Python程式設計：陣列

**目錄**

[Python程式設計：陣列 1](#_Toc206339725)

[1. 定義串列 2](#_Toc206339726)

[# 定義一個數字串列 2](#_Toc206339727)

[2. 讀取串列元素 3](#_Toc206339728)

[3. 串列切片 (list slices) 3](#_Toc206339729)

[4. 串列統計函式 4](#_Toc206339730)

[5. 練習 4](#_Toc206339731)

[5.1 Problem: 尋找最差表現 4](#_Toc206339732)

[6. 練習 4](#_Toc206339733)

[6.1 Problem: 尋找最差表現 4](#_Toc206339734)

[7. 增加串列元素 5](#_Toc206339735)

[7.1 在串列末端增加元素 append() 5](#_Toc206339736)

[7.2 插入串列元素 insert() 5](#_Toc206339737)

[8. 刪除元素 5](#_Toc206339738)

[8.1 刪除串列元素 pop() 5](#_Toc206339739)

[8.2 刪除指定的元素 remove() 5](#_Toc206339740)

[8.3 刪除指定索引的元素 del 6](#_Toc206339741)

[9. 其他串列操作 6](#_Toc206339742)

[9.1 更改串列元素的內容 6](#_Toc206339743)

[9.2 串列的相加 6](#_Toc206339744)

[9.3 串列乘以一個數字 6](#_Toc206339745)

[9.4 串列為空串列的判斷 6](#_Toc206339746)

[10. 練習 7](#_Toc206339747)

[10.1 Problem: 運動會名單整理 7](#_Toc206339748)

[10.2 Problem: 運動會名單整理 7](#_Toc206339749)

[11. 串列進階操作 8](#_Toc206339750)

[11.1 尋找元素索引 index() 8](#_Toc206339751)

[11.2 計算元素數量 count() 8](#_Toc206339752)

[11.3 串列反轉 reverse() 8](#_Toc206339753)

[11.4 串列排序 sort() 8](#_Toc206339754)

[11.5 sorted() 函式 9](#_Toc206339755)

[12. 練習 9](#_Toc206339756)

[12.1 Problem: 尋找重複的學生名單 9](#_Toc206339757)

[12.2 Problem: 籃球隊員排序與排名 9](#_Toc206339758)

[13. 二維陣列 10](#_Toc206339759)

[13.1 範例：成績系統 10](#_Toc206339760)

[13.2 範例：文件系統結構目錄 11](#_Toc206339761)

[14. 練習 11](#_Toc206339762)

[14.1 Problem: 尋找最高分與最低分 11](#_Toc206339763)

[14.2 Problem: 庫存盤點與分類 12](#_Toc206339764)

[15. 用陣列的觀念來看字串 13](#_Toc206339765)

[15.1 islower()/isupper()/isdigit()/isalpha()/isalnum() 13](#_Toc206339766)

[15.2 字串切片 13](#_Toc206339767)

[15.3 字串轉陣列 13](#_Toc206339768)

[15.4 字串分割成陣列 13](#_Toc206339769)

[15.5 陣列組成字串 14](#_Toc206339770)

[15.6 子字串搜尋與索引 14](#_Toc206339771)

[16. 練習 15](#_Toc206339772)

[16.1 Problem: 字串統計與替換 15](#_Toc206339773)

[16.2 Problem: 電子郵件地址驗證與處理 15](#_Toc206339774)

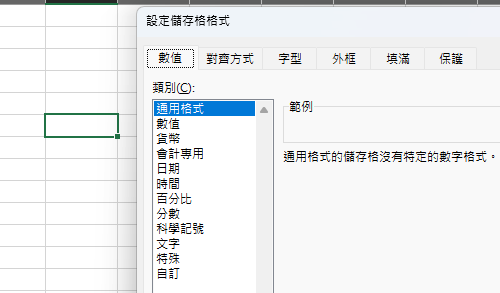
[16.3 Problem: 語音辨識文本處理 16](#_Toc206339775)

[16.4 in 和 not in 運算式 16](#_Toc206339776)

[17. 練習 17](#_Toc206339777)

[17.1 Problem: 尋找重複的學生名單 17](#_Toc206339778)

[18. enumerate() 物件 17](#_Toc206339779)



