = [30, 45, 1024, 2500, 699, 126]

# 過濾出小於1000元的消費
list2 = [num for num in list1 if num<1000]
sum1 = sum(list2)          # 用sum做消費加總
avg1 = sum1/len(list2)     # 用len取消費筆數

print(sum1)
print(avg1)
```

練習：低於 500 的數字做過濾

- 請問小於 500 且同時是 7 和 9 的倍數，列出並計算這樣的自然數總和
- 什麼是自然數?
- 怎麼做?

練習：低於 500 的數字做過濾

- 請問小於 500 且同時是 7 和 9 的倍數，列出並計算這樣的自然數總和
- 什麼是自然
- 怎麼做?

```
# 製作出小於500的自然數串列
list1 = list(range(1, 500))

# 過濾出7或9的倍數
list2 = [num for num in list1 if num%7==0 and num%9==0]

# 結案!
print(list2)
print(sum(list2))
```

是不是簡單又衛生?

雙重串列生成式

- 最具代表性的應該是九九乘法表了！
  - 請問該如何輸出九九乘法表?

```
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
2*7=14
2*8=16
2*9=18
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
```

雙重串列生成式

- 最具代表性的應該是九九乘法表了！
  - 請問該如何輸出九九乘法表?

```
# 傳統解法!
result = []
for x in range(2, 9+1):
    for y in range(1, 9+1):
        # 別忘了數字轉型才能做字串合併
        result.append(str(x)+'*'+str(y)+'='+str(x*y))

for str1 in result:
    print(str1)
```

那 ... 生成式要怎麼寫?

```
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
=14
=16
=18
=3
=6
=9
=12
=15
=18
=21
3*8=24
```

雙重串列生成式

- 最具代表性的應該是九九乘法表了！
  - 請問該如何輸出九九乘法表?

```
# 雙重串列生成
result = [str(x)+'*'+str(y)+'='+str(x*y) \
          for x in range(2, 9+1) \
          for y in range(1, 9+1)]

for str1 in result:
    print(str1)
```

```
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
2*7=14
2*8=16
2*9=18
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
```

練習：計算複利

- 假設最初存簿本金 10 萬，年複利 1.234%，請計算出第 1 到第 20 年的期末（年尾）的存簿金額
- 怎麼做?

練習：計算複利

- 假設最初存簿本金 10 萬，年複利 1.234%，請計算出第 1 到第 20 年的期末（年尾）的存簿金額
- 怎麼做?

```
# 現在國家就可以借你這麼多!
dep = 100000
rate = 1.0+0.01234    # 1.234%利率

# 把年份也製造出來!
list1 = [(i, dep*(rate**i)) for i in range(1, 20+1)]

# 輸出
for item in list1:
    print(item)
```

這個講次中，你應該學到了 ...

- 序例型別如何撰寫與運用
- 序例型別如何進行迭代
- 大家很常用的串列生成式!