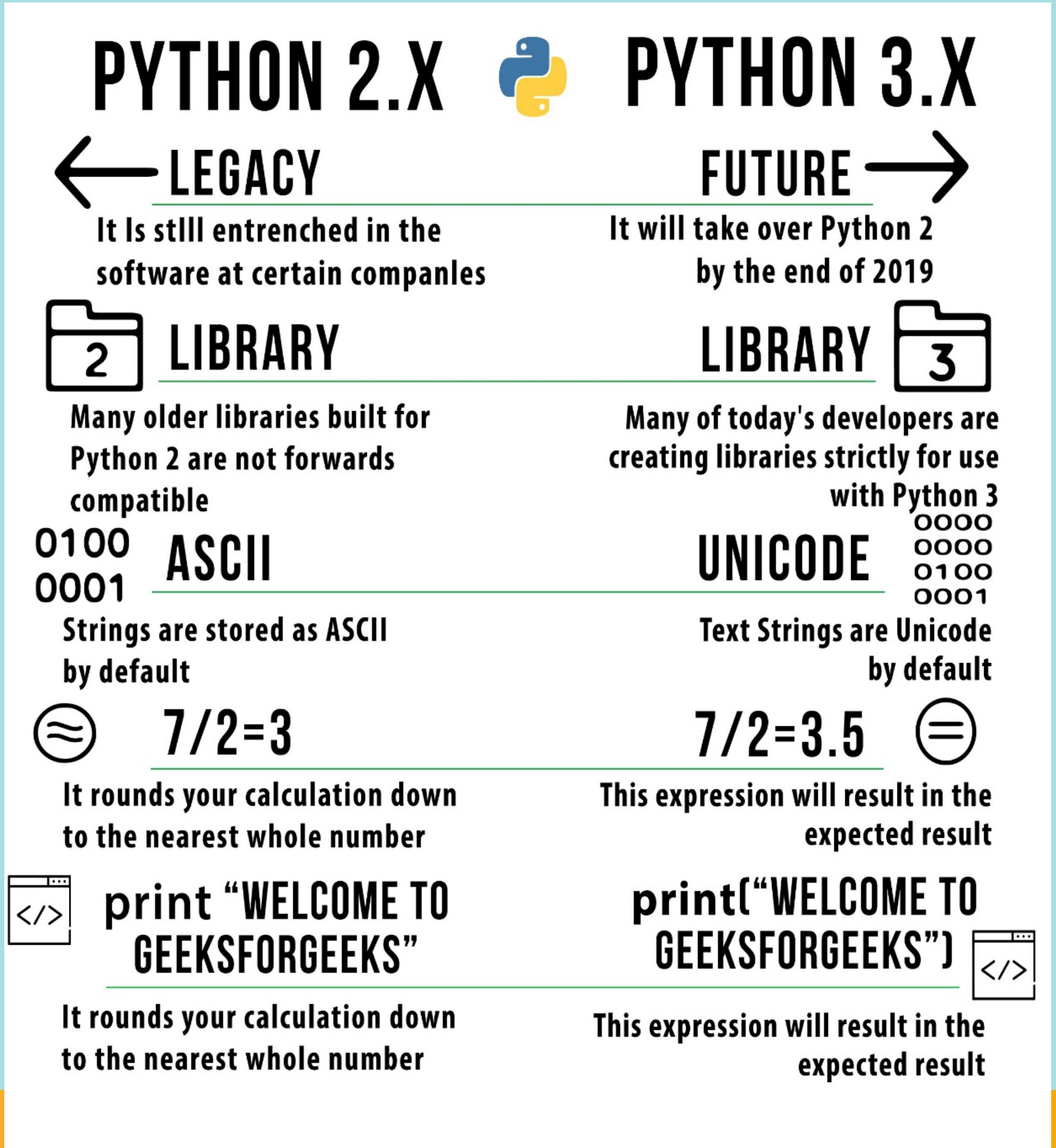
112-1基礎程式設計(2)