串列是 Python 中一種最常見的資料型態，可以儲存多個不同型態的元素，例如數字、字串、布林值等。

# 定義串列

## # 定義一個數字串列

james = [23, 19, 22, 31, 18]

# 定義一個字串串列

fruits = ['apple', 'banana', 'orange']

# 混合型態的串列

leits = [3.5, 6.0, 8.5, 'leits']

# 建立一個空串列，之後可以再加入元素

new\_list = []

# 讀取串列元素

你可以透過索引 (index) 來取得串列中的特定元素。索引從 0 開始。

* x[0]：取得第一個元素
* x[1]：取得第二個元素
* x[-1]：取得最後一個元素

james = [23, 19, 22, 31, 18]

print(james[0]) # 取得第一個元素

print(james[2]) # 取得第三個元素

print(james[-1]) # 取得最後一個元素

# 串列切片 (list slices)

切片可以讓你取得串列中的部分元素，而非單一元素。切片語法為 x[start:end:step]。

* start：開始索引 (包含)
* end：結束索引 (不包含)
* step：步長，每次跳過多少個元素

warriors = ['Curry', 'Durant', 'Iguodala', 'Bell', 'Thompson']

# 取得索引 0 到 3 (不包含) 的元素

print(warriors[0:3]) # 輸出: ['Curry', 'Durant', 'Iguodala']

# 取得索引 2 到最後的元素

print(warriors[2:]) # 輸出: ['Iguodala', 'Bell', 'Thompson']

# 取得所有元素

print(warriors[:]) # 輸出: ['Curry', 'Durant', 'Iguodala', 'Bell', 'Thompson']

# 從頭到尾，每隔一個元素取一個

print(warriors[::2]) # 輸出: ['Curry', 'Iguodala', 'Thompson']

# 反轉串列

print(warriors[::-1]) # 輸出: ['Thompson', 'Bell', 'Iguodala', 'Durant', 'Curry']

# 串列統計函式

Python 提供了一些內建函式，可以對串列進行統計。

* max(x)：回傳串列中的最大值
* min(x)：回傳串列中的最小值
* sum(x)：回傳串列中所有元素的總和 (僅限數字串列)
* len(x)：回傳串列中元素的數量

james = [23, 19, 22, 31, 18]

print(f"最大值: {max(james)}") # 輸出: 31

print(f"最小值: {min(james)}") # 輸出: 18

print(f"總和: {sum(james)}") # 輸出: 113

print(f"元素數量: {len(james)}") # 輸出: 5

# 練習

## Problem: 尋找最差表現

|  |  |
| --- | --- |
| Problem Description:  體育老師想分析學生的體能測驗成績。他有一個包含 5 名學生單槓引體向上次數的串列。請你寫一個程式，找出其中表現最差的學生（引體向上次數最少）的成績，並計算所有學生成績的總和。 | |
| Input:  一個包含 5 個整數的串列，代表 5 名學生的引體向上次數。  例如：[12, 8, 15, 6, 10] | Output:  輸出兩行： 第一行：表現最差學生的成績。 第二行：所有學生成績的總和。 |
| Sample Input:  [12, 8, 15, 6, 10] | Sample Output:  6  51 |
| Answer: | |

# 練習

## Problem: 尋找最差表現

|  |  |
| --- | --- |
| Problem Description:  體育老師想分析學生的體能測驗成績。他有一個包含 5 名學生單槓引體向上次數的串列。請你寫一個程式，找出其中表現最差的學生（引體向上次數最少）的成績，並計算所有學生成績的總和。 | |
| Input:  一個包含 5 個整數的串列，代表 5 名學生的引體向上次數。  例如：[12, 8, 15, 6, 10] | Output:  輸出兩行： 第一行：表現最差學生的成績。 第二行：所有學生成績的總和。 |
| Sample Input:  [12, 8, 15, 6, 10] | Sample Output:  6  51 |
| Answer: | |

# 增加串列元素

## 在串列末端增加元素 append()

cars = ['Honda', 'Toyota', 'Ford']

cars.append('Nissan')

print(cars) # 輸出: ['Honda', 'Toyota', 'Ford', 'Nissan']

## 插入串列元素 insert()

cars = ['Honda', 'Toyota', 'Ford']

# 在索引 1 的位置插入 'BMW'

cars.insert(1, 'BMW') # 插入位置、差入元素

print(cars) # 輸出: ['Honda', 'BMW', 'Toyota', 'Ford']

