**基本觀念**

* # : 註解(不會列印)
* SHIFT+ENTER可以RUN
* (, end='\n') : 換行 (也可插在中間)
* (, end=' ') : 不換行
* 前一行尾端加上/ : 可讓下一行被視為上一行

**Keyword**

* **keyword.kwlist :** 可以秀出所有keyword
* **print(keyword.iskeyword('if'))** : 可以確認是不是keyword

**Identifier**

* 規則 : 數字、符號或字母組成、開頭不能是數字、大小寫有分、不能跟keyword一樣
* **"???".isidentifier() :** 可用來確認是不是identifier

**運算元**

* **a \*\* b =** a的b次方
* **a // b =** 除完後無條件捨去
* **a % b =** 餘數
* **a += b** ===== a = a + b
* **x<<2 :** x往左邊二進位

**Input**

* width = input('input a width: ') : 使用者可以看到input a width: 然後填答
* float(number) : 可讓輸入的數字被知道是數字且是小數點
* int(number) : 可讓輸入的數字被知道是數字且是整數
* print(type(number)) : 可知道系統現在認定number是str(字串)、float、complex(虛數)或是int
* 應用(算面積) :

width = input('input a width: ')

rec\_area = float(width) \* float(height)

print('are of rectangle is {}'.format(rec\_area))

**序列與雜湊**

* 序列分為**string**、**[list]、(tuple)**

1. **String :** '字串' 或是 '''圈住很多行'''
   * + 1. **.lower() :** 全部小寫
       2. **.upper() :** 全部大寫
       3. **.isupper() :** 是大寫
       4. **.isdigit() :** 是數字
       5. **.isalpha() :** 是英文
       6. **.replace(' ','@')** : 取代成後者
       7. **.split(‘ 文字中間用甚麼區隔 ’) :** 拆分文字,後面可再接[0]表示只要區隔前的文字
       8. **.strip() :** 把空格去掉
2. **[list] :** 有序**、**內容可變、可重複
   * + 1. 搜索 : **list** [0]→可看第一個元素
       2. [0,1,2,3]=[-4,-3,-2,-1]
       3. [2:4] : 第3個元素到第4個
       4. [::2] : 從頭到尾 以2為間隔
       5. **可用功能** : 搜索、.append(加在尾端)、.clear(全清空)、.count(數總共有幾個)、.extend(加入)、.insert(位置,插入的東西)、.pop (去掉最後一個)、.remove()、.reverse()、.sort (照順序排好, reverse=True降冪)、print len(list)看長度、print ‘’.join(LIST)可把list內容都合起來、print(sum(LIST))、print(max(LIST))看最大值、print(min(LIST))看最小值
3. **(tuple) :** 有序**、**內容不可變、可重複、可相加(變大)

* 雜湊分為**{set}、{dictionary}**

1. **{set} :** 內容不可重複、無序、無索引
   * + 1. **新增 :** .add("a")、.update(["a", "b", "c"])
       2. **刪除 :** .remove(["d"])、.pop()隨機刪除
       3. **(a & b) :** 交集
       4. **(a | b) :** 聯集
       5. **(a ^ b) :** a或b
       6. 定義一個空白的set : S=**set()**
2. **dictionary={key : value} :** 不可重複
   * + 1. **可用功能** :key in dictionary(true or false)、len(dictionary)、print(dictionary.values())
       2. **刪除 :** del dictionary [key]、.pop(key)
       3. **新增 :** dictionary[key] = value、.update ( [key = value] )

* 布林型別 : true or false

1. **x and y :** 若其中一個是false,結果就是false
2. **x or y :** 若任意一個是true,結果就是true
3. **x in y :** 確認x有沒有在y群集裡
4. **x is not y :** 確認身分
5. **x == y** : 詢問是否等於
6. **x != y** : 詢問是否不等於
7. **0是**false,**1**是true

* **迭代**

1. 可用**it = iter(list)**做為一個閱讀器
2. **print (next(it))**即可將list內的內容依序讀出來
3. 也適用於字典、字串…..

**流程控制(if/else/elif)與迴圈(while/continue/break)**

* 四個space = 縮行
* **if** grade > 90:

print('excellent')

**elif** grade > 60:

print('good')

**else:**

print('you have to practice more!!!')

