程式設計概論
Programming 101
—其他資料型態
(list, tuple, set)

授課老師：邱淑怡

Date:4/10/2021

Outline

- 如何完成一個完整的Python程式
- 基本資料結構
  - 序列(sequence)
    - list, tuple
  - set
  - dictionary
- 各資料結構的操作
- list comprehension

建立完整的Python 程式

- 主程式(main)
- 函式 (function)
- Save XXX.py

題目

- Write a program that lets the user perform arithmetic operations on two numbers.
- Your program must be menu driven, allowing the user to select the operation (+, -, *, or /) and input the numbers.
- Furthermore, your program must consist of following functions:

Functions

1. Function showChoice: This function shows the options to the user and explains how to enter data.
2. Function add: This function accepts two number as arguments and returns sum.
3. Function subtract: This function accepts two number as arguments and returns their difference.
4. Function multiply: This function accepts two number as arguments and returns product.
5. Function divide: This function accepts two number as arguments and returns quotient.

```
def show_choices():
    print('\nMenu')
    print('1. Add')
    print('2. Subtract')
    print('3. Multiply')
    print('4. Divide')
    print('5. Exit')

def add(a, b):
    return a + b

def subtract(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    return a / b

def main():
    while(True):
        show_choices()
        choice = input('Enter choice(1-5): ')
        if choice == '1':
            x = int(input('Enter first number: '))
            y = int(input('Enter second number: '))
            print('Sum =', add(x, y))

        elif choice == '2':
            x = int(input('Enter first number: '))
            y = int(input('Enter second number: '))
            print('Difference =', subtract(x, y))

        elif choice == '3':
            x = int(input('Enter first number: '))
            y = int(input('Enter second number: '))
            print('Product =', multiply(x, y))

        elif choice == '4':
            x = int(input('Enter first number: '))
            y = int(input('Enter second number: '))
            if y == 0:
                print('Error!! divide by zero')
            else:
                print('Quotient =', divide(x, y))

        elif choice == '5':
            break

        else:
            print('Invalid input')

main()
```

序列(sequence)

序列(sequence)

- 有順序的資料組合
- 運作類型
  - 連接運算子: +
  - 重複運算子: *
  - 比較運算子: >, <, >=, <=, ==, !=
  - In 和 not in 運算子
  - 索引與片段運算子:([start:end])指定索引範圍

list (串列)

- a=[“first”, “second”, “third”, “fourth”, 5, 6, 7]
- print(a[0]) # get “first”
- print(a[1]) # get “second”
- print(a[2]) # get “third”
- print(a[4]) # get 5

list (串列)

- 串列(list)是由一群資料排在一起形成的
  - 串列是由一連串資料所組成，有順序且可改變內容的序列
  - 定義時必須使用中括號，並在內容之間以逗點隔開
- 如何建立串列?
  - list()可建立串列：list1=list() #建立空串列
  - list2=list([1,2,3]) #建立包含 1,2,3的串列
  - list3=list(range(5)) #建立包含 0,1,2,3,4的串列
  - list4=list(range(10, -10,-2)) #建立包含 10, 8, 6, 4, 2, 0, -2, -4, -6, -8 的串列
  - list5=list("ABCDE") #建立包含'A','B','C','D','E'的串列

串列的運算

- 連接運算子: [1,2,3]+[“Taipei”,”Tokyo”,”Vienna”]
- 重複運算子: 3*[1,2,3] = [1,2,3]*3
- 比較運算子: >, <, >=, <=, ==, !=
  - [1,2,3] !=[1,2,3,4] # True
  - [1,”Hello”, “Python”] == [“Hello”,”Python”,1] #False
  - ['a','A'] < ['a','B’] #True
- 索引運算子([]):索引0表示串列第一個元素，索引-1表示串列最後一個元素
- 片段運算子: [start:end]指定索引範圍