亞大資工系

Python Versions



PYTHON 2.X | PYTHON 3.X

LEGACY — It Is still entrenched in the software at certain companies

FUTURE — It will take over Python 2 by the end of 2019

LIBRARY — Many older libraries built for Python 2 are not forwards compatible

LIBRARY — Many of today's developers are creating libraries strictly for use with Python 3

ASCII — Strings are stored as ASCII by default

UNICODE — Text Strings are Unicode by default

7/2=3 — It rounds your calculation down to the nearest whole number

7/2=3.5 — This expression will result in the expected result

print "WELCOME TO GEEKSFORGEEKS" — It rounds your calculation down to the nearest whole number

print("WELCOME TO GEEKSFORGEEKS") — This expression will result in the expected result

Python new features:

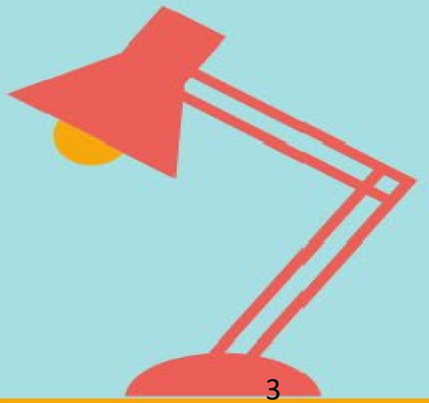
- Python 3.10: Structural Pattern Matching
- Python 3.6 : f-Strings
- Python 3.3 : Virtual Environments
- Python 3.2: Argparse

Python powerful features:

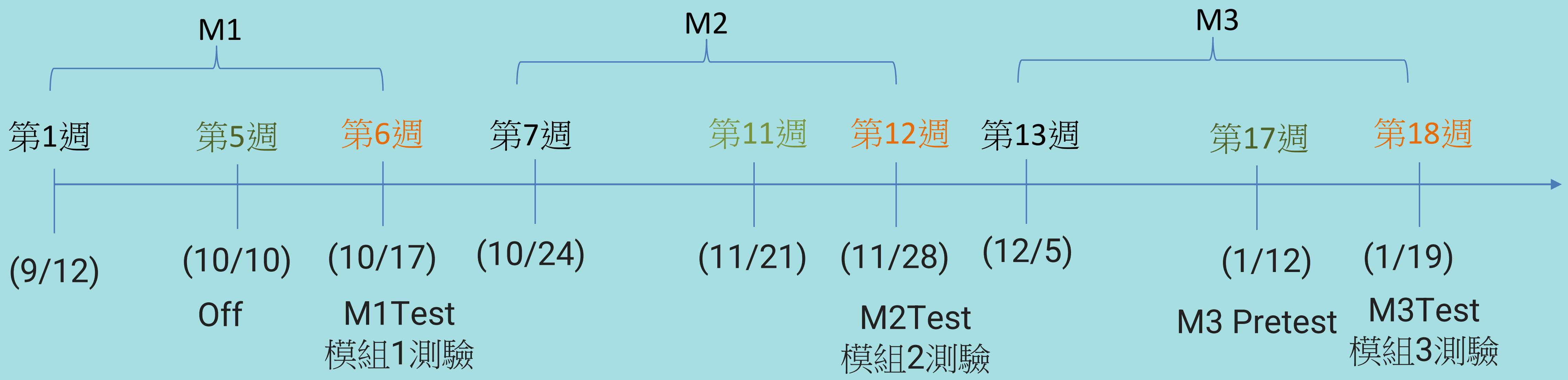
- Iterators
- Generators
- Decorators
- Context Managers