* **while True(無限迴圈)**:可用**break**打破,或用**continue**讓程序直接迴圈 :

**while…:**

**if…:**

**break/continue**

* **for A in range(10):**

**print(A)** = 0 1 2 3 4 5 6 7 8 9

**for A in range(5, 10):**

**print(A)** = 5 6 7 8 9

**for A in (5, 10):**

**print(A)** = 5 10

* **my\_list = [10, -20, 15]**

**for index, i in enumerate(my\_list):**

**print(index,i)**

0 10

1 -20

2 15

* **my\_dict = {'isaac':50, 'andy':80, 'judy':20}**

**for i in my\_dict.values():**

print(i) = 50 80 20

**字串格式**

* 印出a、b、c的方法

**print('a = {}, b = {}, c = {}'.format(a, b, c))**

**或 print('a = {0}, b = {1}, c = {2}'.format(a, b, c))**

* **print('{0[0]}, {0[1]}'.format(my\_list)) :** 可印出my\_list的第一二項
* 虛數可用 **: {0.real}{0.imag}**
* **{:10d} :** 以十位數表示(ex: 3)

**{:10f} :** 浮點法以十位數表示(ex: 2.112321)

**{:10.2f} :** 浮點法小數點後留兩位表示(ex: 2.11)

**{:10s} :** 字串十位數表示(ex:hello )

**{:<10d}** : 靠左(ex:3 )

**{:^10d}** : 置中(ex: 3 )

**{:.2%}.format(B/A) :** 顯示B除以A的值,以百分比顯示到小數點第二位

**{:b} :** 二進位

**{:o} :** 八進位

**{:x} :** 十六進位

**{:#b} :** 顯示0b (ex:0b100110110110101111100)

**{:,} :** 數字每三位以”,”隔開(ex: 1,273,212)

**函數**

* **自訂函數** : **def 函數名字(變數or留空):**

**動作**

**return ???**

**print(函數名字(變數or留空)** →→→ **???**

* 函數命名規則 : 數字、符號或字母組成、開頭不能是數字
* 變數
  + 1. 不確定變數有幾個,可以在變數前輸**”\*”**
    2. 當變數是字典, 可以在變數前輸**”\*\*”**
    3. 變數可先定義 : **”變數=??”**
* **if \_\_name\_\_ == "\_\_main\_\_":** 用來確認如果目前是執行主程式,就跑主程式的函數
* **內建函數**

**round(a) :** 無條件捨去小數點

**pow(x,y) :** xy

**divmod(x,y) :** x除以y的(商數,餘數)

**abs(a) :** 取絕對值

**bool(a) :** 轉換成布林

**ord(b) :** 字元轉序號

**chr(c) :** 序號轉字元

**eval(d) :** 做四則運算

* **區域變數 :** 變數用在def下,就只能用在函數內

**全域變數 :** 變數出現在函數以外的地方,如果函數要借用時,可先輸入**”** **global xxx”(其實好像也不用)**

* **匿名函數** : 把函數精簡成一行

**Ex :** **ANS = lambda x, y: x \* y**

**print(ANS (5,13)) = 65**

* **Filter函數**: 篩選list中符合函數條件的項目

**Ex :** **my\_list = [1,2,3,4,5]**

**E=list(filter(lambda x: (x == 1), my\_list))**

**print (E) = 1**

* **Map函數**: 把list每個項目都應用在函數中

**my\_list = [5, 10 ,15]**

**sqrt\_list = list(map(lambda x: x\*\*2, my\_list))**

**print(sqrt\_list) = [25, 100, 225]**

* **Reduce函數**: 將list多個項目拆解作合併

**from functools import reduce**

**my\_list = [5, 8, 10]**

**reduce\_result = reduce(lambda x,y: x+y, my\_list)**

**print(reduce\_result) = ((5+8)+10)=23**

* **遞迴** : 在函數裡面呼叫函數自己

有呼叫次數的限制,通常是3000,但可以更改 : **sys.setrecursionlimit(4000)**

* **動態規劃** : 因遞迴會不斷重複計算,浪費時間,因此可以令一個list紀錄迴圈的結果,未來要再用的時候直接叫出來就好了

**Class與物件導向**

* **class 開頭為英文大寫的名字():** 
  + - 1. class裡的變數(attribute屬性) : 要取用須加上**self.xxx**
      2. class裡的函數(method方法) : ( )內的變數第一個一定要設**self**
      3. class裡可定義**\_\_init\_\_()**函數,每次呼叫會自動執行
* **class Banks():**