# 刪除元素

## 刪除**串列**元素 pop()

cars = ['Honda', 'Toyota', 'Ford', 'Nissan']

# 刪除最後一個元素

popped\_car = cars.pop()

print(f"被刪除的元素: {popped\_car}") # 輸出: 被刪除的元素: Nissan

print(f"新的串列: {cars}") # 輸出: 新的串列: ['Honda', 'Toyota', 'Ford']

# 刪除索引為 1 的元素

popped\_car = cars.pop(1)

print(f"被刪除的元素: {popped\_car}") # 輸出: 被刪除的元素: Toyota

print(f"新的串列: {cars}") # 輸出: 新的串列: ['Honda', 'Ford']

## 刪除指定的元素 remove()

cars = ['Honda', 'Toyota', 'Ford', 'BMW', 'Ford']

# 刪除第一個 'Ford'

cars.remove('Ford')

print(cars) # 輸出: ['Honda', 'Toyota', 'BMW', 'Ford']

## 刪除指定索引的元素 del

el 語法可以刪除串列中指定索引的元素，或者刪除整個串列。

* del x[index]：刪除指定索引的元素
* del x[start:end]：刪除指定範圍內的元素
* del x：刪除整個串列

cars = ['Honda', 'Toyota', 'Ford', 'Nissan']

# 刪除索引為 1 的元素

del cars[1]

print(cars) # 輸出: ['Honda', 'Ford', 'Nissan']

# 刪除索引 0 到 2 (不包含) 的元素

del cars[0:2]

print(cars) # 輸出: ['Nissan']

# 其他串列操作

## 更改串列元素的內容

cars = ['Honda', 'Toyota', 'Ford']

cars[1] = 'Nissan' # 將索引 1 的元素改成 'Nissan'

print(cars) # 輸出: ['Honda', 'Nissan', 'Ford']

## 串列的相加

cars1 = ['Honda', 'Toyota']

cars2 = ['Ford', 'Nissan']

cars\_all = cars1 + cars2

print(cars\_all) # 輸出: ['Honda', 'Toyota', 'Ford', 'Nissan']

## 串列乘以一個數字

num = [1, 2]

num\_repeated = num \* 3

print(num\_repeated) # 輸出: [1, 2, 1, 2, 1, 2]

## 串列為空串列的判斷

my\_list = []

if not my\_list:

    print("這個串列是空的")

else:

    print("這個串列不為空")

# 練習

## Problem: 運動會名單整理

|  |  |
| --- | --- |
| Problem Description:  某班級有 5 名學生參加運動會，老師用一個串列記錄了他們的報名順序。但比賽前有學生臨時退出，也有新的學生加入。請你根據老師的指示，對串列進行操作。   1. 將原本串列中索引為 2 的學生 'Chris' 刪除。 2. 在串列的結尾處，新增學生 'Frank'。 3. 將原本串列中索引為 1 的學生 'Ben'，更改為 'George'。 | |
| Input:  一個包含 5 個字串的串列，代表 5 名學生的姓名。  例如：['Amy', 'Ben', 'Chris', 'David', 'Eva'] | Output:  輸出一行，顯示最終的串列內容。 |
| Sample Input:  ['Amy', 'Ben', 'Chris', 'David', 'Eva'] | Sample Output:  ['Amy', 'George', 'David', 'Eva', 'Frank'] |
| Answer: | |

## Problem: 運動會名單整理

|  |  |
| --- | --- |
| Problem Description:  教練有兩份籃球隊員名單，一份是先發球員，另一份是替補球員。他想將這兩份名單合併成一份完整的名單，並將合併後的名單複製一份，以便對其進行調整。  任務：   * 將兩份名單合併成一份完整的名單。 * 將合併後的名單複製成三份，以便進行不同陣容的排列組合。 | |
| Input:  第一行：一個包含 3 個字串的串列，代表先發球員。  第二行：一個包含 2 個字串的串列，代表替補球員。  例如：  第一行：['Curry', 'Klay', 'Green']  第二行：['Iguodala', 'Bell'] | Output:  輸出兩行： 第一行：合併後的完整名單。 第二行：將完整名單重複複製三次後的新串列。 |
| Sample Input:  ['Curry', 'Klay', 'Green']  ['Iguodala', 'Bell'] | Sample Output:  ['Curry', 'Klay', 'Green', 'Iguodala', 'Bell']  ['Curry', 'Klay', 'Green', 'Iguodala', 'Bell', 'Curry', 'Klay', 'Green', 'Iguodala', 'Bell', 'Curry', 'Klay', 'Green', 'Iguodala', 'Bell'] |
| Answer: | |