課程大綱

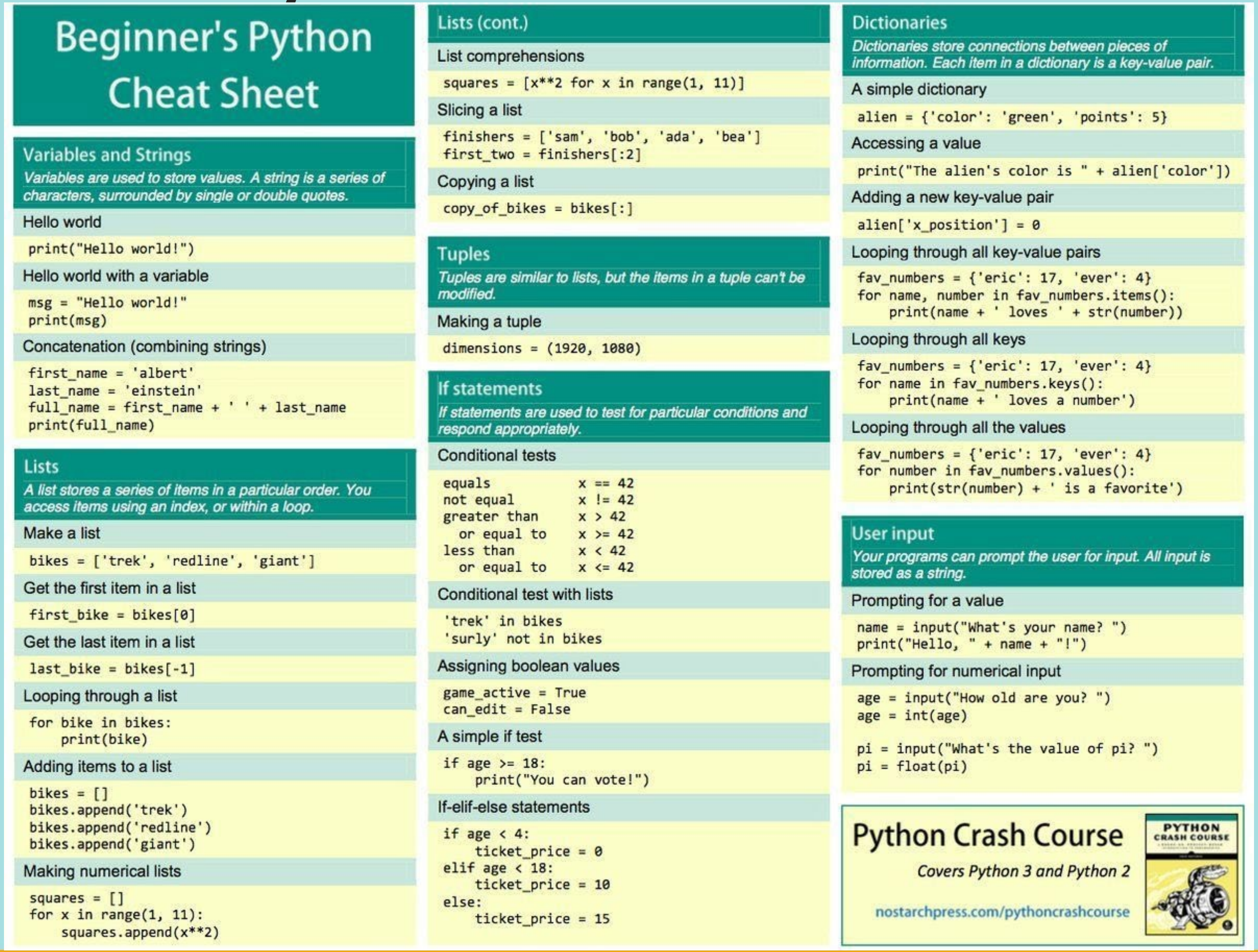
- W1-Python簡介及程式工具
- W2-變數和運算
- W3-迴圈和格式化輸出
- W4-判斷式和容器
- W5-字串處理和輸出入
- W6-M1測驗
- W07-字典容器
- W08-檔案處理
- W09-函數
- W10-進階流程控制
- W11-進階運算和生成器
- W12-M2測驗
- W13-進階函數
- W14-類別
- W15-進階類別
- W16-模組和套件
- W17-進階設計
- W18-M3測驗



課程進度


M1
M2
M3
第1週
(9/12)
第5週
(10/10)
Off
第6週
(10/17)
M1Test
模組1測驗
第7週
(10/24)
第11週
(11/21)
第12週
(11/28)
M2Test
模組2測驗
第13週
(12/5)
第17週
(1/12)
M3 Pretest
第18週
(1/19)
M3Test
模組3測驗

Python cheat sheet



Beginner's Python Cheat Sheet

Variables and Strings

Variables are used to store values. A string is a series of characters, surrounded by single or double quotes.

