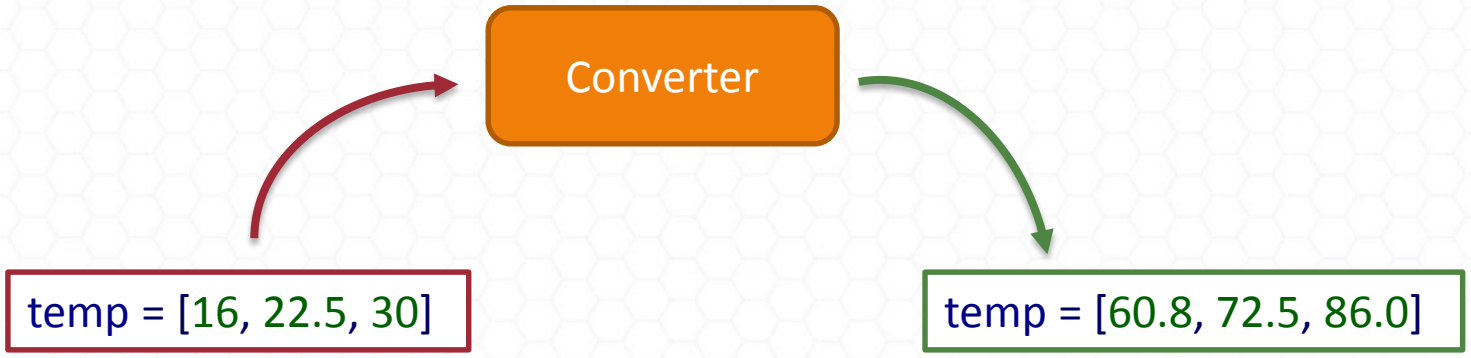
Python 語言進階議題

David Chiu

Map

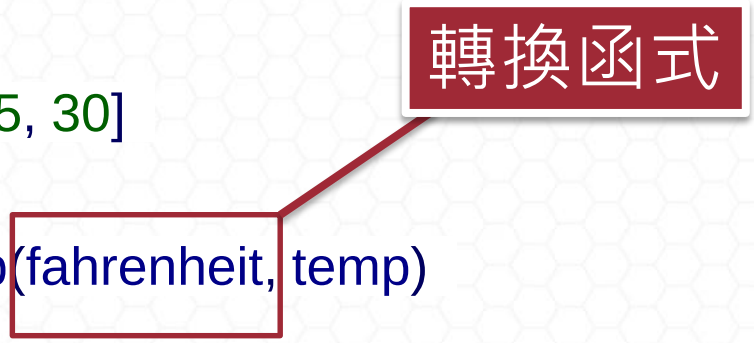
如何批次轉換所有溫度

- 除了使用迴圈以外一個個轉換溫度外，是否可以批次轉換所有資料?



map() (1/2)

```
def fahrenheit(T):
    return ((float(9)/5)*T + 32)
def celsius(T):
    return (float(5)/9)*(T-32)

temp = [16, 22.5, 30]

F_temps = map(fahrenheit, temp)
F_temps

# Convert back
map(celsius, F_temps)
map(lambda x: (5.0/9)*(x - 32), F_temps)
```



map() (2/2)

```
a = [1,2,3,4]
b = [5,6,7,8]
c = [9,10,11,12]

map(lambda x,y:x+y,a,b)

map(lambda x,y,z:x+y+z, a,b,c)
```

Reduce

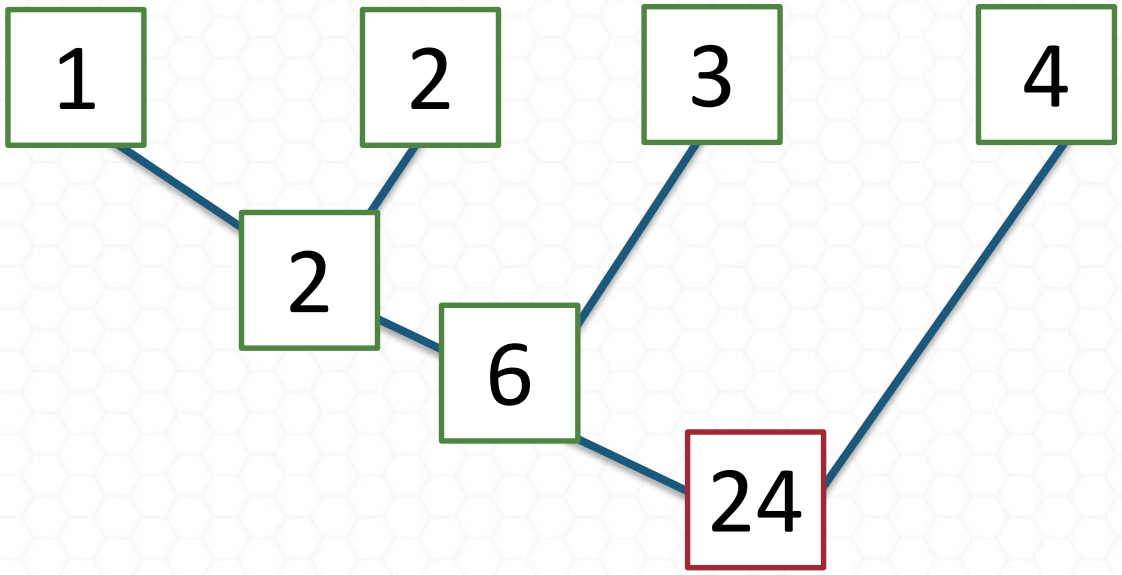
當想要合併或彙整串列中所有資料?

