

第八章 電腦模擬

我們一路走來已經寫過不少的程式，透過規劃，進行測試，找出錯誤，一步一步完成階段性的版本，我們不只是在發展，同時也在維護程式功能的完整。實際軟體發展的過程會更複雜，尤其當需求漸增，程式日趨龐大，測試、偵錯及維護也就變得越來越困難。

程式的發展與軟體的構築在今日都以經成為專門學問，有許多不同的見解與主張，也有許多的方法論，甚至由方法論擴展不同程式語言的設計哲學。我們的選擇是Python，原因不外乎是規劃、測試與除錯的程式發展流程更為簡便，同時Python也提供了例外處理機制。

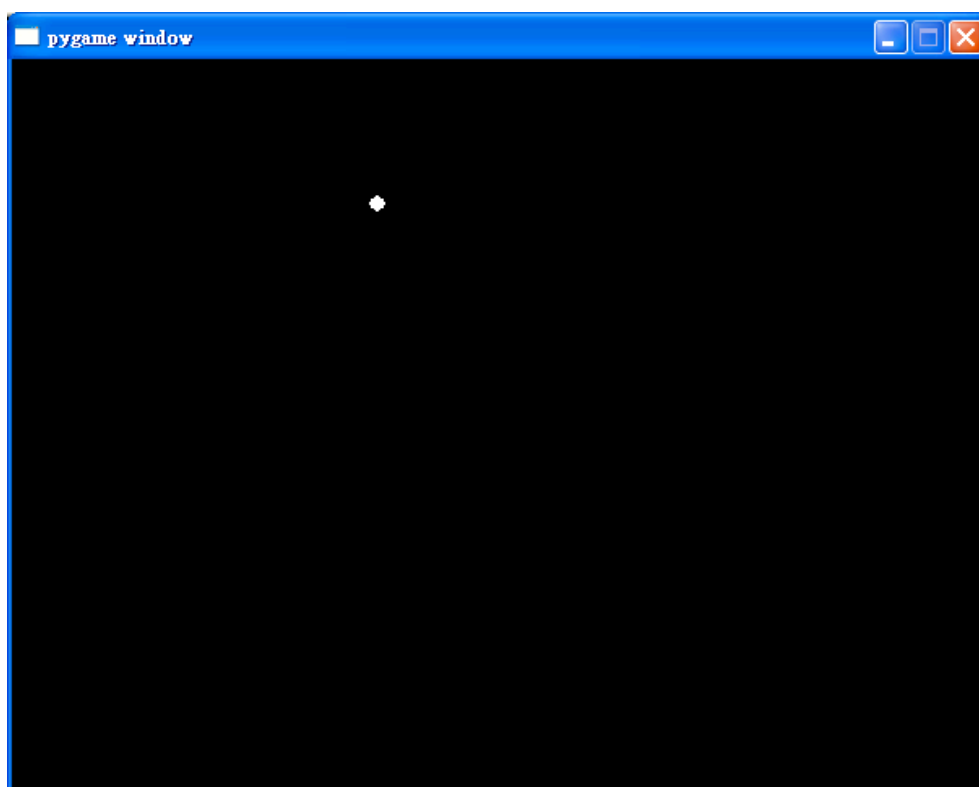
然而程式可以不僅僅是去做我們要程式做的事情，針對許多未知的情況，由以往的經驗與累積的資料，經過計算我們可以預測可能發生的結果，這就是**電腦模擬**。。

例如對颱風路徑的推測，需要收集地面及高空的氣壓、溫度、溼度、風速等等資料，匯集製作圖表，也依往年的氣象資料與颱風的實際路徑，然後統合所有的資料，藉由電腦快速的計算能力，模擬出可能的情况。

商業上也廣泛利用電腦模擬，譬如保險公司要推出一項產品，訴求客戶、理賠事項、保費收取等等，在推出之前都經過嚴密的試算，當然，保險公司以賺錢為主要目標，同時也要讓消費者感到有所保障，製造雙贏的局面。

實務上的電腦模擬有很多種，常見的試算表軟體Excel的目的就是模擬試算之用。其實電腦遊戲也是利用模擬的方式與使用者互動，使用者的操作符合這種情境，電腦也隨即做出相對的反應。