**def \_\_init\_\_(self, uname, money): # 初始化方法**

**self.name = uname # 設定存款者名字**

**self.balance = money # 設定所存的錢**

**self.bankname = "Taipei Bank" # 設定銀行名稱**

**def save\_money(self, money): # 設計存款方法**

**self.balance += money # 執行存款**

**print("存款 ", money, " 完成") # 列印存款完成**

**def get\_balance(self): # 獲得存款餘額**

**print(self.name, " 目前餘額: ", self.balance)**

**isaacbank = Banks('isaac', 100) # 定義物件**

**print("目前開戶銀行 ", isaacbank.bankname) # 列出目前開戶銀行isaacbank.save\_money(500) # 存款500元**

**isaacbank.get\_balance() # 獲得存款餘額**

>>>目前開戶銀行 Taipei Bank

>>>存款 500 完成

>>>isaac 目前餘額: 600

* **capsule(封裝) :** 在attribute或method前面加”**\_\_**”,就不能在class外取用的時候被竄改,以防被外人亂用

**def \_\_init\_\_(self):**

**self.­­\_\_ balance = 0**

isaacbank.\_\_balance = 10000 **# 類別外直接竄改存款餘額**

isaacbank.get\_balance() **>>>isaac 目前餘額: 0**

* **繼承 :** 子類別可以獲得父類別所有功能

**class Son子類別(Father父類別,** **Uncle叔父類別):**

**pass**

**>>>print出來會有父類別和叔父類別的功能,但以父類別為主**

* **多型 :** 當子類別定義一個與父類別名稱一樣的method時,各自的功能都可以維持。若想改變變數,可在子類別的method下使用**super().method名稱 (改變的內容)**

**例外處理**

* **例外處理 :** 當無法執行結果(預期之外)時

**可用try/except :**

a = 3 , b = 0

**try(可以的話執行):**

c = a/b

**except(不然就):**

print('error')

**else(可以執行的話就一定要):**

print('always execute')

* **raise**可用來列舉例外 **:**

if len(my\_passwd) < 10:

**raise Exception('**password is too short**')**

**>>>** password is too short

* **assert**可用來檢查例外 :

**assert** len(my\_passwd) > 10, **'password is too short'**

**>>>** password is too short

**讀寫檔案**

* **f = open("檔案名稱", "r或是w", encoding='utf-8' )**

>>>打開檔案並採取read mode或write mode

>>>解碼比較不會有亂碼

**f.close()**

>>>用完一定要關掉

* 或是用**with open('檔案名稱', 'r', encoding='utf-8') as f:**

>>>這樣寫完後面就不用關掉

* 檔案名稱可以是**路徑**,ex: “**./**my\_document”
* 可print查詢 :
  + - 1. **name :** 檔案名稱
      2. **closed :** T or F
      3. **mode :** r或w
      4. **read(20) :** 讀20個字
      5. **f.readlines() :** 不換行一句一句讀(每個段落尾會顯示”\n”)

若n=f.readlines(),n[0] =第一句

* + - 1. **f.write('文字') :** 寫進檔案
* 寫入CSV檔 :

**with open('CSV檔名', 'a') as f:**

**f.write('name,grade\n')**

>>>可直接用”,”隔開字元並直接用”\n”換行,就可以生成excel了

>>>**a代表直接加在原有資料後面**

**正規表示法(Regular Expression)**

* 可抓出符合指定規則的字串
* 可用網站 : <https://pythex.org/>
* **txt = "The rain in Spain"**

**import re**

**x = re.match("^The.\*Spain$", txt)**

**if x:**

**print("YES! We have a match!")**

→→→ 檢查txt是不是從”The”開頭,中間包含”數個任意字元”,並以”Spain”結尾

* **re.match** : 從開頭就要符合

**re.search** : 整句有出現就好

**re.findall** : 整句符合的都抓出來

* **^a :** 以a開頭

**b$ :** 以b結尾

**a? :** a出現0到1次

**. :** 任意字元

**a\* :** a出現0到多次

**a+ :** a出現1到多次

**a{n} :** a出現n次

**a{n,} :** a至少出現n次

**a{n,m} :** a出現n~m次

**\** : 逃脫字元,加在符號前面可使其失效

**\.** : 真的只要”.”

**\d :** 0-9

**\D :** 非0-9

**\w** : 數字、英文或 \_\_

**\s** : 空白

**|** : 或

**[A-Z]** : 大寫英文

* x = re.search("y", txt)

**print(x.start()) :** 看y在第幾格發生

**print(x.group())** **:** 列出符合條件的字串(也可用**re.findall**)

* x = 'admit, commit, emit, omit, permit'

re.findall("[dor]mit", x) →→→ ['dmit', 'omit', 'rmit']

