



# Python Lecture 01

鄭博宇 Benjamin Cheng

國立臺灣大學資訊管理學系

EMILY in IM



# Agenda

- 第一行程式：“Hello world!”
- 變數是什麼？
- If - else 判斷
- 串列 List
- 迴圈 Loop
- 字串 String



# 課程進行

- .ipynb 請先準備好
- 開啟方式：Google Colab / Jupyter Notebook
- 課程講義、隨堂練習、歡迎先往下做題目



# 講師介紹

Benjamin 鄭博宇

- 資管雙經濟 大二
- IG: @benjami8\_24n



# 第一行程式

EMILY in IM



# 今天的第一行程式

請在介面上打上 `print("Hello world!")`，並且按下 執行按鈕。

你看到了什麼？



# 今天的第一行程式

請在介面上打上 `print("Hello world!")`，並且按下 執行按鈕。

當學習一個程式語言時，第一件事情是練習 輸出方法。在 Python 中，我們使用 `print()` 函數來列印不同的 變數。

引號 ("") 代表內部包含的 變數 是 字串 (string)，我們將在稍後介紹到！



# 小試身手

艾蜜莉在必修的程式設計課覺得很無聊，於是決定來搞個小花樣：她希望每次執行程式碼時，能夠印出 Emily is the smartest student in IM !

請修改剛剛的程式碼，幫助艾蜜莉完成這個目標吧！



# 變數是什麼？

EMILY in IM



# 變數 Variable

我們可以將 變數 理解為 裝內容的容器。

不同的容器能裝不同類型的資料，這就是資料型態。



# 變數 Variable

一些在 Python 中常見到的資料型態 (type) 包括：

1. 整數 (int)：負整數、0、正整數
2. 浮點數 (float)：帶有小數點的數值
3. 字串 (string)：使用“引號”所包住的字元組合，用來表達單字或句子
4. 布林值 (bool)：True / False，我們可以使用 1 表達 True、0 表達 False



# 變數 Variable

我們可以將 變數 理解為 裝內容的容器。

在 Python 中，我們在處理變數時，我們是透過 物件 (object) 的 名稱 來訪問儲存在電腦中的內容。因此，一個變數的 內容值 可以不斷被讀取、修改，但是變數的 名稱 不能改變。



# 變數 Variable - 創立變數

我們可以將 變數 理解為 裝內容的容器。

當我們要新增一個變數時，我們使用 賦值運算子（=）來初始化一個變數。

```
new_variable = 15    # 整數 int 型態  
print(new_variable)  
print(type(new_variable))
```



# 變數 Variable - 創立變數

我們可以將 變數 理解為 裝內容的容器。

之後，我們也能使用 賦值運算子（=）來修改變數的內容。

```
new_variable = 20 # 修改內容  
print(new_variable)
```



## 想法

Python 會根據我們所賦予的值，自動將 變數 調整至對應的 資料型態 喔！

## 試試看

`type()` 函數會告訴我們變數的資料型態。

請自行摸索看看，當我們給定不同類型的變數時，會印出什麼結果呢？



# 變數 Variable - 創立變數

我們可以將 變數 理解為 裝內容的容器 。

有一些 命名規則 是需要遵守的！

1. 只能包含 字母、數字、底線 ，且不能以數字開頭
2. 不能使用 Python 的關鍵字 (例如 if、while、True 等)
3. 習慣上，變數名稱要盡量讓我們清楚了解 此變數的用途



## 注意

變數名稱的大小寫是有差的，因此要特別注意！

## 試試看

如果我們將一個變數的名稱定為 `2024IMcamp`，執行後會發生什麼事？



# 變數 Variable - 比較運算子

```
a = 10
b = 3
print(a == b)    # False
print(a != b)    # True
print(a > b)     # True
print(a < b)     # False
print(a >= b)    # True
print(a <= b)    # False
```



# 變數 Variable - 算術運算子

```
a = 3 + 5      # 8
b = 3 - 5      # -2
c = 3 * 5      # 15
d = 5 / 3       # 1.6667
e = 5 // 3      # 1
f = 5 % 3       # 2
g = 5 ** 3      # 125
```



# 變數 Variable - 邏輯運算子

```
x = 5 > 3      # True  
y = 3 > 5      # False  
print(x and y) # False  
print(x or y)  # True  
print(not x)   # False
```



# 變數 Variable - 更多的...賦值運算子

我們能將 算術運算 以及 賦值 兩個動作合併。

```
a = 10
a += 5      # a = a + 5
print(a)    # 15
a *= 2      # a = a * 2
print(a)    # 30
```