# 串列進階操作

## 尋找元素索引 index()

*x***.***index***(**value**,** start**=**0**,** end**=len(list))**

fruits = ['apple', 'banana', 'cherry', 'date', 'apple']

# 尋找 'cherry' 的索引

index = fruits.index('cherry')

print(f"'cherry' 的索引是: {index}") # 輸出: 'cherry' 的索引是: 2

# 尋找 'apple' 的索引 (會回傳第一次出現的索引)

index = fruits.index('apple')

print(f"'apple' 的索引是: {index}") # 輸出: 'apple' 的索引是: 0

## 計算元素數量 count()

fruits = ['apple', 'banana', 'cherry', 'apple', 'cherry']

# 計算 'apple' 出現的次數

count\_apple = fruits.count('apple')

print(f"'apple' 出現的次數: {count\_apple}") # 輸出: 'apple' 出現的次數: 2

# 計算 'how' 在字串串列中的次數

text = "hello, how are you? How can I help you?"

text\_list = text.split() # 將字串分割成串列

count\_how = text\_list.count('how')

print(f"'how' 出現的次數: {count\_how}") # 輸出: 'how' 出現的次數: 1

## 串列反轉 reverse()

numbers = [1, 2, 3, 4, 5]

numbers.reverse()

print(numbers) # 輸出: [5, 4, 3, 2, 1]

## 串列排序 sort()

numbers = [3, 1, 4, 2, 5]

numbers.sort()

print(f"升序排序: {numbers}") # 輸出: 升序排序: [1, 2, 3, 4, 5]

words = ['apple', 'banana', 'cherry', 'strawberry']

words.sort(reverse=True)

print(f"降序排序: {words}") # 輸出: 降序排序: ['strawberry', 'cherry', 'banana', 'apple']

## sorted() 函式

sorted() 函式會回傳一個新的排序後串列，而不會修改原串列。

numbers = [3, 1, 4, 2, 5]

sorted\_numbers = sorted(numbers)

print(f"原串列: {numbers}") # 輸出: 原串列: [3, 1, 4, 2, 5]

print(f"新排序後串列: {sorted\_numbers}") # 輸出: 新排序後串列: [1, 2, 3, 4, 5]