list 的操作: 會修改該串列的內容

- a = [12, 24, 46, 18, 90, 70]
- a.append('IP') # 加入一個元素到尾端
- a.extend(['3Q']) # 把另一個list從尾端街上去, 或用a+['3Q']
- a.insert(3,' THX') # 在指定位置插入元素
- print(a)
- x = a.pop() # 移除最後一個元素並回傳
- y = a.pop(3) # 移除指定位置的元素並回傳
- print(x,y)
- a.reverse() # 反轉list a
- print(a)
- a.sort()
- print(a)

list 練習題1

- 請撰寫一個python程式，令它要求使用者輸入比賽中三位評審給每位選手的分數，然後計算總分(提示:先建立空串列，使用串列存放分數，可以參考append用法，然後計算串列中三個元素的總和，用sum 函式)

list排序

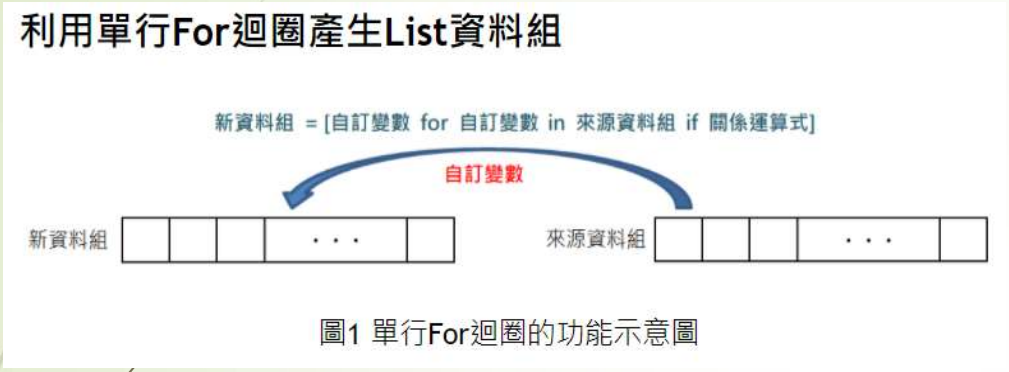
- 排序的方式使用自己定義的大小來排序
  - a1 = ['Matlab', 'C', 'Ruby', 'Java', 'Python', 'R‘, 'JavaScript']
  - b1=sorted(a1)
  - print(b1)
  - c1=sorted(a1, key=len)
  - print(c1)

list 練習題2

- 請撰寫一個python程式，先定義一個函式(tranEng)可以將傳入一個整數後進行轉換成對應的英文，然後回傳對應英文及該數值，再令使用者輸入1-10的整數，然後印出該整數對應的英文(one,two,...)，若輸入的資料不是1-10的整數，就印出'您輸入的資料超過範圍'(提示:使用串Engnum=['one','two','three','four','five','six','seven','eight','nine','ten'])

```
def tranEng(num):
    statements
    return string, value
```

串列解析(list comprehension): 提供一種更簡潔的方法建立串列



Syntax of list comprehension:
[expression for item in list]
Ex: list_num=[letter for letter in 'human']

- list_name=[自訂變數 for 自訂變數 in 資料組 (if 關係運算式)]
- newlist=[*expression* for *item* in *iterable* (if *condition* == True)]
- Example:如果要產生一個1到10的整數數列，並且存入list資料組

```
num_list=[]
for i in range(1,11)
    num_list +=[i]
```

=

```
num_list=[i for i in range(1,11)]
```

題目: 印出human的每一個字元
List comprehension vs. for loop vs. lambda function

For loop

```
h_letters = []
for letter in 'human':
        h_letters.append(letter)
print(h_letters)
```

Lambda function

```
Letters = list(map(lambda x: x, 'human'))
print(letters)
```

List comprehension

```
h_letters = [ letter for letter in 'human' ]
print( h_letters)
```