## 小試身手

艾蜜莉在放學後，跑到對面的公館商圈購物，總共發現了三樣想買的商品：

保暖耳罩 售價 300 元，滑鼠墊 售價 500 元，髮箍 售價 100 元。艾蜜莉決定三樣商品都各買兩件。

買完後，店家開心地提供了 8 折的優惠。請使用前面學過的運算子，幫艾蜜莉算算看，這樣總共會花多少錢？



# 小試身手

艾蜜莉今天遇到了一個謎題，試著協助她找到答案吧！謎題如下：

試著幫下列程式中變數 a、b 和 c 都分別填入 True 或 False，使最終輸出的值為 True 吧！

```
a = ?    # True or False?  
b = ?    # True or False?  
c = ?    # True or False?  
#試著讓程式碼輸出 True 的結果吧！  
print(((a and b) or (b or c)) and ((not a) and (b == False) ))
```



# Answer

```
a = False  
b = False  
c = True
```

#試著讓程式碼輸出 True 的結果吧！

```
print(((a and b) or (b or c)) and ((not a) and (b == False)))
```



# If - else 判斷

EMILY in IM



# if? else?

`if` 條件式:

# 如果條件式為 True 時，執行這一段

`elif` 條件式:

# 如果不符合更前面的條件，且條件式為 True 時，執行這一段

(中間可以放很多個 `elif`)

`else:`

# 假設前面都不符合，執行這一段



# 小試身手

艾蜜莉今天在 Python 課堂中，遇到一個跟生活情境有關的題目：

請幫助她寫一個程式，來判斷一個人的年齡是否符合考駕照的規定  
(上網查查看，台灣幾歲才能考駕照？)，同時，超過 70 歲者，應  
印出 “高齡駕駛” 。



## 補充

在這個任務中，我們需要使用者 輸入自訂的內容，因此會用到  
`input()` 函數。

這個函數會記錄使用者輸入的內容，預設以 字串（string） 儲存，  
如果是數字內容要記得轉換成 整數（int） 型態喔！

```
# 範例
a = input("請在此輸入內容：")
print(a)
```



因此，關於考駕照的年齡判斷問題，應該會如此開頭：

```
age = int(input("請輸入您的年齡："))    # 將輸入的內容從字串轉換為整數  
  
# 後面交給你們囉 :D
```



# 串列 List

EMILY in IM



# 串列 List - 動機

當我們今天想存 3 個類似的數值時...

```
a = 5  
b = 7  
c = 6
```



# 串列 List - 動機

如果今天變成 50 個？

如果當中有兩個變數要 交換順序？

如果我需要從中間 拿掉一個，其他往前遞補？

如果臨時有人從 中間插入？



# 串列 List - 動機

我們需要一個 系統化的容器：

1. 將一串元素們按照順序擺在一起
2. 元素可以各式各樣：整數、字母、字串、布林值
3. 有一些設計好的功能，方便我們使用（例如：變換順序、拿掉其中一個、從中間插入...）



# 串列 List - 宣告

在 Python 中，我們使用 [中括號] 表達串列

創建一個串列可以透過...

```
1. new_list = [0, 1, 2, 3, 4]
```

```
2. another_list = list()
```

兩種方法。



## 試試看

艾蜜莉想要幫自己的三次段考成績，用一個 list 來記錄，成績分別是：95、47、68。

請創建一個名為 score 的串列，將成績儲存起來。



# 串列 List - 取值

在 Python 中，我們使用 [中括號] 表達串列

我們想要取得串列中索引值 = i 的內容...

```
print(new_list)
print(new_list[2]) #注意！誰被印出了？
```



## 想法

在程式中，我們從 0 開始計數，  
所以索引值從 0 開始算。

## 試試看

請幫艾蜜莉印出第二次段考的成績！  
(提示：第二次段考的索引值是多少？)



# 串列 List - 索引值專論

在程式中，我們從 0 開始計數，所以索引值從 0 開始算。

- 一般情況下， $[i]$  表達 第  $i + 1$  個 元素
- 從後方開始取值，使用  $[-i]$  表達 倒數第  $i$  個 元素
- $[i:j]$  表示 從  $i$  (包含) 到  $j$  (不包含) 的元素