# 練習

## Problem: 尋找重複的學生名單

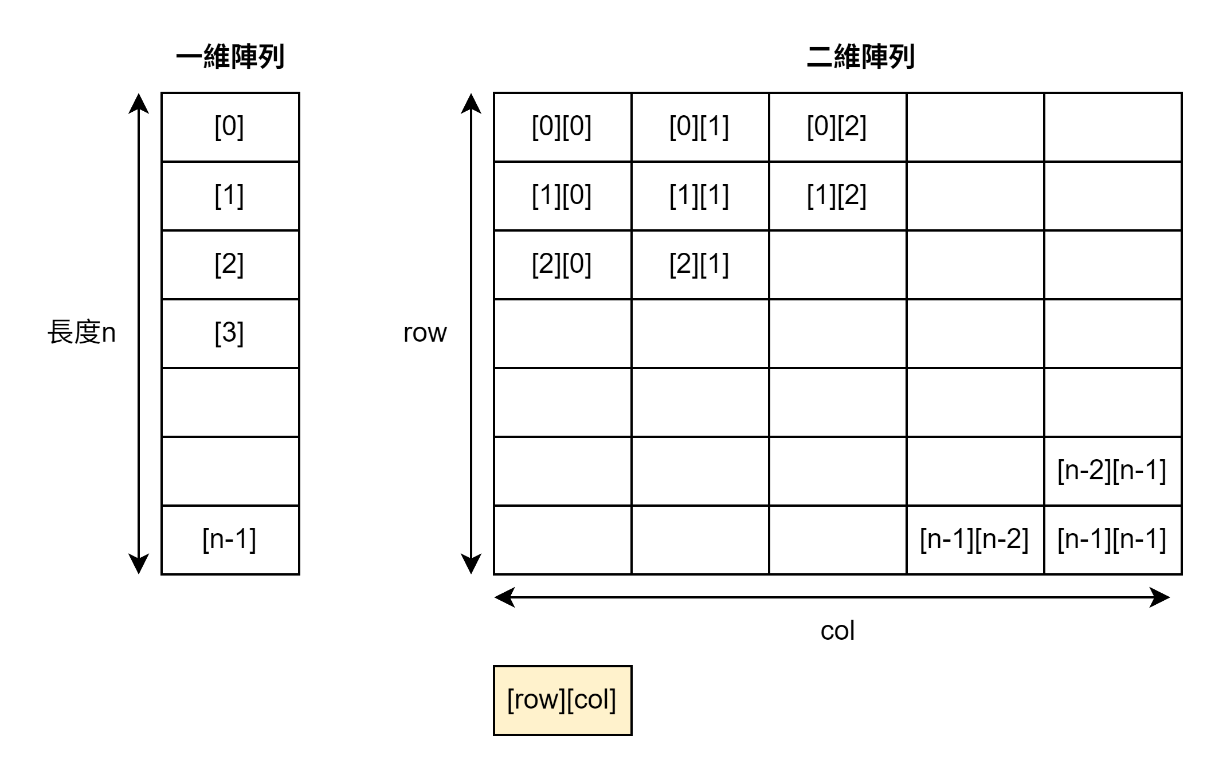
|  |  |
| --- | --- |
| Problem Description:  老師在整理學生名單時，發現有些學生姓名重複出現。請你寫一個程式，找出某個特定學生姓名在名單中出現的次數，並將整個名單按照字母順序重新排序。  任務：   * 計算給定姓名在名單中出現的次數。 * 將學生名單按照字母順序進行升序排序。 | |
| Input:  第一行：一個包含多個字串的串列，代表學生名單。  例如：['Ben', 'Amy', 'David', 'Ben', 'Chris', 'Amy', 'Ben']  第二行：一個要尋找的學生姓名。  例如：'Ben' | Output:  輸出兩行：  第一行：該學生姓名出現的次數。  第二行：排序後的學生名單。 |
| Sample Input:  ['Ben', 'Amy', 'David', 'Ben', 'Chris', 'Amy', 'Ben'] 'Ben' | Sample Output:  3  ['Amy', 'Amy', 'Ben', 'Ben', 'Ben', 'Chris', 'David'] |
| Answer: | |

## Problem: 籃球隊員排序與排名

|  |  |
| --- | --- |
| Problem Description:  教練整理了一份籃球隊員的表現分數（介於 1 到 100 之間），他想將這份分數從高到低進行排序，以便為球員排名。同時，他也想找出表現最差球員（分數最低）在原先未排序名單中的位置（索引值）。  任務：   * 將分數串列進行降序排序，並儲存到一個新的串列中。 * 找出分數最低的球員，他在原始名單中的索引位置。 | |
| Input:  一個包含 5 個整數的串列，代表 5 名隊員的分數。  例如：[85, 92, 78, 95, 88] | Output:  輸出兩行：  第一行：降序排序後的新串列。  第二行：表現最差球員在原始串列中的索引值。 |
| Sample Input:  [85, 92, 78, 95, 88] | Sample Output:  [95, 92, 88, 85, 78]  2 |
| Answer: | |

# 二維陣列

二維陣列可以想像成一個二維空間，就像一個表格或一張棋盤。它是由多個「串列」所組成的「串列」。也就是說，它的每個元素本身又是一個串列。



## 範例：成績系統

用一個陣列sc表示每個學生的[姓名、國文、英文、數學、總分]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 姓名 | 國文 | 英文 | 數學 | 總分(還未計算) |
| 建富1號 | 80 | 95 | 88 | 0 |
| 建富2號 | 98 | 97 | 96 | 0 |
| 建富3號 | 91 | 93 | 95 | 0 |
| 建富4號 | 92 | 94 | 90 | 0 |
| 建富5號 | 92 | 97 | 80 | 0 |

# 建立學生資料陣列：姓名、國文、英文、數學、總分（初始為 0）

sc = [

    ["建富1號", 80, 95, 88, 0],

    ["建富2號", 98, 97, 96, 0],

    ["建富3號", 91, 93, 95, 0],

    ["建富4號", 92, 94, 90, 0],

    ["建富5號", 92, 97, 80, 0]