**Datetime模組(日期時間)**

* **印出現在時間**

**import datetime**

**T = datetime.datetime.today()**

**print(T)**

或是

**from datetime import datetime as T**

**print(T.now())**

* **印出某個時間並格式化日期**

**T = datetime.datetime(2018,6,1)**

**T.strftime("%Y-%m-%d %H:%M:%S")**

* **常用格式**
  + - 1. **%a :** Wed
      2. **%A :** Wednesday
      3. **%d :**日(數字)
      4. **%b :** Dec
      5. **%B :** December
      6. **%m :** 12
      7. **%Y :** 2018
      8. **%y :** 18
      9. **%H :** 00-23
      10. **%I :** 00-12
      11. **T.date()** : 顯示到日
      12. **T.day** : 印出日
* **時差**

**時差 = datetime.timedelta(hours=3)**

**newT = T + 時差**

**Time模組(算時間)**

**import time**

* **time.time() :** 從 1970/1/1 00:00:00 至今的秒數
* **time.ctime() :** 當地時間

**>>>** Fri Nov 13 10:14:25 2020

* **time.strftime("%m/%d/%Y", time.localtime()) :** 依指定格式顯示當地時間
* **time.sleep(1.5) :** 暫停 1.5 秒再執行下個動作

**Math模組(算數)**

**import math**

* **math.ceil(?) : 無條件進位**
* **math.floor(?) : 無條件捨去**
* **math.trunc(?) = 去掉小數點**
* **math.sqrt(?) : 開根號**
* **math.fabs(?) = 絕對值**
* **math.sin(?)**
* **math.cos(?)**
* **math.pi = 3.1415**
* **math.log(1) = math.log10(1) = math.log(1,10) = 0**

**Random模組(隨機產生亂數)**

**import random**

* **print(random.random()) :** 產生亂數
* **print(random.randrange(3, 9)) :** 亂數在3-9之間產生
* **random.seed(隨意數字) :** 可固定每次跑出來的亂數都一樣
* **random.shuffle(list) :** 把list裡的物件洗牌
* **print(random.sample(list, 2)) :** 隨機抽list裡的2個物件
* **print(random.choices(list, weights = [10, 1, 1], 14)) :** 照著分配的權重抽出list裡的物件,總數為14

**Os模組(處理作業系統)**

**import os**

* **os.environ.get('PATH')** : 取得環境變數內容
* **os.path.expanduser('~')** : 取得家目錄路徑
* **os.getcwd()** : 取得目前工作路徑
* **os.rename("原檔名", "新檔名")**
* **os.remove("檔名")**
* **os.mkdir("資料夾名稱")** : 創建資料夾
* **os.path.exists("檔名") = os.path.isfile("檔名")** : 檢查檔案是否存在(T/F)

常會用if not os.path.exists("./XXXX"),

os.mkdir("./XXXX ") 以免每print一次就創一個

* **os.listdir('檔名')** : 查此資料夾下有甚麼檔案

# Shutil模組(移動/複製/打包/壓縮/解壓檔案)

**import shutil**

* **shutil.copytree('欲複製的資料夾名', '複製出的資料夾名')**
* **shutil.copyfile("欲複製的檔名", "複製出的檔名")**
* **shutil.rmtree('欲移除的資料夾名')**
* **shutil.move("欲移動的檔名", "目的地檔案夾")**

# zipFile模組(對zip檔進行壓縮或解壓縮)

**from zipfile import ZipFile**

**with ZipFile(“zip檔名”, 'r') as z:**

**z.printdir()** #印出檔案內容物

**z.extractall()** #解壓縮

# Json模組(將Json字串轉換成物件)

* **Json語言 :** 常用於資料儲存與轉換
* **import json**

**json形式的字串x = '{ "name":"jim", "age":25, "city":"Taiwan"}'**

**json.loads(x) : 轉換json成list,每個list都是一個dictionary**

**json.dumps(x) : 轉換json成str**

**json.dump(x,F) : 轉完直接寫入檔案(搭配讀寫檔案使用)**

# Logging模組(顯示警示程度)

* **Logging模組 :** 當最後實際開始跑程式卻不希望print出來時,可以依警示程度留下警語就好
* **警示程度**

**logging.notset < logging.debug < logging.info < logging.warning < logging.error < logging.critical**

* **import logging**
* **FORMAT = '%(asctime)s %(levelname)s: %(message)s'**

#如要顯示時間可設

* **logging.basicConfig(level=logging.DEBUG, format=FORMAT)**

#將開始警示程度拉低到DEBUG(因內建是**warning以上**才開始警示)