假設我們要設計球類的遊戲，我們就需要寫出模擬球移動的程式，如下圖假設球移動到視窗的邊緣會以隨機的角度彈回。



嗯，我們還沒正式介紹Python的**圖形使用者介面**。Pygame是Python關於遊戲設計的第三方模組庫，我們在下一篇中才會正式介紹，雖說如此，我們已經寫了個簡易的鬥獸棋遊戲。現在，來模擬看看吧！

鬥獸棋的模擬

因為鬥獸棋需要使用者輸入要操作的棋子以及方向，棋子共有四種，方向也有四種，假設是隨機的進行過程，那麼我們可以借用random模組中的randint()函數，以1及4為參數，藉此隨機產生數字1、2、3、4中的任一個。

但是輸入的都必須是字母，因此我們先用字典設定字母與數字間的對應。

```
#動物棋子對應的編號
animal = {1:"e", 2:"t", 3:"c", 4:"m"}
#方向對應的編號
direction = {1:"u", 2:"d", 3:"r", 4:"l"}
```

這樣一來，在遊戲迴圈中便能以randint(1,4)當作animal及direction的key，然後直接呼叫handle函數。

```
handle(animal[randint(1,4)], direction[randint(1,4)])
```

因為不在要求使用者輸入，所以我們再把以下兩行刪除。

```
print "操作「象」鍵入e，「虎」鍵入t，「貓」鍵入c，「鼠」鍵入m。"
print "往「上」鍵入u，往「下」鍵入d，往「左」鍵入l，往「右」鍵入r。"
```

然後把handle()函數中的“請輸入正確的方向！！”改為“不可能的情況！！”，而“請輸入正確的棋子！！”改為“該棋子已不存在！！”。同時我們利用系統清除螢幕的指令，在程式中加進上一章寫過的clear()函數，然後放在遊戲迴圈的開頭。

每一輪迴我們再借用time模組中的sleep()函數，放到遊戲迴圈的結束，讓整個畫面顯示一秒，好讓我們目睹該輪迴做了什麼事情。所有改寫過的程式碼摘錄如下。

```
#-*- coding: UTF-8 -*-

"""簡化版鬥獸棋遊戲。"""

from time import sleep
from random import randint
from os import system
from sys import platform
from vector.point import Point
from jungle.checker import Checker, Jungle

#初始條件設定
#參與遊戲的動物棋子
players = {"e":[Jungle("E"),Point(1,1)], "t":[Jungle("T"),Point(1,2)], \
           "c":[Jungle("C"),Point(2,2)], "m":[Jungle("M"),Point(2,1)]}
#動物棋子對應的編號
animal = {1:"e", 2:"t", 3:"c", 4:"m"}
#方向對應的編號
direction = {1:"u", 2:"d", 3:"r", 4:"l"}

#印出棋盤的函數
def status(square):
    """以4×4的方格顯示棋盤。"""

    print "棋盤狀態顯示"
    print
    for j in range(square):
        for i in range(square):
```

```

        if players.has_key("e") and i == players["e"][1].x \
            and j == players["e"][1].y:
            print "象",
        elif players.has_key("t") and i == players["t"][1].x \
            and j == players["t"][1].y:
            print "虎",
        elif players.has_key("c") and i == players["c"][1].x \
            and j == players["c"][1].y:
            print "貓",
        elif players.has_key("m") and i == players["m"][1].x \
            and j == players["m"][1].y:
            print "鼠",
        else:
            print "口",
        print

```

#往上移動

```

def up(name):
    if players[name][1] - Point(0,1) in players[name][0].board:
        players[name][1] = players[name][1] - Point(0,1)
    else:
        print "超過邊界囉！"

```

#往下移動

```

def down(name):
    if players[name][1] + Point(0,1) in players[name][0].board:
        players[name][1] = players[name][1] + Point(0,1)
    else:
        print "超過邊界囉！"

```

#往左移動

```

def left(name):
    if players[name][1] - Point(1,0) in players[name][0].board:
        players[name][1] = players[name][1] - Point(1,0)
    else:
        print "超過邊界囉！"