]

# 這邊我要去將每個學生的總分算出來

# 我們可以先以建富1號做實驗

# print(sc[0][1])

# print(sc[0][2])

# print(sc[0][3])

# print(sc[0][1:4])

# print(sum(sc[0][1:4]))

# sc[0][4] = sum(sc[0][1:4])

# print(sc[0][4])

# 重複，使用迴圈

# sc[0][4] = sum(sc[0][1:4])

# sc[1][4] = sum(sc[1][1:4])

# sc[2][4] = sum(sc[2][1:4])

# sc[3][4] = sum(sc[3][1:4])

# sc[4][4] = sum(sc[4][1:4])

## 範例：文件系統結構目錄

|  |  |  |  |
| --- | --- | --- | --- |
| 文字檔 | 01.txt | 02.txt |  |
| 圖片檔 | image01.png | image02.png | image03.png |
| 影片檔 | viedo01.avi | viedo02.avi |  |

files = [

    ["文字檔", "01.txt", "02.txt"],

    ["圖片檔", "image01.png", "image02.png", "image03.png"],

    ["影片檔", "viedo01.avi", "viedo02.avi"]

]

# 練習

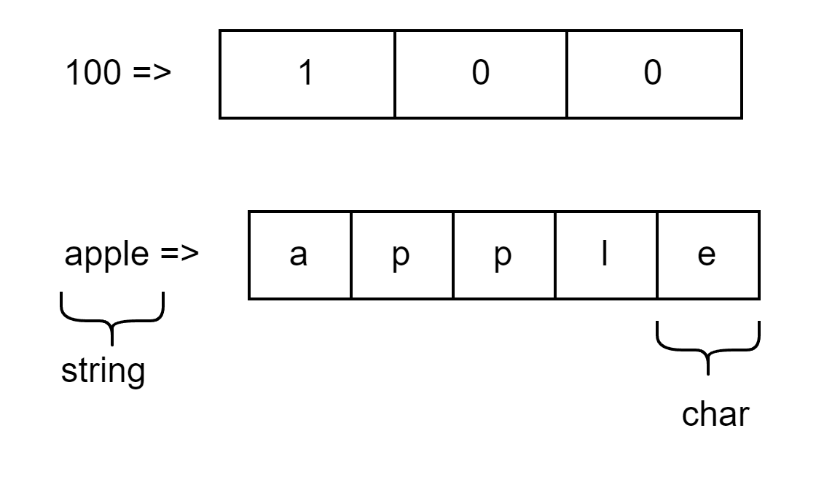
## Problem: 尋找最高分與最低分

|  |  |
| --- | --- |
| Problem Description:  你有一份學生名單和他們的數學成績，存放在一個二維陣列中。請你找出數學成績最高的學生姓名，以及數學成績最低的學生姓名。  任務：   * 找出成績最高的學生姓名。 * 找出成績最低的學生姓名。 | |
| Input:  一個包含多個子串列的二維陣列，每個子串列包含 [姓名, 數學成績]。 | Output:  輸出兩行：  第一行：成績最高的學生姓名。  第二行：成績最低的學生姓名。 |
| Sample Input:  [  ["小明", 85],  ["小華", 92],  ["小芳", 78],  ["小強", 95],  ["小麗", 88]  ] | Sample Output:  小強  小芳 |
| Answer: | |

## Problem: 庫存盤點與分類

|  |  |
| --- | --- |
| Problem Description:  你是一家商店的庫存管理員，你的庫存資料存放在一個二維陣列中，每個子串列代表一種商品類別，包含類別名稱和該類別下的所有商品名稱。現在，老闆請你幫忙盤點。  任務：   * 找出索引為 0 的子串列，計算該類別（飲料）的商品總數（不含類別名稱）。 * 將所有商品名稱合併成一個新的單維串列，並將該新串列反轉排序。 | |
| Input:  一個包含多個子串列的二維陣列，每個子串列的第一個元素代表商品類別，後面是該類別下的所有商品名稱。 | Output:  輸出兩行：  第一行：索引為 0 的子串列中的商品數量。  第二行：反轉排序後的新串列。 |
| Sample Input:  [  ["飲料", "可樂", "汽水"],  ["零食", "洋芋片", "餅乾", "巧克力"],  ["文具", "原子筆"]  ] | Sample Output:  2  ['原子筆', '巧克力', '餅乾', '洋芋片', '汽水', '可樂'] |
| Answer: | |