試試看

如果是 [i: (留白) ] 以及 [ (留白) :j] 時，範圍是在哪裡？

試試看

當今天變成 [i:j:k] 時，k 有什麼意義？



# 串列 List - 插入資料

從尾端 插入資料：

1. `append( element )` — 將另一個 元素 加在後面
2. `extend( list )` — 將另一個 串列 加在後面



```
# 使用 append
fruits = ['apple', 'banana', 'cherry']
fruits.append('orange')
print(fruits) # 輸出: ['apple', 'banana', 'cherry', 'orange']

# 使用 extend
fruits = ['apple', 'banana', 'cherry']
more_fruits = ['mango', 'pineapple', 'grape']
fruits.extend(more_fruits)
print(fruits) # 輸出: ['apple', 'banana', 'cherry', 'mango', 'pineapple', 'grape']
```



## 試試看

在 `my_list` 的後方加上新的元素 `"New_element"`。

## 試試看

建立一個新的串列 `another_list`，內容物自訂，並且將這個新串列接在原本的 `my_list` 後面。



## 注意

如果使用 `append()` 插入一個 串列 會發生什麼事？

我們將會擁有一個新元素，該元素是一個串列。

```
[1, 2, 3, [4, 5, 6], 87, 0]
```



# 串列 List - 插入資料

從 中間 插入資料：

```
insert( index, element )
```

給定 索引值 `index` 以及 想插入的元素 `element` 作為參數，就可以  
從中間插入新資料！

補充

其餘的資料將會往後順延一格



## 試試看

在每兩次段考之間，還會有一次隨堂小考。

艾蜜莉兩次小考成績分別是：99、107。

請依照時間順序，將小考成績插入 score 串列中，並且將最後的  
score 結果列印出來。



# 串列 List - 移除資料

給定 索引值 移除資料：

```
pop( index )
```

給定 索引值 `index` 作為參數，該位置的資料會被刪除

補充

其餘的資料將會往前順延一格

若未給定參數，預設刪除最後一格的元素



# 試試看

`pop()` 其實會回傳資料！

試試看將 `pop( index )` 放在等號後面，有什麼東西被回傳了？



# 串列 List - 移除資料

給定 想移除的內容：

```
remove( element )
```

給定 元素 element 作為參數，第一個出現的該元素會被刪除

補充

其餘的資料將會往前順延一格



# 串列 List - 複製資料

我們有兩種複製資料的方法：

## 1. Shallow Copy

我們做 `B = A`，看似將串列 A 複製了一份給 B，但是實際上，  
真的如此嗎？



## 試試看

將 `my_list` 按照上頁的方法複製一份給 `copy_list`。

若我們將 `my_list[0]` 的內容移除，試試看將 `copy_list` 列印出來，觀察看看發生什麼問題了？



# 串列 List - 複製資料

我們有兩種複製資料的方法：

## 1. Shallow Copy

我們做 `B = A`，看似將串列 A 複製了一份給 B，但是實際上，  
並不是如此。

A 與 B 此時共用著同一個串列，我們將兩者 引用 到同一個地方。  
因此，改變其中一個串列，會影響到另外一個串列。



# 串列 List - 複製資料

因此，我們介紹第二種複製資料的方法：

## 2. Deep Copy

`B = A[ : ]`，將串列 A 中的每一個元素，都依序複製到一個新的串列 B 中。



# 串列 List - 還有一些功能值得你探索...

## 1. 排列 `list.sort(reverse=True/False)`

我們能夠使用 **sort()** 來排序，其中 **參數** 預設為 `False`，會將串列重新由小到大排序；若我們將參數設定為 `True`，則會由大排到小。



# 串列 List - 還有一些功能值得你探索...

## 2. 長度 `len( list )`

我們能夠藉此方便取得 串列的長度 ，對於後續迴圈的操作十分  
重要！



# 串列 List - 還有一些功能值得你探索...

## 3. 次數 `list.count( element )`

在參數中放入 元素**element**，將會回傳該元素在串列中的 出現  
次數



## 小試身手

艾蜜莉是一名剛進大學的新生，她希望能夠用 Python 幫助她管理生活和學習計畫。請幫助她解決以下問題...

### 題目 1: 建立課程表

艾蜜莉需要建立她的每週課程表，包括她加簽的所有課程名稱。

請建立一個名為 `courses` 的串列，包含以下課程：Calculus, English, Python Programming, Economics, Accounting, Management，然後將她的課程表列印出來。



## 小試身手

艾蜜莉是一名剛進大學的新生，她希望能夠用 Python 幫助她管理生活和學習計畫。請幫助她解決以下問題...