```

#往右移動

```

def right(name):
    if players[name][1] + Point(1,0) in players[name][0].board:
        players[name][1] = players[name][1] + Point(1,0)
    else:
        print "超過邊界囉！"

```

#從字典型態變數players的value找相對應的key

```

def findkey(p):
    j = 0
    for i in players.items():
        if players.items()[j][1][1] == p:
            return players.items()[j][0]
        j = j + 1
    return 0

```

#處理兩棋子的相遇情況

```

def encounter(first, second):
    if players[first][0].capture(players[second][0]):
        del players[second]
        return True
    else:
        return False

```

#控制遊戲的函數

```

def handle(name, direction):
    try:
        if direction == "u":
            if findkey(players[name][1] - Point(0,1)):
                if encounter(name, findkey(players[name][1] - Point(0,1))):
                    up(name)
            else:
                up(name)
        elif direction == "d":
            if findkey(players[name][1] + Point(0,1)):
                if encounter(name, findkey(players[name][1] + Point(0,1))):
                    down(name)
            else:
                down(name)
        elif direction == "l":
            if findkey(players[name][1] - Point(1,0)):
                if encounter(name, findkey(players[name][1] - Point(1,0))):
                    left(name)
            else:
                left(name)
        elif direction == "r":
            if findkey(players[name][1] + Point(1,0)):
                if encounter(name, findkey(players[name][1] + Point(1,0))):
                    right(name)
            else:
                right(name)
        else:
            print "不可能的情況!!"
    except KeyError:
        print "該棋子已不存在!!"

def clear():
    if platform == "win32":
        return "cls"
    elif platform == "linux2" or "darwin":
        return "clear"
    else:
        return False

def run():
    while len(players) > 1:
        system(clear())
        status(4)

        #操作提示
        print
        handle(animals[randint(1,4)], direction[randint(1,4)])

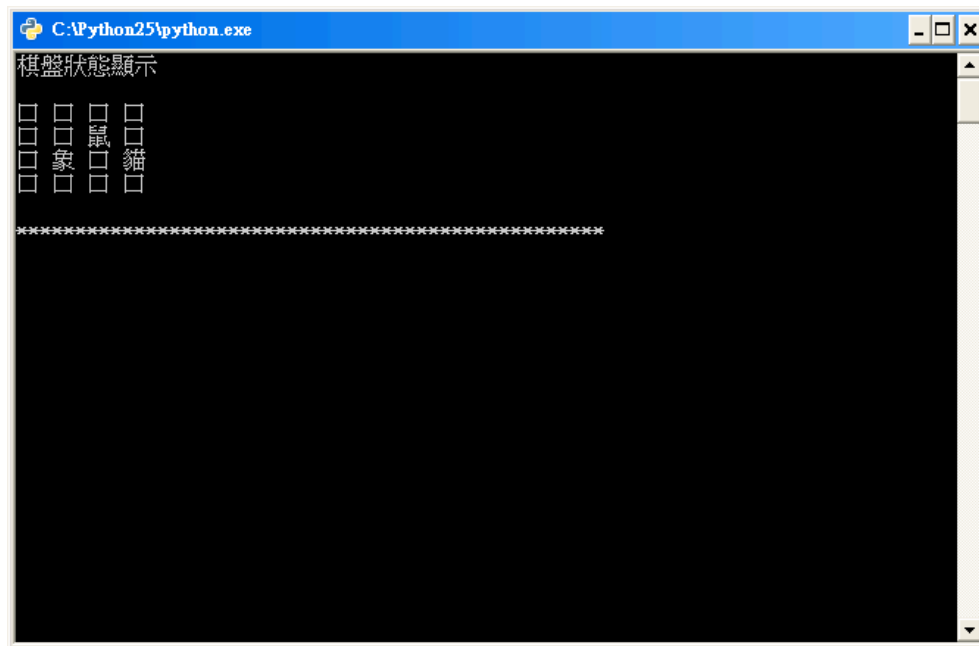
        #印出間隔線
        print "*" * 50
        print
        sleep(1)

    #印出遊戲勝利者
    for winner in players.values():
        if winner[0].alive == True:
            print winner[0].name, "是最後的存活者!"

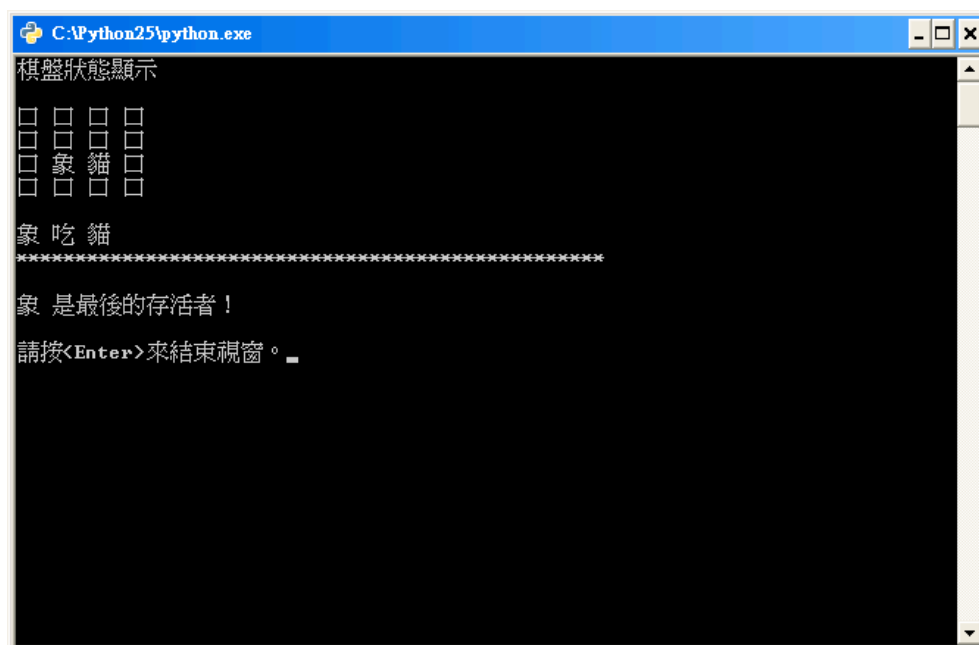
```

```
if __name__ == "__main__":  
    run()  
    print  
    raw_input("請按<Enter>來結束視窗。")
```