# 用陣列的觀念來看字串



s = "apple"

print(s[0])

print(s[1])

print(s[-1])

## islower()/isupper()/isdigit()/isalpha()/isalnum()

s = "apple"

print(s.isupper()) # 是否全部大寫

print(s.islower()) # 是否全部小寫

print(s.isdigit()) # 是否全部都是數值

print(s.isalpha()) # 是否全部都是字母

## 字串切片

s = "丹納斯颱風重創南台灣，其中台南嘉義偏鄉很多地方都停電。"

print(s[0:3])

print(s[1:4])

print(s[-3:])

print(s[0:-1])

## 字串轉陣列

s = "丹納斯颱風重創南台灣，其中台南嘉義偏鄉很多地方都停電。"

arr = list(s)

print(s)

print(arr)

## 字串分割成陣列

*string***.***split***()** # 以空格分割字串

string**.***split***(**','**)** # 以,分割字串

s = "丹納斯颱風重創南台灣，其中台南嘉義偏鄉很多地方都停電。"

print(s.split("南"))

test = "1 2 3 4 5"

print(test.split())

# ['1', '2', '3', '4', '5'] 這邊你可以再找一下，要怎麼把裡面的每一個元素，轉成int

## 陣列組成字串

*連接字串***.***join***(**陣列**)**

s = r"D:\github\kcwc1029.github.io\Python\_Tutor\Python程式設計：判斷式"

spilt\_s = s.split("\\")

print(spilt\_s)

join\_s = "\\".join(spilt\_s)

print(join\_s)

## 子字串搜尋與索引

* find()：回傳子字串第一次出現的位置，找不到時回傳-1
* index()：回傳子字串第一次出現的位置，找不到時回傳ValueError

text = "hello world"

# 使用 find()

print(text.find("o"))      # 4

print(text.find("world"))  # 6

print(text.find("x"))      # -1（找不到）

# 使用 index()

print(text.index("o"))     # 4

print(text.index("world")) # 6

# print(text.index("x"))   # ValueError: substring not found

# 可選參數

print(text.find("o", 5))       # 7（從 index 5 開始找）

print(text.index("o", 5, 9))   # 7（只在 index 5 到 8 之間找）

# 練習

## Problem: 字串統計與替換

|  |  |
| --- | --- |
| Problem Description:  你有一段包含多個單字的字串，現在需要找出某個特定單字在字串中出現的次數，並將它替換成另一個單字。任務：   * 計算給定的單字在字串中出現的次數。 * 將所有出現的該單字替換成新的單字。 | |
| Input:  第一行：一個包含多個單字的字串，以空格分隔。  例如：the dog is on the table  第二行：一個要尋找與替換的單字。  例如：the  第三行：一個要替換成的新單字。  例如：a | Output:  輸出兩行：  第一行：該單字出現的次數。  第二行：替換後的新字串。 |
| Sample Input:  the dog is on the table  the  a | Sample Output:  2  a dog is on a table |
| Answer: | |

## Problem: 電子郵件地址驗證與處理

|  |  |
| --- | --- |
| Problem Description:  你拿到一個電子郵件地址字串，需要對其進行處理。請你檢查這個字串是否包含 @ 符號，如果包含，請將其拆分成使用者名稱和網域兩部分，並將這兩部分用 | 符號重新連接起來。如果沒有 @ 符號，則回報「無效地址」。  任務：   * 檢查字串中是否包含 @ 符號。 * 如果包含，將字串以 @ 分割成兩部分。 * 將分割後的兩部分用 | 重新連接成一個新的字串。 * 如果不包含，則輸出「無效地址」。 | |
| Input:  一個電子郵件地址字串。  例如：test\_user@example.com | Output:  輸出一行：  如果地址有效，輸出重新連接後的字串。  如果地址無效，輸出「無效地址」。 |
| Sample Input:  test\_user@example.com | Sample Output:  test\_user|example.com |
| Sample Input:  invalid\_email | Sample Output:  無效地址 |
| Answer: | |

