2.3 判斷式

在日常生活中，我們經常會遇到一些需要做決策的情況，然後再依決策結果從事不同的事件，例如：暑假到了，如果所有學科都及格的話，媽媽就提供經費讓自己與朋友出國旅遊；如果有某些科目當掉，暑假就要到校重修了！程式設計也一樣，常會依不同情況進行不同處理方式，這就是「判斷式」。

2.3.1 單向判斷式（if…）

「if…」為單向判斷式，是 if 指令中最簡單的型態，語法為：

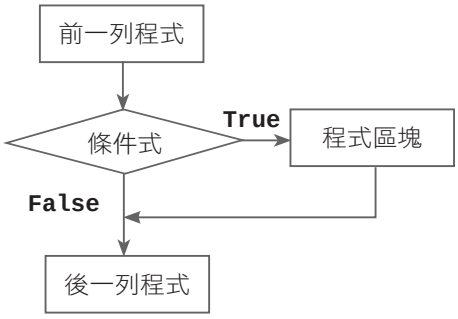
```
if 條件式：
    程式區塊
```

當條件式為 True 時，就會執行程式區塊的敘述；當條件式為 False 時，則不會執行程式區塊的敘述。

條件式可以是關係運算式，例如：「x > 2」；也可以是邏輯運算式，例如：「x > 2 or x < 5」，如果程式區塊只有一列程式碼，則可以合併為一列，直接寫成：

```
if 條件式：程式碼
```

以下是單向判斷式的流程圖：



Python 程式碼縮排格式

大部分語言如 C、Java 等，多是以一對大括號「{}」來表示程式區塊，例如：

```
if(score>=60) {
    grade = "及格";
}
sum = sum + score
```

程式區塊

下一列程式

Python 語言以冒號「:」及縮排來表示程式區塊，縮排為 1 個 Tab 鍵或 4 個空白鍵，例如：

```
if(score>=60):
    grade = "及格"
sum = sum + score
```

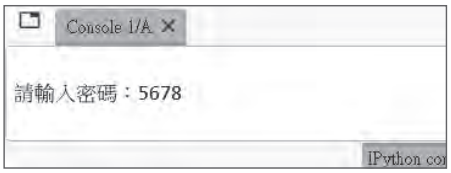
程式區塊

下一列程式

Tab 鍵或 4 個空白鍵

範例：密碼輸入判斷

讓使用者輸入密碼，如果輸入的密碼正確 (1234)，會顯示「歡迎光臨！」；如果輸入的密碼錯誤，則不會顯示歡迎訊息。



程式碼：password1.py

```
1 pw = input("請輸入密碼：")
2 if pw=="1234":
3     print("歡迎光臨！")
```

程式說明

- 2-3　預設密碼為「1234」，若輸入的密碼正確就執行第 3 列程式列印訊息，若輸入的密碼錯誤就結束程式。

因為此處 if 程式區塊的程式碼只有一列，所以第 2-3 列可改寫為：

```
if pw=="1234" : print("歡迎光臨！")
```

2.3.2 雙向判斷式（if…else）

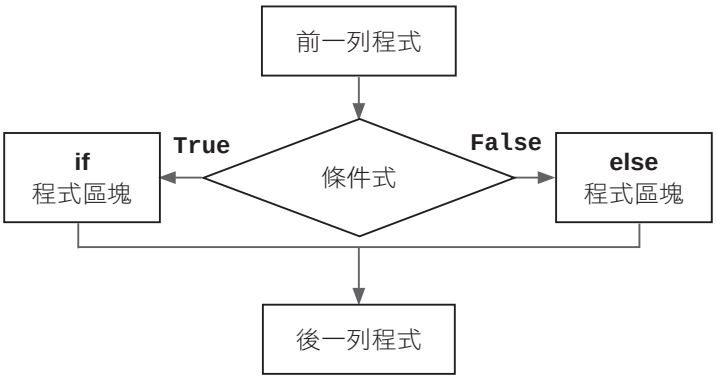
感覺上「if」語法並不完整，因為如果條件式成立就執行程式區塊內的內容，如果條件式不成立也應該做某些事來告知使用者。例如密碼驗證時，若密碼錯誤應顯示訊息告知使用者，此時就可使用「if…else…」雙向判斷式。