實際來跑跑看吧！



「虎」很快就被吃掉了，到結束會是怎麼樣的呢？



顯然這次的模擬「象」是最後遊戲的勝利者。

幕後運作

從整個鬥獸棋遊戲的模擬過程，每一次的輪迴程式就對畫面作一次更新，實際上我們所見到螢幕顯示出的畫面，也是在非常短的時間內不斷的重複輸出目前的作業階段。大體上，我們所用的軟體包括作業系統，程式都用類似的方式設計。

我們學習寫程式的目的除了能自己寫程式，不論解決問題或是完成某項工作，其餘更是要了解電腦幕後運作的道理，曉得這道理，就不會對龐雜的電腦系統內部感到困惑，進而掌握個中奧妙。

選單的互動模式

接下來我們稍做複習，利用上一章寫的選單程式，我們把寫過的範例整合在一起，程式開始執行時印出現在時間，另外選單預計包含下列五種功能。

- (1) 計算費伯納西數列
- (2) 計算平均值
- (3) 讀取文字檔
- (4) 猜數字遊戲
- (5) 簡易鬥獸棋遊戲

這五種功能我們都已經寫過程式，現在則是要利用選單程式整合一起。選單程式儲存在menu.py，其餘都以模組，也就是把定義放在其他的Python檔案之中。第一項計算費伯納西數列儲存在fibonacci.py中，程式碼如下。

```
#!/-- coding: UTF-8 --

#計算費伯納西數列
def fi(n):
    f = [0, 1] #儲存費伯納西數列
    m = 2
    #計算的迴圈
    while m <= n:
        r = f[m-1] + f[m-2]
        f = f + [r]
        m = m + 1
    return f

def print_fi():
    try:
        n = int(raw_input("請輸入所要計算的費伯納西數： "))
        print "此數為", fi(n)[n]
    except ValueError:
        print "輸入的並非整數!!"
```