* **logging.debug('message1')**

#指定警語並print出來

* **logging.basicConfig(level=logging.DEBUG, filename='檔名', filemode='w', format=FORMAT)**

#如要產生PYTHON日誌

# Jieba模組(將中文句子斷成單字)

* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install jieba”
* **import jieba**
* **精確模式(預設) : words1=jieba.cut(句子, cut\_all=False)**
* **全模式 : words1=jieba.cut(句子, cut\_all=True)**
* **搜索引擎模式 : words1=jieba.cut\_for\_search(句子)**

**for word in words1:**

**print(word+'/', end='')**

* 不滿意斷詞方式時,可自訂 :
* **加入字詞 : jieba.add\_word(斷詞 , freq=None, tag=None)**
* **載入自定義詞庫 :**

創建一個Text File,一個ROW打一個詞

匯入: **jieba.load\_userdict(檔名)**

* **停用字字典(不想要某些詞時) :**

也是創建一個Text File打入不想要的詞,讀取他並把每個詞用迴圈放進一個list,若我們的單字not in這個list,就可以把這個單字加進新的list

# Pillow模組(用來處理圖像)

* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install pillow”
* **from PIL import Image**

**P = Image.open("圖檔名")**

* **打開圖片 : P.show()**
* **顯示長寬 : print(P.size)**
* **改變長寬 : P.resize( (寬,長), Image.BILINEAR )**
* **轉180度 : P.transpose(Image.ROTATE\_180)**
* **左右翻轉 : P.FLIP\_LEFT\_RIGHT**
* **上下翻轉 : P.FLIP\_TOP\_DOWN**
* **轉存成png檔或其他名稱 : P.save("新檔名")**

# Pytube模組(用來下載youtube影片)

* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install pytube”
* **from pytube import YouTube**

**link = '影片連結'**

* **下載影片 : YouTube(link).streams.first().download()**
* **下載最高解析度影片 : YouTube(link).streams.get\_highest\_resolution().download()**

# QRcode模組(用來產生QRcode)

* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install qrcode”
* **import qrcode**

**A = "要生成QRcode的網址"**

**Q = qrcode.QRCode(version=1,box\_size=10,border=5)**

**Q.add\_data(A)**

**Q.make(fit=True)**

**P = Q.make\_image(fill='black', back\_color='white')**

**P.save('圖檔名')**

# Pytesseract模組(辨識照片內的文字)

* 先下載(並記住執行檔存放路徑) :

<https://github.com/UB-Mannheim/tesseract/wiki>

我的執行檔存放路徑 : C:/Program Files/Tesseract-OCR/tesseract.exe

* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install pytesseract”
* **from PIL import Image**

**import pytesseract**

**pytesseract.pytesseract.tesseract\_cmd = '執行檔存放路徑'**

**P = Image.open("圖檔名")**

**A = pytesseract.image\_to\_string(P)**

**print(A)**

# 自定義模組(自己寫好的py檔)

* **import py檔名**(不包含.py)

**py檔名.函數指令**

* **from 以上提到的任意模組 import 函數**

**函數指令**

# 執行緒(執行程式的程序)與平行處理(多個執行緒同時執行)

* **開啟電腦其他程式 :**

**from subprocess import Popen**

**Popen('欲開啟的程式.exe')**

* **平行處理**

**import threading**

**t = threading.Thread(target = 要跟主函數同時跑的子函數名稱 )**

**t.start()**

**執行主函數**

**t.join() >>>等待子函數執行結束**

# SQLite資料庫

* 下載sqlitebrowser : [**https://sqlitebrowser.org/**](https://sqlitebrowser.org/)(no installer版本)
* **import sqlite3**

**DB = sqlite3.connect("資料庫名.db")** # 資料庫連線或創建

**C = DB.cursor()** # 類似創建滑鼠

**C.execute(“SQL指令”)**  # 執行SQL指令

**DB.commit()**  # 刷新資料庫

**DB.close()**  # 關閉資料庫連線

* **SQL指令**
  + - **創建表格 :**

**'create table 表格名(項目1 int, 項目2 text….)'**

* + - **刪除表格 :**

**"drop table if exists 表格名"**

* + - **查詢資料 :** 
      * 1. **'select \* from 表格名'** #真的可以用”\*”
        2. **'select 項目1, 項目2 from 表格名 where 項目3 =”XXX” '**
        3. **“select 項目1, 項目2 from 表格名 where 項目3 > '%A' “ % ('1000')** # %A只是變數,括號外%(‘’)才把內容寫出來
        4. 可把結果轉換成tuple : **results=C.fetchall()**再用for/in迴圈print出 >>>通常[0]是ID
    - **插入data :** 
      * 1. **x = (項目1, 項目2, 項目3)**