### 題目 2: 插入新課程

艾蜜莉的朋友推薦她修一門甜涼通識，課名叫做「普通心理學」，她想將它插入課程表的「第二個位置」。

請使用 `insert` 函數，將 "Psychology" 插入到 `courses` 的第二個位置，並且列印修改後的課程表。



## 小試身手

艾蜜莉是一名剛進大學的新生，她希望能夠用 Python 幫助她管理生活和學習計畫。請幫助她解決以下問題...

### 題目 3: 移除課程

由於課程時間衝堂了，艾蜜莉決定退掉「會計學」，請幫助她將其從課程表中移除，並且列印更新後的課程表。

### 題目 4: 課程數量統計

艾蜜莉想知道她現在的課程總數是多少，請列印出目前 `courses` 中的元素數量。



## 小試身手

艾蜜莉是一名剛進大學的新生，她希望能夠用 Python 幫助她管理生活和學習計畫。請幫助她解決以下問題...

### 題目 5: 排序課程

艾蜜莉希望改變課程表的順序，改成按照字母順序排列，方便查找。請使用 `sort` 函數，將課程表排序，並列印結果。



## 小試身手

### 綜合應用

艾蜜莉的課表最終有以下課程：Calculus, English, Python Programming, Economics, Psychology, Management。她希望進行以下操作：

1. 將課表改成依照字母順序的相反排列。
2. 計算課程表中包含 "English" 的次數。
3. 最後，列印出課程表以及出現 "English" 的次數。



# 迴圈 Loop

EMILY in IM



# 迴圈 - 動機

有時，我們希望機器幫我們處理 重複的事情，於是 While 迴圈 誕生了。

我們使用一個 條件判斷式 (condition) ，只要符合就會不斷做迴圈內的事情。



# 迴圈 - 動機

以下其中一件事情發生，我們跳出迴圈，繼續往後續程式執行。

1. 條件判斷式 不成立了
2. 遇到 `break`



2

重複執行 1，直到 condition 為 False 後離開迴圈

**while** condition:



1

若 condition 為 True  
執行 statements 區塊

試試看

如果想要使用 `break`，應該要怎麼做呢？



# 迴圈 - 計數器

例如，我們希望做一個計數器，確保迴圈能跑 10 次，並且依序印出 0 ~ 9。

```
i = 0  
  
while i < 10:  
    print( i )  
    i += 1  
  
print( "finish!" )
```

## 試試看

如果想要改成列印從 10 ~ 1，應該怎麼做？



# 迴圈 - 與串列一起使用！

例如，我們想要取出串列中第 5 個到第 9 個內容，並且依序印出。

```
new_list = [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
i = 4
while i < 9:
    print( new_list[ i ] )
    i += 1
print( "finish!" )
```

試試看

如果我們想要從串列的第 5 個開始印到最尾巴，應該怎麼做？

(提示: 可以善用 `len()` 來取得串列的長度！)



# 迴圈 - 更直覺的方法：For 迴圈

有點不方便？

於是我們發明了 For 迴圈，又稱為 計數迴圈。

搭配 `range()`，我們能更方便控制函數執行特定的圈數，並且省去了處理 `i` 的困擾！

```
for i in range(5):
    print(i)

# 將會依序印出 0, 1, 2, 3, 4
```

## 試試看

使用 For 迴圈，將串列的第一個到第五個元素印出。



# 迴圈 - Range專論

我們將以下程式展開。

```
for i in range(5):  
    print(i)
```

# 將會依序印出 0, 1, 2, 3, 4

比較以下的程式，執行效果是一樣的...

```
for i in [0, 1, 2, 3, 4] :  
    print(i)
```

# 將會依序印出 0, 1, 2, 3, 4



## 想法

1. `range()` 非常像是一個串列，裡面是由許多 由小到大的數字元素 所組成。
2. For 迴圈 會將後方串列中的元素逐一執行，並且分別將每個元素 賦值給 `i` 。



# 迴圈 - Range專論

Range() 還能做更多事情！

1. range(5) 可以理解為... 從 0 ~ 4 的串列
2. range(3, 10) 可以理解為... 從 3 ~ 9 的串列 (含首不含尾)
3. range(2, 10, 2) 可以理解為... 2, 4, 6, 8 (公差為 2)