Note

因為沒有打算讓這個模組成為可執行程式，所以沒有

```
if __name__ == "__main__":
```

的部份。

第二項計算整數平均值儲存在average.py之中，程式碼如下。

```
#!/-- coding: UTF-8 --

#依使用者輸入的整數計算平均值
def average():
    count = 0
```

```
total = 0
state = True
print "鍵入quit就離開輸入。"
while state:
    try:
        r = raw_input("請輸入整數： ")
        if r != "quit":
            n = int(r)
            if n <= 0:
                raise ValueError
            total = total + n
            count = count + 1
        else:
            state = False
    except ValueError:
        print "請不要輸入小於或等於零的整數，或是除了整數以外的符號。"

print
print "輸入數字的平均為", total/count
print
```

第三項讀取文字檔儲存在readfile.py之中，程式碼如下。

```
#-*- coding: UTF-8 -*-
```

#讀純文字檔

```
def read():
    try:
        f = open(raw_input("請輸入檔名： "), "r")
        print "*" * 50
        print f.read()
        print "*" * 50
        f.close()
        print
    except IOError:
        print "沒有這個檔案！"
```

第四項猜數字遊戲儲存在guess.py，程式碼如下。

```
#-*- coding: UTF-8 -*-
```

```
from random import randint

def guess():
    answer = randint(1,99)
    print " **猜數字遊戲** "
    print
    state = True
    while state:
        try:
            guess = int(raw_input("請輸入1到99的數字： "))
            if guess == answer:
                print
                print "正確答案！"
                print
                state = False
            elif answer < guess < 100:
                print "太大囉！再試一次。"
            elif 0 < guess < answer:
                print "太小囉！再試一次。"
            else:
                raise ValueError
```

```
except ValueError:
    print "請不要輸入小於1或大於99的整數，或是除了整數以外的符號。"
```

第五項簡易鬥獸棋遊戲儲存在board4.py之中，程式碼如前所述。印出現在時間部份的模組則放在now.py之中，程式碼如下。

```
#-*- coding: UTF-8 -*-

from time import sleep, asctime

weeks = {"Mon": "星期一", "Tue": "星期二", "Wed": "星期三", "Thu": "星期四", \
         "Fri": "星期五", "Sat": "星期六", "Sun": "星期日"}
months = {"Jan": "一月", "Feb": "二月", "Mar": "三月", "Apr": "四月", \
         "May": "五月", "Jun": "六月", "Jul": "七月", "Aug": "八月", \
         "Sep": "九月", "Oct": "十月", "Nov": "十一月", "Dec": "十二月"}
moment = asctime()

def now():
    print "今天是西元",moment[20:], "年", months[moment[4:7]],
    print moment[8:10], "號", weeks[moment[:3]]
    print "現在的時間是", moment[11:19]
    print

def seconds():
    now()
    print
    print "倒數十秒開始....."
    i = 10
    while i > 0:
        print "時間還有", i, "秒",
        sleep(1)
        print "\r",
        i = i - 1
    print " *20,
    print
    print "十秒過後.....你看見消失的時間了嗎？"
    print "現在的時間是", asctime()[11:19]
    print
```