Hello world

```
print("Hello world!")
```

Hello world with a variable

```
msg = "Hello world!"
print(msg)
```

Concatenation (combining strings)

```
first_name = 'albert'
last_name = 'einstein'
full_name = first_name + ' ' + last_name
print(full_name)
```

Lists

A list stores a series of items in a particular order. You access items using an index, or within a loop.

Make a list

```
bikes = ['trek', 'redline', 'giant']
```

Get the first item in a list

```
first_bike = bikes[0]
```

Get the last item in a list

```
last_bike = bikes[-1]
```

Looping through a list

```
for bike in bikes:
    print(bike)
```

Adding items to a list

```
bikes = []
bikes.append('trek')
bikes.append('redline')
bikes.append('giant')
```

Making numerical lists

```
squares = []
for x in range(1, 11):
    squares.append(x**2)
```

Lists (cont.)

List comprehensions

```
squares = [x**2 for x in range(1, 11)]
```

Slicing a list

```
finishers = ['sam', 'bob', 'ada', 'bea']
first_two = finishers[:2]
```

Copying a list

```
copy_of_bikes = bikes[:]
```

Tuples

Tuples are similar to lists, but the items in a tuple can't be modified.

Making a tuple

```
dimensions = (1920, 1080)
```

If statements

If statements are used to test for particular conditions and respond appropriately.

Conditional tests

```
equals                x == 42
not equal             x != 42
greater than          x > 42
  or equal to         x >= 42
less than             x < 42
  or equal to         x <= 42
```

Conditional test with lists

```
'trek' in bikes
'surly' not in bikes
```

Assigning boolean values

```
game_active = True
can_edit = False
```

A simple if test

```
if age >= 18:
    print("You can vote!")
```

If-elif-else statements

```
if age < 4:
    ticket_price = 0
elif age < 18:
    ticket_price = 10
else:
    ticket_price = 15
```

Dictionaries

Dictionaries store connections between pieces of information. Each item in a dictionary is a key-value pair.

A simple dictionary

```
alien = {'color': 'green', 'points': 5}
```

Accessing a value

```
print("The alien's color is " + alien['color'])
```

Adding a new key-value pair

```
alien['x_position'] = 0
```

Looping through all key-value pairs

```
fav_numbers = {'eric': 17, 'ever': 4}
for name, number in fav_numbers.items():
    print(name + ' loves ' + str(number))
```

Looping through all keys

```
fav_numbers = {'eric': 17, 'ever': 4}
for name in fav_numbers.keys():
    print(name + ' loves a number')
```

Looping through all the values

```
fav_numbers = {'eric': 17, 'ever': 4}
for number in fav_numbers.values():
    print(str(number) + ' is a favorite')
```

User input

Your programs can prompt the user for input. All input is stored as a string.

Prompting for a value

```
name = input("What's your name? ")
print("Hello, " + name + "!")
```

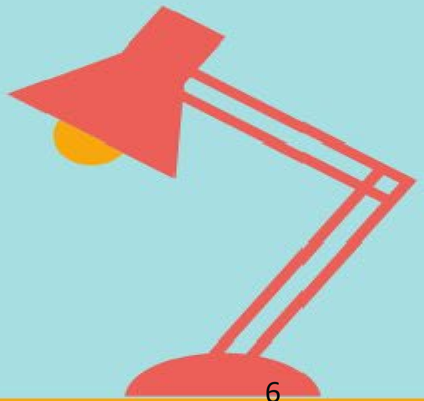
Prompting for numerical input

```
age = input("How old are you? ")
age = int(age)

pi = input("What's the value of pi? ")
pi = float(pi)
```