## 想法

與 串列 的表達方法很相近！

## 試試看

使用 For 迴圈 與 `range()` ，將 `my_list` 中的第 3 個到第 6 個元素印出來。



# 迴圈 - Continue

執行以下兩種程式，你發現什麼不同？

```
for i in range(10):
    if i == 5:
        continue
    else:
        print(i)
```

```
for i in range(10):
    if i == 5:
        break
    else:
        print(i)
```

想法

continue 會跳回 迴圈的開頭，而 break 會 跳出迴圈



# 迴圈 - 巢狀迴圈

當我們想要做的工作需要不只一層迴圈時，我們能夠 將迴圈包含在更大的迴圈 中，我們稱之為巢狀迴圈。

以下為一個雙重迴圈的範例：

```
for i in range(10):
    for j in range(10):
        print("i=", i, "j=", j )
```



## 想法

在 Python 中，縮排是重點！

縮排，是讓電腦知道迴圈範圍的好方法。

(點擊鍵盤上的 tab 鍵，可以自動縮排喔！)

## 試試看

請修改前一頁的程式碼，實做一個 九九乘法表。



# 常見資料型態

## 字串 String

EMILY in IM



## 什麼是字串？

在電腦程式中，字元（character）是一種數據類型，通常用來表示單個文字（如字母、數字、符號）。

我們要表達 單字 或 句子 時，就需要使用 字串（string） 來將多個字元組合起來。

例如："Word" 或是 "Welcome to IM CAMP !" 都是字串。



## 想法

在 Python 中，我們使用引號 (""" ) 來表達字串。

單引號 (') 與 雙引號 (") 在 Python 中都可以表達字串，只要成對使用即可！

## 試試看

宣告一個新變數 `my_string`，並且將 "My name is (你的名字) " 字串儲存在裡面。



## 什麼是字串？

我們能夠使用 中括號 [] ，來將字串中的某個字元取出來。

例如：

```
new_string = "IM Camp"  
  
print(new_string)  
print(new_string[0])  
print(new_string[4])
```



## 想法

此時，字串就像是一個很多字元的串列。

記得 索引值 是從 0 開始喔！

## 試試看

我們能否比照 串列 ，使用 [:] 來截取範圍呢？

試著將 `my_string` 中，(你的名字) 列印出來。



# 字串 String - 換行字元

當我們使用 `print` 函數後，預設會換行。

如果我們想要手動換行，可以使用 换行字元 `\n`。

```
print("IMcamp 2025")
print("IMcamp\n2025")
```



## 試試看

將 "Welcome to IM CAMP !" 使用一個 `print` 函數印出，且每個字都要獨立一行。



# 字串 String - 串聯

我們能夠使用 + (加號) , \* (乘號) , 將字串進行串聯。

1. + (加號) — 將左右的兩個字串連接起來
2. \* (乘號) — 將原本的字串重複數遍

```
print("IM" + "Camp" + "2025")
print("IM Camp" + "萬歲" * 3)
```



# 字串 String - 比較

我們能夠使用 `==` (等於) , `!=` (不等於) , 檢查字串是否相同。

```
string_1 = "IM"  
string_2 = "Camp"  
  
print(string_1 == string_2)    # False  
print(string_1 != string_2)    # True
```



# 字串 String - 比較

另外，我們能夠使用 `>`（大於）`、<`（小於），比較字串的 字典序。

```
print(string_1 > string_2)      # True
```

## 想法

字典序：是從第一個字元開始比較，若相同則比下一個字，直到分出大小或一方結束為止。



# 字串 String - 切換大小寫

我們能夠使用 `upper()` 、`lower()`，將字串的所有字元切換大小寫。

```
string_1 = "IM"  
string_2 = "Camp"  
  
print(string_1.lower())  
print(string_2.upper())
```



# 字串 String - 檢查字串性質

我們能夠使用 `isupper()` 、 `islower()` ，檢查字串是否皆為大 / 小寫。

```
string_1 = "IM"  
string_2 = "Camp"  
  
print(string_1.islower())    # False  
print(string_2.isupper())    # False
```



# 字串 String - 檢查字串性質

我們能夠使用 `isalpha()`，檢查字串是否皆為 英文字母 。

我們能夠使用 `isnumeric()`，檢查字串是否皆為 數字 。

我們能夠使用 `isalnum()`，檢查字串是否皆為 英文字母或數字 。

```
string_3 = "IMcamp2024"

print(string_3.isalpha())      # False
print(string_3.isnumeric())    # False
print(string_3.isalnum())     # True
```