接著我們繼續完成menu.py，這時候我們所要做的事情就是把定義都引入menu.py之中，然後依上一章所完成骨架程式，視情況呼叫所需函數或其他定義，程式碼如下。

```
#-*- coding: UTF-8 -*-

from sys import exit, platform, version
from os import system
from fibonacci import fi, print_fi
from average import average
from readfile import read
from guess import guess
from board4 import *
from time import sleep, asctime
from now import now, seconds

def whatplatform():
    if platform == "win32":
        print "您所使用的是MS-Windows作業系統。"
    elif platform == "linux2":
        print "您所使用的是Linux作業系統。"
    elif platform == "darwin":
        print "您所使用的是Mac作業系統。"
    else:
```



```
print "程式未能偵測到您所使用的作業系統種類。"

def clear():
    if platform == "win32":
        return "cls"
    elif platform == "linux2" or "darwin":
        return "clear"
    else:
        return False

def fun1():
    try:
        system(clear())
        print_fi()
        raw_input("請按<Enter>繼續。")
    except TypeError:
        print "系統並無清除螢幕的指令，程式仍繼續運作....."
        print_fi()
        raw_input("請按<Enter>繼續。")

def fun2():
    try:
        system(clear())
        average()
        raw_input("請按<Enter>繼續。")
    except TypeError:
        print "系統並無清除螢幕的指令，程式仍繼續運作....."
        average()
        raw_input("請按<Enter>繼續。")

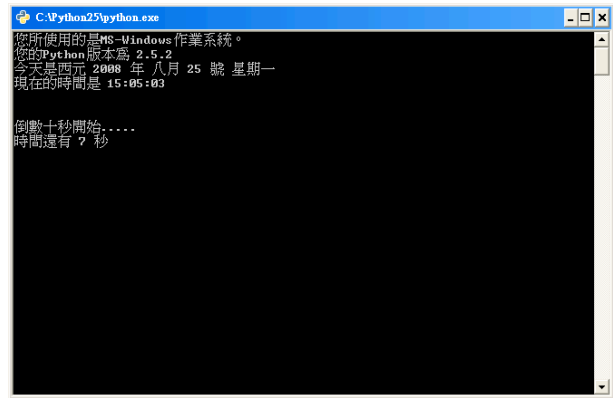
def fun3():
    try:
        system(clear())
        read()
        raw_input("請按<Enter>繼續。")
    except TypeError:
        print "系統並無清除螢幕的指令，程式仍繼續運作....."
        read()
        raw_input("請按<Enter>繼續。")

def fun4():
    try:
        system(clear())
        guess()
        raw_input("請按<Enter>繼續。")
    except TypeError:
        print "系統並無清除螢幕的指令，程式仍繼續運作....."
        guess()
        raw_input("請按<Enter>繼續。")

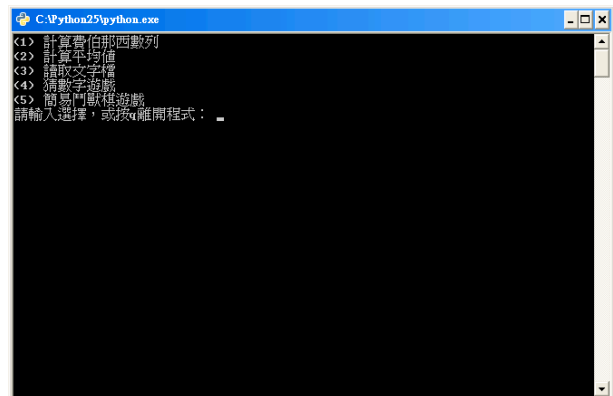
def fun5():
    try:
        system(clear())
        run()
        raw_input("請按<Enter>繼續。")
    except TypeError:
        print "系統並無清除螢幕的指令，程式仍繼續運作....."
        run()
        raw_input("請按<Enter>繼續。")
```

```
def main():
    while True:
        try:
            system('clear()')
            print "(1) 計算費伯那西數列"
            print "(2) 計算平均值"
            print "(3) 讀取文字檔"
            print "(4) 猜數字遊戲"
            print "(5) 簡易鬥獸棋遊戲"
            c = raw_input("請輸入選擇，或按q離開程式： ")
            if c == "1":
                fun1()
            elif c == "2":
                fun2()
            elif c == "3":
                fun3()
            elif c == "4":
                fun4()
            elif c == "5":
                fun5()
            elif c == "q":
                exit()
            else:
                print "您所輸入的不是1或2或q。"
                raw_input("請按<Enter>繼續。")
        except TypeError:
            print "系統並無清除螢幕的指令，程式仍繼續運作....."
            print "(1) 計算費伯那西數列"
            print "(2) 計算平均值"
            print "(3) 讀取文字檔"
            print "(4) 猜數字遊戲"
            print "(5) 簡易鬥獸棋遊戲"
            c = raw_input("請輸入選擇，或按q離開程式： ")
            if c == "1":
                fun1()
            elif c == "2":
                fun2()
            elif c == "3":
                fun3()
            elif c == "4":
                fun4()
            elif c == "5":
                fun5()
            elif c == "q":
                exit()
            else:
                print "您所輸入的不是1或2或q。"
                raw_input("請按<Enter>繼續。")
if __name__ == "__main__":
    whatplatform()
    print "您的Python版本為", version[:5]
    seconds()
    raw_input("請按<Enter>繼續。")
    main()
```