Conditionals in list Comprehension

- Using if with list comprehension
  - listA= [x for x in range(20) if x % 2 ==0]
- Nested if with list comprehension
  - listB= [y for y in range(100) if y%2==0 if y%5==0]
- if…else with list comprehension
  - listC=[“Even” if i%2==0 else “Odd” for i in range(10)]

串列解析(list comprehension)

串列的中括號裡面有一個for敘述，後面跟著0個、1個或多個for或 if敘述

- list1=[i for i in range(10)]
- print(list1)
- list2=[i*2 for i in range(10)]
- print(list2)
- list3=[i for i in range(10) if i<8]
- print(list3)
- lista=[-1,-5,-2,0,4,8]
- listb=[abs(i) for i in lista]
- print(listb)
- listc=[i for i in lista if i>=0]
- print(listc)
- listd=[i**2 for i in lista]
- print(listd)

二維串列(two-dimension list)

- 二維表格或矩陣，可用二維串列來存放
- Ex:儲存五個學生國英數成績
  Grades=[[96,65,73],[88,76,82],[92,84,89],[82,73,64],[70,83,68]]

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

對應的索引值

	國文	英文	數學
學生1	[0][0]	[0][1]	[0][2]
學生2	[1][0]	[1][1]	[1][2]
學生3	[2][0]	[2][1]	[2][2]
學生4	[3][0]	[3][1]	[3][2]
學生5	[4][0]	[4][1]	[4][2]

如何運用?

- Ex:儲存五個學生國英數成績
  Grades=[[96,65,73],[88,76,82],[92,84,89],[82,73,64],[70,83,68]]

```
#取出某個學生的全部成績或某一科成績:
Grades[0]
Grades[1]
Grades[0][0]
Grades[1][2]
```

實例說明

- 印出每位學生的總分

```
Grades =[[96,65,73],[88,76,82],[92,84,89],[82,73,64],[70,83,68]]
for i in range(5):
    subtotal=0
    for j in range(3):
        subtotal = subtotal+Grades[i][j]
    Grades[i].append(subtotal)

for i in range(5):
    print("學生", i+1, "的總分為", Grades[i][3])
```

二維串列的練習題

- 二維串列可以用來存數學的矩陣(matrix)，下面有個4*3的矩陣，請撰寫一行敘述定義一個名稱為mar1是4*3的二維串列來存放該矩陣 $\begin{bmatrix} 1 & 2 & 4 \\ 5 & 7 & 8 \\ 12 & 3 & 14 \\ 14 & 6 & 9 \end{bmatrix}$

Tuple(序對)

- 是由一連串的資料所組成，有順序且不可以改變內容的序列(sequence)
- 序對的前後以小括號標示，裡面資料以逗號隔開，資料的型別可以不同
- 建立空序對: tuple1=tuple()
- tuple2=tuple((1,2,3))=(1,2,3)
- tuple3=tuple(range(5))
- tuple4=tuple([i*2 for i in range(5)])

Tuple(序對)運作

- 注意: tuple不能改變數值，所有變更元素內容的敘述都會發生錯誤，如: T[0]=100
- 連接運算子: +
  - (1,2,3)+(“Taipei”,”Tokyo”,”Vienna”)
- 重複運算子: *
  - 3*(1,3,6)
- 比較運算子: >, <, >=, <=, ==, !=
  - (1,”Python”,”R”) == (“Python”,”R”,1) #False
  - (1,2,3) < (1,2,3,4) # True
- In 和 not in 運算子
  - “Taipei” in (1, “Taipei”, 2, “Tokyo”) # True
- 索引與片段運算子:([start,end])指定索引範圍

Tuple(序對)運作(cont.)