「if…else…」為雙向判斷式，語法為：

```
if 條件式:
    程式區塊一
else:
    程式區塊二
```

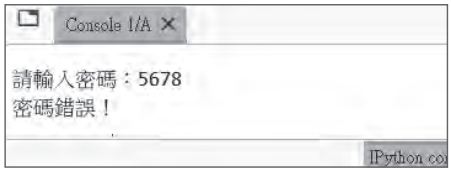
當條件式為 True 時，會執行 if 後的程式區塊一；當條件式為 False 時，會執行 else 後的程式區塊二，程式區塊中可以是一列或多列程式碼，如果程式區塊中的程式碼只有一列，可以合併為一列。

以下是雙向選擇流程控制的流程圖：



範例：進階密碼判斷

讓使用者輸入密碼，如果輸入的密碼正確 (1234)，會顯示「歡迎光臨！」；如果輸入的密碼錯誤，則會顯示「密碼錯誤！」訊息。



13.2 OpenCV：臉部辨識登入系統

臉部偵測僅能偵測出臉部的位置，應用有限，而臉部辨識則可以辨識人臉是屬於什麼人，例如目前較高階的手機可使用人臉解鎖，就是臉部辨識的應用。OpenCV 不僅可進行臉部偵測，也可以執行臉部辨識。本節專題使用臉部辨識進行登入。

13.2.1 擷取攝影機影像

OpenCV 除了可以讀取、顯示靜態圖片，也可以載入及播放動態影片，還可以讀取內建或外接攝影機影像資訊。筆記型電腦通常都具有攝影功能 (cam)，OpenCV 以 VideoCapture 啟動攝影機，語法為：

```
攝影機變數 = cv2.VideoCapture(n)
```

n 為整數，內建攝影機為 0，若還有其他攝影機則依次為 1、2、…等。例如開啟內建攝影機並存於 cap 變數：

```
cap = cv2.VideoCapture(0)
```

攝影機是否處於開啟狀態可由 isOpened 方法判斷，語法為：

```
攝影機變數.isOpened()
```

攝影機開啟會傳回 True，攝影機關閉則傳回 False。

攝影機若開啟可用 read 方法讀取攝影機影像，語法為：

```
布林值變數, 影像變數 = 攝影機變數.read()
```

- **布林值變數**：True 表示讀取影像成功，False 表示讀取影像失敗。
- **影像變數**：若讀取影像成功則將影像存於此變數中。

例如讀取攝影機影像，布林值存於 ret 變數，影像存於 img 中：

```
ret, img = cap.read()
```

最後要以 release 方法關閉攝影機並釋放資源：

```
攝影機變數.release()
```

取得使用者按鍵

攝影機為動態攝影，要如何取得特定時間的靜態相片呢？可讓使用者按特定鍵，程式就擷取按鍵時的靜態相片。OpenCV 的 waitKey 方法會等待使用者按鍵，此方法同時可取得按鍵的 ASCII 碼，語法為：

```
按鍵變數 = cv2.waitKey(n)
```

按鍵變數儲存按鍵的 ASCII 碼，這是一個 0 到 255 的數值，例如「A」的 ASCII 碼為 65。下面為設定使用者 10 秒內需按鍵，並將傳回的 ASCII 碼存於 key 變數中：

```
key = cv2.waitKey(10000)
```

若使用者按「A」鍵，則 key 變數的值為 65。

Python 的 ord 函式可取得字元的 ASCII 碼，以按鍵變數與字元的 ASCII 碼做比對，就可確認使用者是否按了特定鍵，例如：

```
if key == ord('A')
```

若結果是 True 表示使用者按了「A」鍵，False 表示使用者按了其他鍵。

範例：擷取攝影機靜態影像

程式執行後會自動開啟攝影機，使用者按「z」鍵就會擷取影像存檔。

程式碼：camPicture1.py

```
1 import cv2
2 cv2.namedWindow("frame")
3 cap = cv2.VideoCapture(0)
4 while(cap.isOpened()):
5     ret, img = cap.read()
6     if ret == True:
```

```
 7         cv2.imshow("frame", img)
 8         k = cv2.waitKey(100)
 9         if k == ord("z") or k == ord("Z"):
10             cv2.imwrite("media\\catch.jpg", img)
11             break
12 cap.release()
13 cv2.destroyWindow("frame")
```