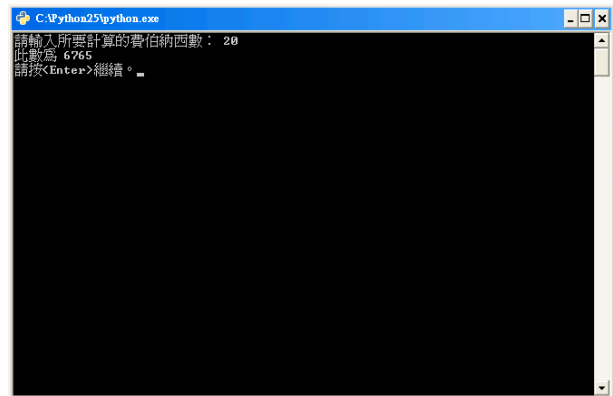
程式一開始，自然是印出目前時間。



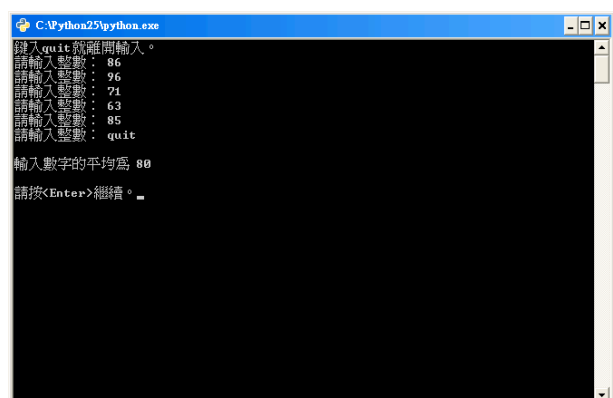
倒數結束按下<Enter>，就進入了主選單。



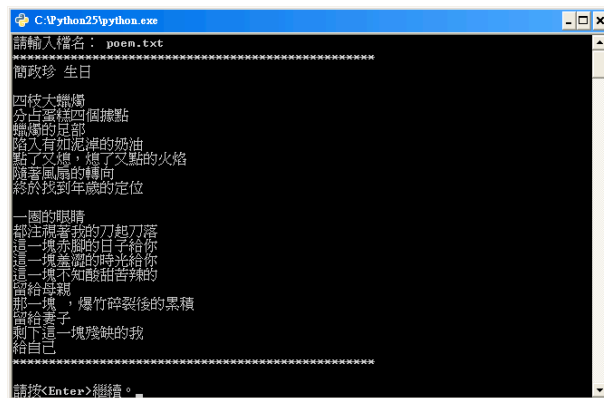
輸入1，就可以計算費伯納西數。



按下<Enter>回主選單，輸入2，就可以計算平均值。



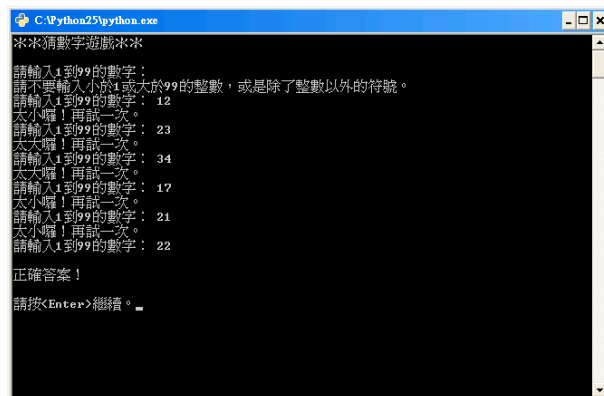
再一次按下<Enter>回主選單，輸入3，讀取純文字檔。



```
C:\Python25\python.exe
請輸入擇名： poem.txt
*****
聞政珍 生日
*****
四枝大蠟燭
分占蛋糕四個據點
燭燭的足部
陷入有如泥濘的奶油
點了文煙，燐了文點的火焰
隨著風扇的轉向
終於找到年歲的定位

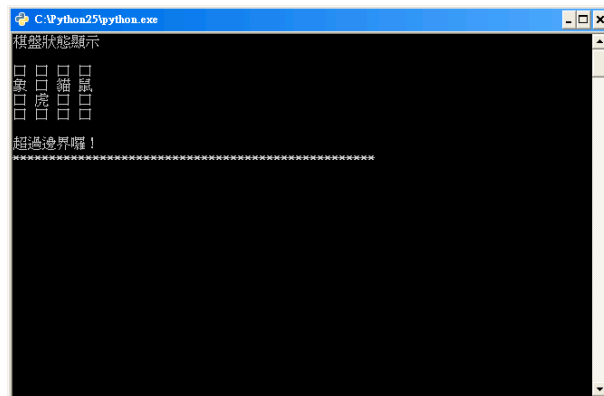
一團的眼睛
都主視著我的刀起刀落
這一塊赤腳的日子給你
這一塊羞澀的時光給你
這一塊不知酸甜苦辣的
當給母親
那一塊，爆竹碎裂後的累積
白給妻子
剩下這一塊殘缺的我
給自己
*****
請按<Enter>繼續。
```

再一次按下<Enter>回主選單，輸入4，進行猜數字遊戲。



```
C:\Python25\python.exe
***猜數字遊戲***
請輸入1到99的數字：
請不要輸入小於1或大於99的整數，或是除了整數以外的符號。