本週主題:變數和運算

- Essential-基本的
  - IPO model: input–process–output (輸入-處理-輸出)
    - Input: input()函數, 變數型別轉換(int, float)
    - Process: 算術運算子和表示式(expressions)
    - Process: 運算子運算優先順序
    - Output: print()函數的參數(sep, end, file, flush)
  - 標準庫math的應用
  - 程式註解
- Advanced-進階的
  - 多行的字串
  - MarkDown語法



啟思博Kissipo 學習法

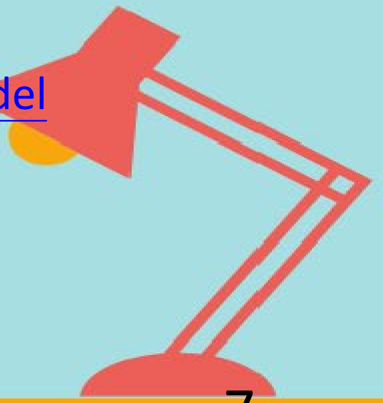
Kissipo = KISS principle + IPO model

KISS principle

"keep it simple, stupid" or "keep it stupid simple", is a design principle noted by the U.S. Navy in 1960.
https://en.wikipedia.org/wiki/KISS_principle

IPO model

The input–process–output (IPO) model is a widely used approach in systems analysis and software engineering for describing the structure of an information processing program or other process.
https://en.wikipedia.org/wiki/IPO_model



Kissipo Learning for Programming with Python(PWP)

Courseware: Notebook+ Github

(1) 使用Notebook(Google Colab)教學。
(2) 使用Github建立教案

Keep:

Variables and assignment
operator and expression
left-hand side and right-hand side
unpacking

S&S:

help(), type(), len(), size()

IPO-I: input

input()
int(), float(), str()
split(), map()

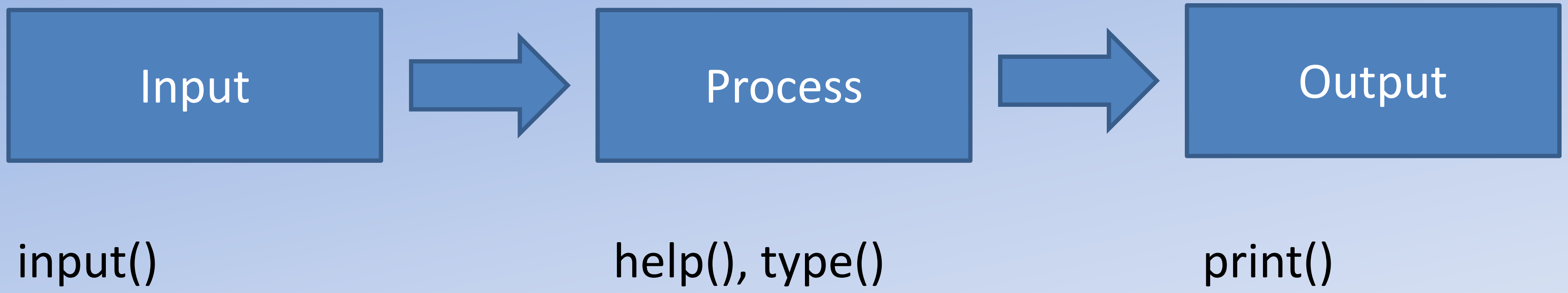
IPO-P: Process

變數宣告, 資料容器
for-loop/while-loop
if, elif, else
range()