程式說明

- 3　開啟內建攝影機。
- 4　只要攝影機為開啟狀態就執行此無窮迴圈：通常等待按鍵需以無窮迴圈檢查使用者是否按鍵。
- 5　讀取影像。
- 6-7　如果讀取成功就在視窗顯示。
- 8　每隔 0.1 秒檢查一次是否按鍵。
- 9　使用者可能按大寫或小寫「z」鍵，所以兩者都要檢查。
- 10-11　存檔後離開無窮迴圈。
- 12　關閉攝影機。

13.2.2 實戰：建立會員臉部模型

臉部偵測只能找到人臉的位置，而臉部辨識則可以進一步指出人臉是屬於誰的臉部。我們以 OpenCV 的臉部辨識功能建立一個臉部辨識登入系統。

OpenCV 臉部辨識功能是先以個人的若干張相片進行訓練建立模型，再以此模型來辨識未知臉部圖片是屬於何者。建立 OpenCV 臉部辨識模型需使用 opencv-contrib-python 模組，請以下列語法安裝：

```
pip install opencv-contrib-python --user
```

OpenCV 提供了三種辨識用的演算法：

- Eigenfaces
- Fisherfaces
- Local Binary Pattern Histogram (LBPH)

經實測，LBPH 的訓練速度最快、辨識效果最好，本專題使用 LBPH 演算法。

首先建立模型物件，語法為：

```
模型變數 = cv2.face.LBPHFaceRecognizer_create()
```

接著以模型物件的 train 方法進行訓練，語法為：

```
模型變數.train( 圖片串列 , 標記串列 )
```

「圖片串列」為圖片的 numpy 陣列，「標記串列」為圖片索引 (會員編號，整數) 的 numpy 陣列，即「0、1、2、……」的 numpy 陣列。

例如模型變數為 model，圖片存於 images 串列，圖片索引存於 labels 串列，進行訓練的程式碼為：

```
model = cv2.face.LBPHFaceRecognizer_create()
model.train(np.asarray(images), np.asarray(labels))
```

訓練完成後可用 save 方法將模型儲存起來，以便在臉部辨識中使用，例如：

```
model.save('faces_LBPH.yml')
```

蒐集個人臉部圖片

建立個人臉部模型時需要若干張個人臉部圖片，圖片越多辨識效果越好，但模型檔案越大。要如何取得個人臉部圖片呢？最簡單的方法是以攝影機擷取相片，可以在極短時間內就蒐集到大量的個人臉部圖片。

本專題以會員姓名做為資料夾名稱，以攝影機擷取 100 張圖片存於 <images> 資料夾的會員姓名資料夾中。

程式碼：**create_data.py**

```
 1 import cv2
 2 import os, glob
 3 import numpy as np
 4 from time import sleep
 5
 6 def saveImg(image, index):  # 圖片存檔
 7     filename = 'images/' + name + '/face{:03d}.jpg'.format(index)
 8     cv2.imwrite(filename, image)
 9
10 index = 1
11 total = 100  # 人臉取樣總數
```

```
12
13 name = input('輸入姓名 (使用英文)：')
14 if os.path.isdir('images/' + name):  # 使用姓名做為資料夾名稱
15     print('此姓名已存在！')
16 else:
17     os.mkdir('images/' + name)  # 建立資料夾
18     face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
         "haarcascade_frontalface_alt2.xml")
19     cap = cv2.VideoCapture(0)  # 開啟攝影機
20     cv2.namedWindow('video', cv2.WINDOW_NORMAL)
21     while index > 0:  # 取樣
22         ret, frame = cap.read()
23         frame = cv2.flip(frame, 1)
24         gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
25         faces = face_cascade.detectMultiScale(gray, 1.1, 3)
26         for (x, y, w, h) in faces:
27             frame = cv2.rectangle(frame, (x, y), (x + w, y + h),
                 (0, 255, 0), 3)  # 框選臉部
28             image = cv2.resize(gray[y: y + h, x: x + w],
                 (400, 400))  # 存檔圖片大小
29             saveImg(image, index)
30             sleep(0.1)
31             index += 1
32             if index > total:
33                 print('取樣完成！')
34                 index = -1  # 離開迴圈
35                 break
36         cv2.imshow('video', frame)
37         cv2.waitKey(1)
38     cap.release()  # 關閉 cam
39     cv2.destroyAllWindows()
```