## 補充

在這三個判斷方法中，如果提供 空字串 皆會回傳 False。

## 試試看

將字串設定為中文字元 “一二三” ， `isnumeric()` 會判斷他是數字嗎？

如果改成 “IMcamp 2024” ，多出的空格是否會改變 `isalnum()` 的判斷？



# 字串 String - 檢查子字串

我們能夠使用 `in`、`not in`，檢查字串擁有某個 子字串 。

```
string_1 = "IM"  
string_2 = "Camp"  
string_3 = "IMcamp2024"  
  
print(string_1 in string_3)      # True  
print(string_1 not in string_3)  # False  
print(string_2 in string_3)      # False
```



## 想法

在一個字串中任意取一段字元，稱為該字串的子字串。



# 字串 String - 分割字串

我們能夠使用 `split()`，將字串以 空格 分割，產生出一個串列。

```
string_4 = "IM Camp is so great!"  
print(string_4.split())
```

另外，我們也能給定 分割符 ，依據不同東西分割串列。

```
string_5 = "myemail@gmail.com"  
print(string_5.split("@"))
```



# 字串 String - 去除首尾

我們能夠使用 `strip()`，將字串以 空格 去頭去尾，產生出一個新字串。

```
string_4 = "    IM Camp is so great!    "
print(string_4.strip())
```

另外，我們也能給定 指定字符 ，依據不同東西切割頭尾。

```
string_5 = "\n\n\nmyemail@gmail.com\n\n"
print(string_5.strip("\n"))
```



## 想法

`strip()` 所回傳的是一個新字串，而 `split()` 回傳的是一個串列  
喔！

## 試試看

觀察一下，`strip()` 函數會將 不在頭尾 的 指定字符 移除嗎？



# 字串 String - 還有更多功能！

我們能夠使用 `find()`，給定 子字串 作為 參數，回傳「第一個子字串 出現時，第一個字元的索引值」。

```
string_4 = "IM Camp is so great!"  
print(string_4.find("great"))
```



# 字串 String - 還有更多功能！

另外，我們也能使用 `replace()`，將原本的子字串 全部換成 紿定的新內容。

```
string_4 = string_4.replace("great", "excellent")
print(string_4)
```

## 注意

如果找不到該子字串，會回傳 -1 !



# 小試身手

艾蜜莉是一名大學生，除了忙碌的學業，她還加入了一個服務性社團，負責幫助其他學生解決程式問題。她希望能用 Python 幫助她管理一下學業和社團的日常任務。以下是幾個問題，請幫助她完成目標。



# 小試身手

## 題目 1: 任務清單建立

艾蜜莉記錄了一些待完成的任務，包括學校作業與社團活動。請使用 while 迴圈讓艾蜜莉逐一輸入她的待辦事項，直到輸入 "done" 字串為止。

請製作一個程式，能夠將這些待辦事項存入一個名為 tasks 的串列中，最後列印出待辦事項清單。



目前，艾蜜莉目前的待辦事項如下：*calculus\_hw, check\_meeting, python\_tutor, economics\_hw, accounting\_hw, python\_hw, algorithm\_hw, practice\_taekwondo*



# 小試身手

## 題目 2: 任務分類

艾蜜莉希望將她的任務分成兩大類：學校作業與社團活動。請使用 `for` 迴圈逐一檢查 `tasks` 串列中的每個任務。

如果任務中包含關鍵字 "hw"，則將其加入 `school_tasks` 串列；否則，請加入 `club_tasks` 串列。

最後，分別列印 `school_tasks` 和 `club_tasks`。

提示：使用字串函數 `in` 判斷是否包含關鍵字。



# 小試身手

## 題目 3: 進行任務

艾蜜莉決定依次完成所有學校作業，並在完成後將其從清單中移除。請使用 while 迴圈，逐步製作以下事項：

- 列印[正在完成的任務],[剩餘學校作業數量]。
- 使用 `pop(0)` 移除該項（已完成的）任務。

最後，當任務全部完成後，列印「學校作業已完成！」。

提示：可以使用 `len()` 確認串列是否為空。



# 小試身手

## 題目 4: 計算社團活動所需時間

艾蜜莉需要估算她完成社團活動所需的時間，每項活動需要的時間由使用者指定。

請使用 for 迴圈遍歷 club\_tasks，針對每項活動，請使用者輸入所需的時間（分鐘）。在累加所有活動的時間後，列印出總時間。

提示：良好的使用者介面很重要。請記得要先印出每項社團任務的名稱，再請艾蜜莉輸入該項任務所需的時間！