## Problem: 語音辨識文本處理

|  |  |
| --- | --- |
| Problem Description:  你拿到一段語音辨識後轉成的文字，但這段文字中包含一些雜訊，例如數字和多餘的空格。請你處理這段文字，將所有數字移除，並將所有單字以句號 . 隔開，最後將處理後的文字全部轉為小寫。  任務：   * 建立一個空串列。 * 遍歷輸入字串中的每個字元，如果該字元是字母，則將其加入串列中。 * 將串列中的所有字元用 . 連接起來。 * 將最終的字串全部轉換為小寫。 | |
| Input:  一個包含字母、數字和空格的字串。  例如：Hello 123 World 456 Python | Output:  輸出一行：處理後的最終字串。 |
| Sample Input:  Hello 123 World 456 Python | Sample Output:  h.e.l.l.o.w.o.r.l.d.p.y.t.h.o.n |
| Answer: | |

## in 和 not in 運算式

in 和 not in 運算式主要用於判斷一個元素（物件）是否屬於另一個複合型物件，例如字串（string）、串列（list）、元組（tuple）或字典（dict）。

* boolean = obj in A：如果 obj 在物件 A 中，會回傳 True。
* boolean = obj not in A：如果 obj 不在物件 A 中，會回傳 True

fruits = ["apple", "banana", "cherry", "date"]

# 檢查 'cherry' 是否在串列中

if "cherry" in fruits:

    index = fruits.index("cherry")

    print(f"'cherry' 的索引是: {index}") # 輸出: 'cherry' 的索引是: 2

# 檢查 'grape' 是否不在串列中

if "grape" not in fruits:

    print("串列中沒有 'grape'") # 輸出: 串列中沒有 'grape'

# 練習

## Problem: 尋找重複的學生名單

|  |  |
| --- | --- |
| Problem Description:  老師有一份學生名單。為了避免重複，他想知道某位學生是否已經在名單中。請你寫一個程式，檢查某位學生是否在名單中，並根據結果輸出相應的訊息。：   * 判斷第二行輸入的學生姓名是否在第一行的學生名單中。 * 如果在，輸出該學生姓名。 * 如果不在，輸出 "This student is not on the list."。 | |
| Input:  第一行：一個包含多個字串的串列，代表學生名單。  例如：['Ben', 'Amy', 'David', 'Chris']  第二行：一個要尋找的學生姓名。  例如：'Ben' | Output:  輸出一行，顯示判斷結果。 |
| Sample Input:  ['Ben', 'Amy', 'David', 'Chris']  'Ben' | Sample Output:  Ben |
| Sample Input:  ['Ben', 'Amy', 'David', 'Chris']  'Frank' | Sample Output:  This student is not on the list. |
| Answer: | |

# enumerate() 物件

enumerate() 函式可以用來處理可迭代（iterable）類型物件的元素，它會將每個元素與其對應的索引值配對，然後以 enumerate 物件的形式回傳。這個功能在處理資料時非常有用，特別是當你需要同時使用元素的索引和值時。

在 Python 中，可迭代物件是指那些可以被一個一個遍歷的物件，例如：串列 (list)、元組 (tuple)、集合 (set)、字典 (dict)、字串 (string)

*obj* **=** **enumerate(**iterable**,** start**=**0**)**

# iterable：你要處理的可迭代物件，例如一個串列或元組。

# start：可選參數，用來指定索引的起始值。如果沒有設定，預設會從 0 開始。

# 1. 定義一個串列

drinks = ["coffee", "tea", "wine"]

# 2. 將串列轉換為 enumerate 物件，初始索引為 0

enumerate\_drinks = enumerate(drinks)

# 3. 將 enumerate 物件轉換成串列並輸出

print(f"轉換成串列輸出，初始索引值是 0 = ", list(enumerate\_drinks))

# 輸出: 轉換成串列輸出，初始索引值是 0 =  [(0, 'coffee'), (1, 'tea'), (2, 'wine')]

# 4. 再次將串列轉換為 enumerate 物件，初始索引為 10

enumerate\_drinks = enumerate(drinks, start=10)

# 5. 將 enumerate 物件轉換成串列並輸出

print(f"轉換成串列輸出，初始索引值是 10 = ", list(enumerate\_drinks))

# 輸出: 轉換成串列輸出，初始索引值是 10 =  [(10, 'coffee'), (11, 'tea'), (12, 'wine')]