**sql = 'insert into 表格名 values(?,?,?)'** #真的是用”?”

**C.execute(sql,x)**

* + - * 1. **sql = 'insert into 表格名(項目1, 項目2, 項目3) values(XX,XX,XX)'**
    - **更新data :**

**' update 表格名 set 項目 = "XXX" where 項目2 = “X” '**

* + - **刪除data :**

**' delete from 表格名 where 項目2 = “X” '**

# MySQLdb

* 下載mysql : [**https://dev.mysql.com/downloads/installer/**](https://dev.mysql.com/downloads/installer/)
* 為第三方模組需安裝 : ANACONDA/Environments/open terminal/key入”pip install mysqlclient”
* **import MySQLdb**

**DB = MySQLdb.connect("host","username(root)","password*(850710)*"," newschema名" )**

然後就跟SQLite一模一樣了

**保險起見加入try:**

**C.execute(“SQL指令”)**

**except:**

**DB.rollback()**

* **SQL指令**
  + - **創建資料庫 :**

**“create database資料庫名”**

* + - **刪除資料庫 :**

**“drop database if exists 資料庫名”**

* + - 其他功能跟SQLite一模一樣

# Tkinter(建立GUI-視窗產生器)

**from tkinter import \***

**A = Tk(className=”視窗名稱”)**

* **產生視窗**

**A.mainloop()**

* **產生按鈕**

**B = Button(A, text='按鈕上的文字', width=按鈕寬度, height=按鈕高度, bg='#33CC33', fg='#FFFFFF', activebackground='#44DD44', activeforeground='#FFFFFF')**

**B.pack()**

* **產生問題及輸入欄**

**Q = Label(A, text="問題文字")**

**Q.pack(side = LEFT)**

**E = Entry(A, bd =輸入欄邊界厚度)**

**E.pack(side = RIGHT)**

* **產生純文字框**

**T = Text(A)**

**T.insert(INSERT, "輸入文字")**

**T.pack()**

**Tag文字**

**T.tag\_add("tag主題名稱", "第幾行.第幾個位置", "第幾行.第幾個位置")**

**T.tag\_config("tag主題名稱", background="背景顏色", foreground="文字顏色")**

* **產生滾輪**

**S = Scrollbar(A)**

**S.pack( side = RIGHT, fill = Y )**

**滾輪旁的內容**

**LIST = Listbox(A, yscrollcommand = S.set )**

**LIST.insert(END, "輸入內容")**

**LIST.pack(side = LEFT, fill = BOTH )**

**S.config( command =LIST.yview )**

* **選項按紐**

**def sel():**

**label.config(text = "選完後出現的文字")**

**var = IntVar()**

**R1 = Radiobutton(A, text="選項1的文字", variable=var, value=1,command=sel)**

**R1.pack( anchor = W )**

**label = Label(A)**

**label.pack()**

* **Check紐**

**Var1 = IntVar()**

**C1 = Checkbutton(A, text = "選項1的文字", variable = Var1, onvalue = 1, offvalue = 0, height=選項高度,width =選項寬度)**

**C1.pack()**

* **跳出警示窗**

**A.withdraw()**

**messagebox.showerror("Error", "Error message")**

**messagebox.showwarning("Warning","Warning message")**

**messagebox.showinfo("Information","Informative message")**

* **跳出照片視窗**

**class Window(Frame):**

**def \_\_init\_\_(self, master=None):**

**Frame.\_\_init\_\_(self, master)**

**self.master = master**

**self.pack(fill=BOTH, expand=1)**

**load = Image.open("圖檔名")**

**render = ImageTk.PhotoImage(load)**

**img = Label(self, image=render)**

**img.image = render**

**img.place(x=0, y=0)**

**app = Window(A)**

**A.wm\_title("視窗名稱")**

**A.geometry("?x?")**

* **Menubar**

**menubar = Menu(A)**

**選條1 = Menu(menubar, tearoff=0)**

**選條1.add\_command(label="選項1", command=點了這個選項會做的事)**

**選條1.add\_command(label="Exit", command=root.quit)**

**menubar.add\_cascade(label="選條1名字", menu=選條1)**

**A.config(menu=menubar)**