- 索引與片段運算子:([start:end])指定索引範圍

```
T=(5,10,15,20, 25, 30, 35, 40)
T[0] # 索引第一個元素
T[2 : 5]    #索引2到4的元素(不含索引5)
T[-1]       # 索引最後一個元素
T[6 : -1]  #索引6到-2的元素(不含索引-1)
```

SET(集合)

- 集合包含沒有順序、沒有重複且可改變內容的多個資料，概念上就像數學的集合，用大括號標示
- 集合沒有連接運算子(+)、重複運算子(*)、索引運算子([])、片段運算子([start:end])或其他與順序有關的運算
- 建立空集合: set1=set{}
- set2={“Taipei”,”NY”}
- set3=set([1,2,3])
- set4=set(range(5))
- set5=set([i*2 for i in range(5)])

SET(集合): 有比較運算子(>, <, >=, <=, ==, !=)

- S1={'Python','Java','matlab'}
- S2={'Python','Java','matlab','R'}
- S3={'Python','matlab','Java'}
- print(S1==S3) #True
- print(S1 != S2) #True
- print(S1<= S2) # True (S1是S2的子集合)
- print(S1< S2) # S2集合至少有一個元素不存在S1集合

SET操作

- S1={10, 20, 30, 40, 50}
- S1.add(60)
- S1.remove(30)
- S1.pop()
- S2=S1.copy()
- S1.clear()

SET操作(數學集合的操作)

集合運算	集合運算子	對應的集合物件方法
聯集	|	union()
交集	&	intersection()
差集	-	difference()
對稱差集	^	symmetric_difference()

```
s = {1, 2, 'abc', (10, 20)}
t = {1, (10, 20, 30)}

print( s | t)
print( s & t)   # 結果為{1}
print(s - t)   # 結果為{2, 'abc', (10, 20)}
print( t - s)   # 結果為{(10, 20, 30)}
print( s ^ t)   # 結果為{2, 'abc', (10, 20), (10, 20, 30)}
```

分組討論: 串列解析(list comprehension)
分別說明每一題心產生(紅色list)的資料為何?

1. **num_list** = [i for i in range(1, 11) if i % 2 == 0]
2. # scores是儲存考試成績的list
   c_list = [item for item in scores]
   f_list = [item for item in scores if item < 60]
3. **sum** = [i + j for i in range(1, 4) for j in range(10, 14)]
4. **sum1** = [i + j for i in range(1, 4) if i % 2 == 1 for j in range(10, 14) if j % 2 == 0]
5. va1=[[1,2,3],[4,5,2],[3,2,6]]
   va1_list=[max(x) for x in va1]
6. strs = ['chb', 'ydb', 'thd', 'hgh','cbsb','caa']
   new_strs = [name for name in strs if name.endswith('b') and name.startswith('c')]

分組討論: 串列解析(list comprehension) 練習_參考程式

1. num_list = [i for i in range(1, 11) if i % 2 == 0]
2. # scores是儲存考試成績的list
   c = [item for item in scores]
   f = [item for item in scores if item < 60]
3. sum = [i + j for i in range(1, 4) for j in range(10, 14)]

```
sum = []
for i in range(1, 4):
    for j in range(10, 14):
        sum += [i + j]
```

list 的指派

- list的指派，跟一般變數不太一樣，改了源頭，其他會跟著改變
  - a = [12, 35, 62, 44, 90]
  - b = a
  - a [0] = 99
  - print(a)
  - print(b) # 跟著變了

但若不想b 跟著a 被改變，怎麼辦？

解決方案

```
a = [12, 35, 62, 44, 90]
b = a[:]
a [0] = 99
print(a)
print(b) # 不會變了
```

list (advance)

- 串列可以把不同類型的資料放在一起，甚至可以再放入串列
  - a = [[12, 35], 62, 44, 93]
  - b = a[:]
  - a[0][0] = 67
  - print(a)
  - print(b) # 又變了

串列裡面又有串列，包了好多層；但是「[:]」這種語法，只能複製一層

解決方案

- import copy
- a = [[12, 35], 62, 44, 93]
- b = copy.deepcopy(a)
- a[0][0] = 53
- print(a)
- print(b) # 真的OK