程式說明

- 6-8　儲存圖片的自訂函式。
- 6　image 參數為圖片，index 參數為檔名流水號。
- 7　設定圖片檔案名稱：「face{:03d}.jpg」將流水號設為 3 個數字，例如 index 為 5 則檔案名稱為「face005.jpg」。
- 8　儲存檔案。
- 10-11　index 為圖片檔名流水號，total 為圖片總數。
- 13　讓使用者輸入姓名。

16.2 Pygame 動畫處理

動畫是遊戲不可或缺的要素，在遊戲中，只有角色動起來，遊戲才會擁有生命，但動畫也是遊戲設計者最感頭痛的部分。Pygame 套件透過不斷重新繪製繪圖視窗，短短幾列程式就讓圖片動起來了！

16.2.1 動畫處理基本程式架構

Pygame 基本程式架構中，最後是以無窮迴圈檢查使用者是否按  鈕關閉繪圖視窗，設計者可將不斷重新繪製繪圖視窗的程式碼置於此無窮迴圈內。動畫處理的基本程式架構為：(<basicmotion.py>)

```
……略
 7 background.fill((255,255,255))
 8
 9 clock = pygame.time.Clock()  # 建立時間元件
10 running = True
11 while running:
12     clock.tick(30)  # 每秒執行 30 次
13     for event in pygame.event.get():
14         if event.type == pygame.QUIT:
15             running = False
16     screen.blit(background, (0,0))  # 清除繪圖視窗
17
18     pygame.display.update()  # 更新繪圖視窗
19 pygame.quit()  # 關閉繪圖視窗
```

第 9 列建立 clock 元件，第 12 列利用此元件的 tick 方法設定每秒重繪次數，此處設為每秒重繪 30 次。重繪次數越多，動畫會越流暢，但 CPU 負擔越重，如果超過負荷，程式可能當機。如無特殊需求，一般設為「30」。

第 16 列以 background 背景畫布覆蓋繪圖視窗，會將繪圖視窗中所有內容清除，讓設計者重新繪製。

設計者可將一次性設計工作如建立幾何圖形、載入圖片、變數初值設定等程式碼置於第 8 列，再將移動後繪製圖片的程式碼置於第 17 列，相當於圖片一秒會移動 30 次，造成流暢的動畫效果。

16.2.2 角色類別 (Sprite)

Pygame 遊戲中有許多元件會重複使用，例如射擊太空船遊戲，外星太空船可能多達數十艘，只要建立「角色類別」，即可創造多個相同物件。

Pygame 角色類別是最被遊戲設計者稱道的功能，它不但能複製多個物件，還能進行動畫繪製、碰撞偵測等。建立角色類別的基本語法為：

```
class 角色名稱(pygame.sprite.Sprite):
    屬性1 = 值1
    屬性2 = 值2
    ..........
    def __init__(self, 參數1, 參數2, ……):
        pygame.sprite.Sprite.__init__(self)
        程式碼
```

屬性 1、屬性 2、……可有可無，通常做為類別中不同函式的共用變數。

「__init__」為類別建構式，類別中一定要有此函式，建立物件時會執行此函式，僅執行一次，通常用於建立圖形、載入圖片、初始化變數值等。

類別中可自行撰寫特定功能函式，例如繪製動畫功能，再於物件中呼叫。

以角色類別建立的角色物件無法直接在畫布中顯示，必須加入角色群組才能繪製。建立角色群組的語法為：

```
角色群組名稱 = pygame.sprite.Group()
```