A = [1,2,3,4]

Reduce(1/2)

A =[1,2,3,4]

reduce(lambda x,y: x*y,A)



Reduce (2/2)

```
#找到list 中最大值
max_find = lambda a,b: a if (a > b) else b

reduce(max_find,A)
```

Filter

當想要過濾符合條件的值?

- 挑出成績大於90分的學生?

$$A = [\ 55\ ,\ 92\ ,\ 35\ ,\ 98\ ]$$

Filter(1/2)

```
def getgoodgrade(num):
    if num > 90:
        return True

print filter(getgoodgrade,A)
```

Filter(2/2)

```
def even_check(num):
    if num%2 ==0:
        return True

lst =range(20)

filter(even_check,lst)
filter(lambda x: x%2==0,lst)
```

Zip

如何合併姓名與身高資訊?

```
name   = ['David', 'John', 'Marry']
height  = [180    , 172   ,  166]
```

除了使用For 迴圈外
可以使用zip

zip 範例

```
name   = ['David', 'John', 'Marry']
height = [180    , 172   ,  166]
zip(name, height)

# 合併長度不等的list?
x = [1,2,3]
y = [4,5,6,7,8]

zip(x,y)
```

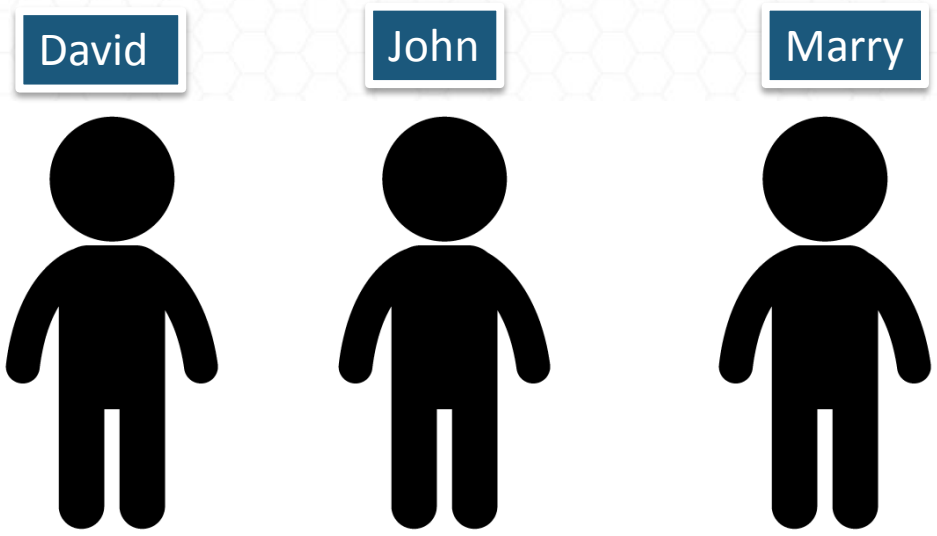
交换字典中的值

```
d1 = {'a':1,'b':2}
d2 = {'c':4,'d':5}
zip(d1,d2)
zip(d2,d1.itervalues())

def switcharoo(d1,d2):
    dout = {}
    for d1key,d2val in zip(d1,d2.itervalues()):
        dout[d1key] = d2val
    return dout

switcharoo(d1,d2)
```

Enumerate

怎麼知道資料出現在第幾個位置?

- 要怎麼知道John 出現在第二個位置?



http://goo.gl/Aq2ayC

Enumerate

```
lst = ['a','b','c']

for number,item in enumerate(lst):
    print number
    print item

for count,item in enumerate(lst):
    if count >= 2:
        break
    else:
        print item
```

all & any

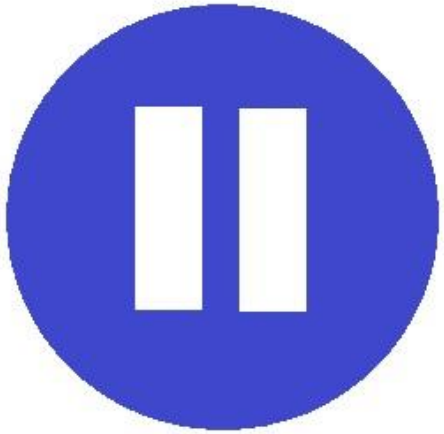
all() & any()

```
lst = [True,True,False,True]

all(lst)

any(lst)
```