請輸入1到99的數字： 12
太小囉！再試一次。
請輸入1到99的數字： 23
太大囉！再試一次。
請輸入1到99的數字： 34
太大囉！再試一次。
請輸入1到99的數字： 17
太小囉！再試一次。
請輸入1到99的數字： 21
太小囉！再試一次。
請輸入1到99的數字： 22
正確答案！
請按<Enter>繼續。
```

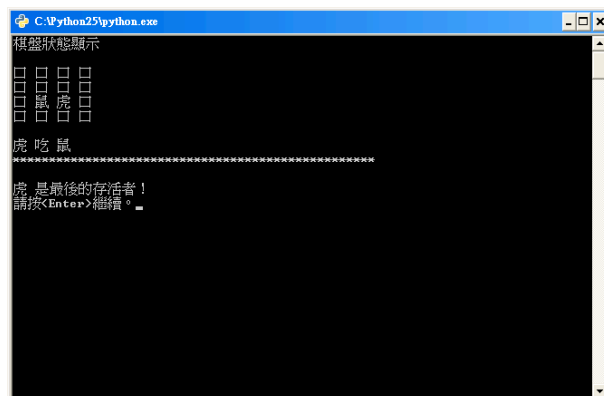
再一次按下<Enter>回主選單，輸入5，模擬簡化的鬥獸棋遊戲。



```
C:\Python25\python.exe
棋盤狀態顯示
□ □ □
□ □ 貓 鼠
□ 虎 □
□ □ □

超過邊界囉！
*****
```

最後的結果是...



```
C:\Python25\python.exe
棋盤狀態顯示
□ □ □
□ □ 虎 鼠
□ 鼠 虎 □
□ □ □

虎 吃 鼠
*****
虎 是最後的存活者！
請按<Enter>繼續。
```

顯然，這次是「虎」贏得了最後的勝利。

下一步

這就是以使用者互動的模式運作，我們可以持續發展menu.py，精巧妥善的利用印出符號美化顯示，也可以多加入一些給使用者的訊息，因為一個好的應用程式介面，不僅僅容易讓使用者理解程式的功能，也便於利用程式。

然而這仍是在命令列模式底下，事實上我們日常所用的電腦都已經是圖形介面，圖形更給人直覺，容易了解。下一篇開始，我們介紹第三方模組庫中的Pygame，這是專門為遊戲所開發的模組庫，也是我們將會介紹的第一個圖形使用者介面。