使用角色群組的 add 方法可將角色物件加入角色群組，語法為：

```
角色群組名稱.add(角色物件)
```

角色群組中可包含多個角色物件。最後使用角色群組的 draw 方法可將群組內全部角色物件繪製到畫布上，語法為：

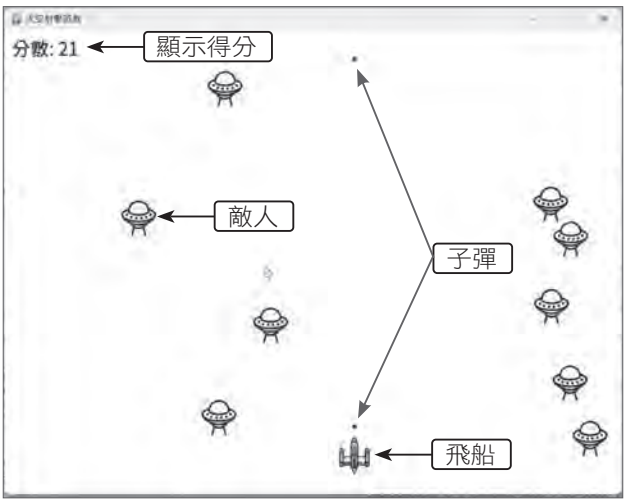
```
角色群組名稱.draw(畫布)
```

16.3 實戰：ChatGPT 製作太空射擊遊戲

ChatGPT 是目前最夯的大型語言模型，可以應用在非常多場合，使用 ChatGPT 產生程式碼也是重要應用之一。這個實戰應用是以自然語言讓 ChatGPT 為我們撰寫 Pygame 模組的遊戲程式，實作一個太空射擊遊戲。

16.3.1 應用程式總覽

遊戲畫面是飛船位於畫面下方，使用者可用滑鼠移動飛船，飛船只能左右移動。敵人會隨機由上方產生，然後敵人將不斷下降，使用者按滑鼠左鍵可發射子彈，當子彈碰到敵人就會消滅敵人，每消滅一個敵人可得到一分。當敵人碰到飛船時，遊戲就結束。<gametest7.py>



16.3.2 初步撰寫太空射擊遊戲程式

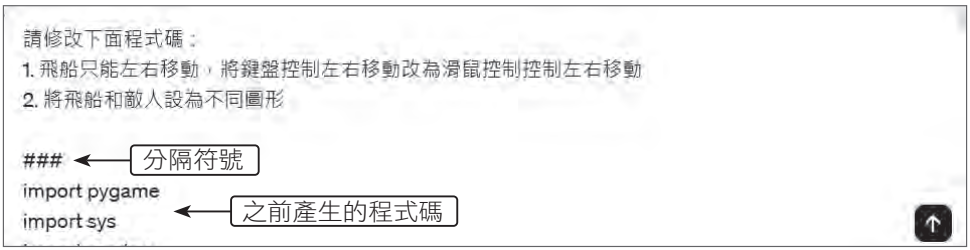
ChatGPT 產生程式觀念釐清

ChatGPT 的功能非常強大，不但可以使用自然語言指揮 ChatGPT 撰寫大部分語言的程式，也能讓 ChatGPT 為程式除錯，為什麼本章前兩個小節還要學習 Pygame 的程式語法呢？

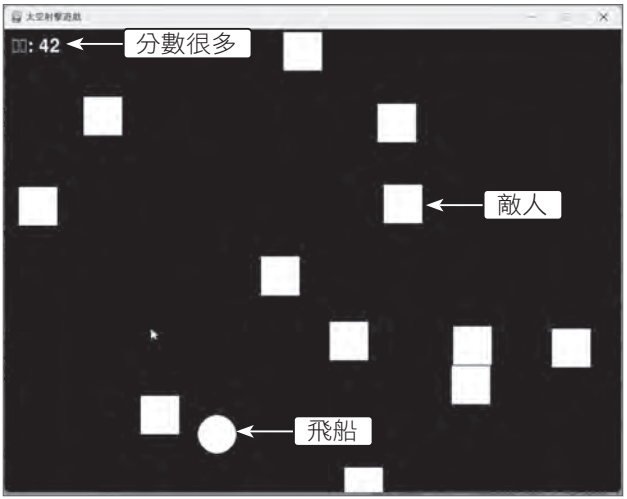
修改控制方式及圖形

觀察程式碼 67~78 列，可看到程式是以鍵盤的左、右鍵來控制飛船移動，非常不方便，應改用滑鼠操作才會順暢。另外，飛船和敵人的形狀完全相同，根本分不清是飛船還是敵人。

依照下圖對話修正 (所有對話儲存於本章範例 <prompt.txt> 檔)。雖然 ChatGPT 會記住之前對話的內容，但為了確保 ChatGPT 根據之前程式碼繼續修正，最好將程式碼傳送給 ChatGPT，而以對話和資料之間以「###」分隔，可達到最佳效果。