Generator

Generator

- 與其將所有資料全部載入記憶體，是否可以先暫停，之後再恢復程式的執行？



使用Generator

```
def gencubes(n):
    for num in range(n):
        yield num**3

for x in gencubes(10):
    print(x)
```

比較Generator 與一般函數的差別

Generator

```
def genfibon(n):
    a = 1
    b = 1
    for i in range(n):
        yield a
        a,b = b,a+b

for num in genfibon(10):
    print(num)
```

Normal Function

```
def fibon(n):
    a = 1
    b = 1
    output = []
    for i in range(n):
        output.append(a)
        a,b = b,a+b
    return output
fibon(10)
```

Decorator

Decorators

- 修改函數功能的函數
  - 讓你的程式更Pythonic
- 通常用在開發大型系統
  - 例如用在Django 與 Flask上

回傳函數

```
def hello(name='Qoo'):
    def greet():
        return 'This is inside the greet() function'
    def welcome():
        return "This is inside the welcome() function"
    if name == 'Qoo':
        return greet
    else:
        return welcome

x = hello()
print(x())
```

將函數當成參數

```
def hello():
    return('Hi QOO!')

def other(func):
    print('Other')
    print(func())

other(hello)
```

建立 Decorator

```
def new_decorator(func):

    def wrap_func():
        print("Before func")
        func()
        print("After func()")

    return wrap_func

def func_needs_decorator():
    print("Function needs Decorator")
```

利用Decorator 修改函數

```
# Without Decorator
func_needs_decorator()

# With Decorator
func_needs_decorator = 
new_decorator(func_needs_decorator)
func_needs_decorator()
```

使用@ 描述Decorator

```
@new_decorator
def func_needs_decorator2():
    print("Function need Decorator")

func_needs_decorator2()
```

Pythonic

Pythonic

- 讓你寫的程式更加 Python
  - 要求程式碼簡練，明確，優雅
  - 提升程式碼的執行效率
  - 程式碼越少越不容易出錯

運算元比較

- Pythonic

```
a = 3
b = 1
1 <= b <= a < 10
```

- Non-Pythonic

```
b >= 1 and b <= a and a < 10
```

真假比較

- Pythonic

```
name = 'Tim'
langs = ['AS3', 'Lua', 'C']
info = {'name':'Tim', 'Sex':'Male', 'age':23}
if name and langs and info:
    print('All True')
```

- Non-Pythonic

```
if name != '' and len(langs) > 0 and info != {}:
    print('All True')
```

字串反轉

- Pythonic

```
s = 'hello world'
reverse_str = lambda s: s[::-1]
reverse_str(s)
```

- Non-Pythonic

```
def reverse_str(s):
    t = ''
    for x in range(len(s)-1, -1,-1):
        #print(x)
        t += s[x]
    return(t)
reverse_str(s)
```

字串合併

- Pythonic

```
strList = ["Python", "is", "good"]

res =  ' '.join(strList) #Python is good
res
```

- Non-Pythonic

```
res = ''
for s in strList:
    res += s + ' '
res
```

求和與乘積

Pythonic

```
numList = [1,2,3,4,5]

s = sum(numList)
maxNum = max(numList)
minNum = min(numList)
from operator import mul
from functools import reduce
prod = reduce(mul, numList, 1)
prod
```

Non-Pythonic

```
sum = 0
maxNum = -float('inf')
minNum = float('inf')
prod = 1
for num in numList:
    if num > maxNum:
        maxNum = num
    if num < minNum:
        minNum = num
    sum += num
    prod *= num
prod
```

資料篩選

- Pythonic

```
l = [x*x for x in range(10) if x % 3 == 0]
```

- Non-Pythonic

```
l = []
for x in range(10):
    if x % 3 == 0:
        l.append(x*x)
```

字典預設值

- Pythonic

```
dic = {'name':'Tim', 'age':23}

dic['workage'] = dic.get('workage',0) + 1
```

- Non-Pythonic

```
if 'workage' in dic:
    dic['workage'] += 1
else:
    dic['workage'] = 1
```

For … Else

- Pythonic

```
for x in range(1,5):
    if x == 5:
        print('find 5')
        break
else:
    print('can not find 5!')
```

- Non-Pythonic

```
find = False
for x in range(1,5):
    if x == 5:
        find = True
        print('find 5')
        break
if not find:
    print('can not find 5!')
```

條件判斷式

- Pythonic

```
a = 3

b = 2 if a > 2 else 1
```

- Non-Pythonic

```
if a > 2:
    b = 2
else:
    b = 1
```

Enumerate

- Pythonic

```
array = [1, 2, 3, 4, 5]

for i, e in enumerate(array,0):
    print(i, e)
```

- Non-Pythonic

```
for i in range(len(array)):
    print(i, array[i])
```

ZIP

- Pythonic

```
keys = ['Name', 'Sex', 'Age']
values = ['Tim', 'Male', 23]

dic = dict(zip(keys, values))
Dic
```

- Non-Pythonic

```
dic = {}
for i,e in enumerate(keys):
    dic[e] = values[i]
dic
```

THANK YOU