IPO-O: output

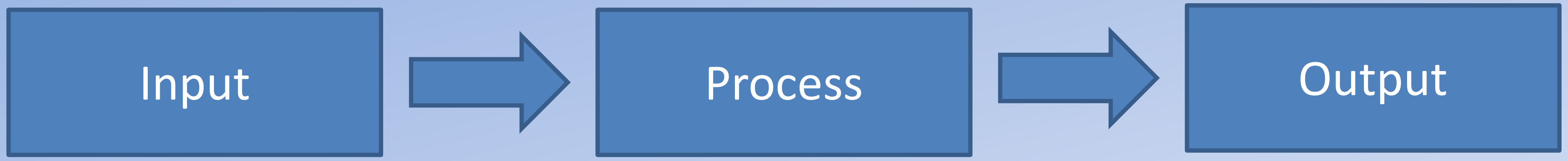
print()
open(), write()

IPO Model(1)



本章基本觀念是同學要知道：
輸入用input(), 輸出用print()
help()可以查看函數或類別的說明
type()可以查看變數的型別

IPO Model (2)



input()輸入一個變數
使用int()轉換成整數變數
使用float()轉換成浮點數變數

算術運算子
運算子優先序
程式中的字串表示
註解

print()函數的參數sep 和end
Escape Sequence (逸出序列)

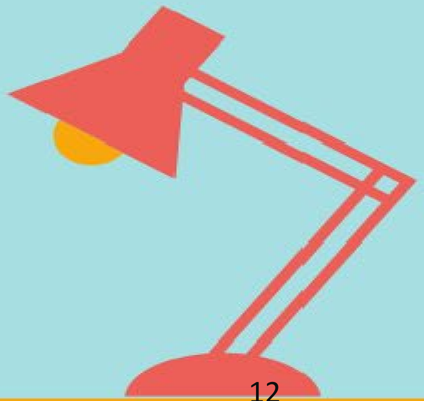
本章基本觀念是同學要知道：
如何用input()輸入不同型別的數數
輸出print()有兩個參數sep 和end來控制輸出
Python 的算數運算包括：加減乘除(+ - * /), 次方(**), 商(//)和取餘數(%)。加減乘除(+ - * /), 次方(**)的計算結果是浮點數。商(//) 和取餘數(%)的計算結果是整數。

Topic 1(主題1)-輸入一個整數或浮點數

- Step 1: 輸入一個整數
- Step 2: 輸入一個浮點數

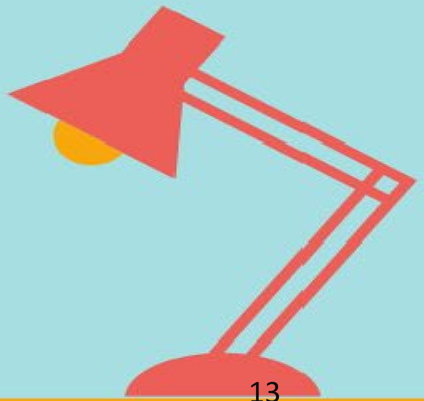
Topic 2(主題2)-算術運算子和表示式 (expressions)

- Step 1: 加減乘除
- Step 2: 商和餘數
- Step 3: 次方



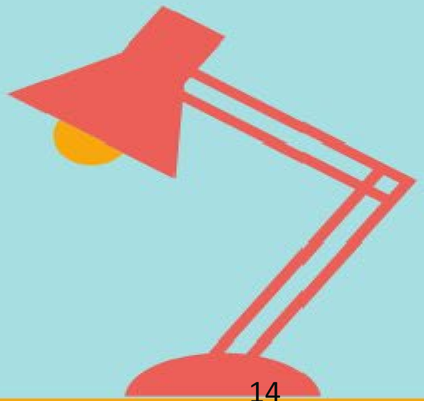
Topic 3: 運算子優先序（Operator precedence）

- Step 1: 先乘除後加減, 括號優先
- Step 2: 次方比加減乘除優先



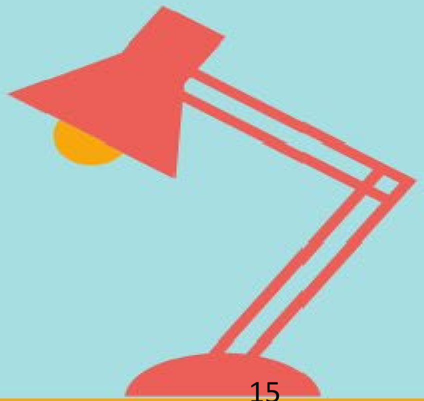
Topic 4(主題3)-標準庫math的應用

- Step 1: 計算pi 和sin(π/3) 函數
- Step 2: 使用math標準庫的pi 和sin 函數
- Step 3: 使用as
- Step 4: 使用標準庫math的角度(degree)和弧度(radian)轉換



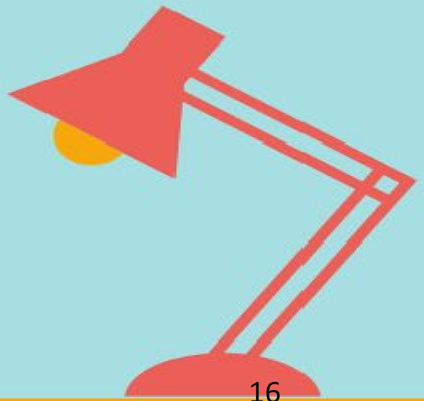
Topic 5: print()函數的參數

- Step 1: Hello World with 其他參數
- Step 2:Escape Sequence (逸出序列)



Topic 6: 多行的字串

- Step 1: 使用 字串尾部的\來建立長字串
- Step 2: 使用六個雙引號來建立長字串 ''' … ''' 或 """ … """



Toic 7: 原始碼的字元編碼 (encoding)

- 預設 Python 原始碼檔案的字元編碼使用 UTF-8。在這個編碼中，世界上多數語言的文字可以同時被使用在字串內容、識別名 (identifier) 及註解中 --- 雖然在標準函式庫中只使用 ASCII 字元作為識別名，這也是個任何 portable 程式碼需遵守的慣例。如果要正確地顯示所有字元，您的編輯器需要能夠認識檔案為 UTF-8，並且需要能顯示檔案中所有字元的字型。

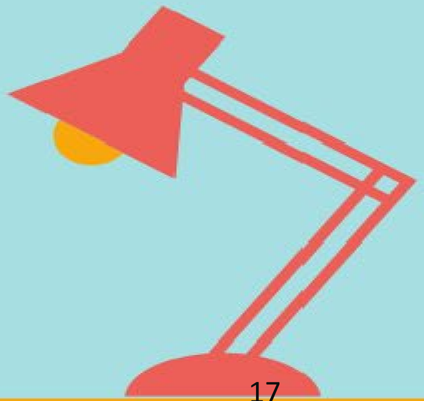
- 如果不使用預設編碼，則要聲明檔案的編碼，檔案的第一行要寫成特殊註解。語法如下：

- # -*- coding: encoding -*-
- 其中， encoding 可以是 Python 支援的任意一種 codecs。

- 比如，聲明使用 Windows-1252 編碼，源碼檔案要寫成：

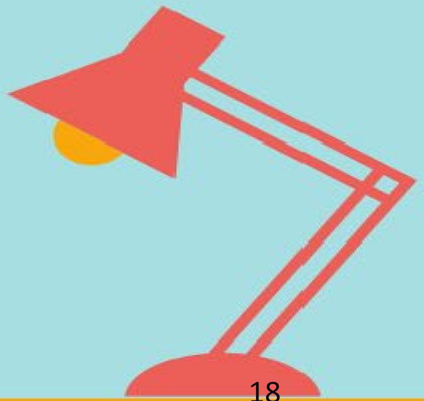
- # -*- coding: cp1252 -*-
- 第一行的規則也有一種例外情況，在源碼以 UNIX "shebang" line 行開頭時。此時，編碼聲明要寫在檔案的第二行。例如：

- # !/usr/bin/env python3
- # -*- coding: cp1252 -*-



Topic 8: MarkDown語法

- Step 1: 標題
- Step 2: 分隔線
- Step 3: 粗體及斜體
- Step 4: 清單



Thanks! Q&A