將程式碼儲存成 <gametest2.py> 檔，執行結果為：

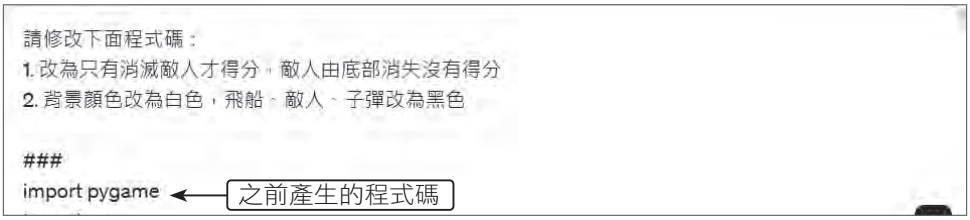


修改得分方式及顏色

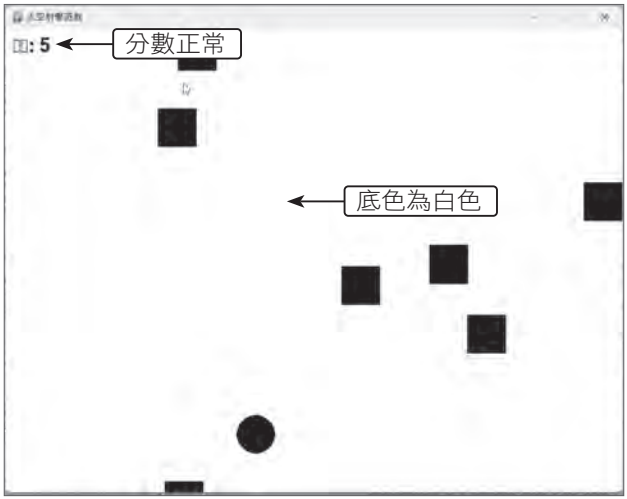
飛船的圖形已修改為圓形，飛船的控制也已改為滑鼠控制。

操作時只消滅了幾個敵人，左上角的得分高達 42 分。觀察程式碼 81~83 列，原來是當敵人消失在螢幕下方時也會得分，應修正為只有子彈消滅敵人時才得分。另外，因為最後使用的飛船圖形 (ship.jpg) 及敵人圖形 (enermy.jpg) 的背景色都是白色，因此需將遊戲畫面的背景顏色更改為白色。

依照下圖對話修正。



將程式碼儲存成 <gametest3.py> 檔，執行結果為：



以角色類別方式呈現

得分方式已修改為只有子彈消滅敵人時才得分，遊戲畫面的背景顏色已更改為白色。

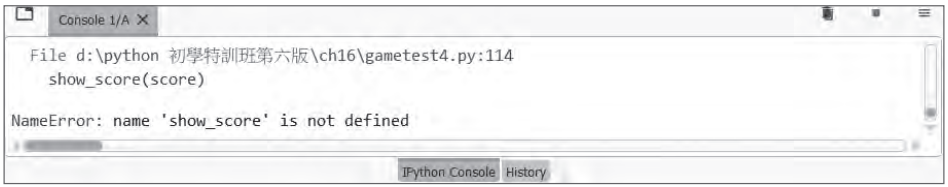
觀察程式碼 68~72 及 112~115 列，程式是將新產生的子彈及敵人加入串列，這樣的方式並不符合 Pygame 設計的方式，且效率較差，應使用角色類別方式建構飛船、敵人、子彈角色。

依照下圖對話修正。

將程式碼儲存成 <gametest4.py> 檔。

16.3.3 程式除錯及顯示中文

<gametest4.py> 檔執行時產生錯誤：

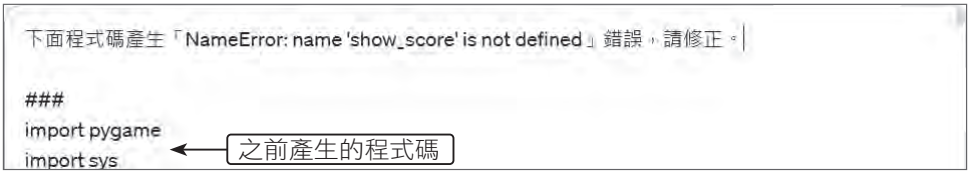


錯誤訊息顯示第 114 列程式錯誤，錯誤原因為 show_score 函式沒有定義。

程式除錯

將執行時產生的錯誤訊息傳送給 ChatGPT 進行修正。

依照下圖對話修正。



將程式碼儲存成 <gametest5.py> 檔。

執行結果與 <gametest3.py> 相同。

角色使用圖形檔案

目前飛船和敵人的圖形是 Pygame 畫的圓形及方形，並不美觀，本章範例檔案已放置了飛船圖形檔案 (ship.jpg) 及敵人圖形檔案 (enermy.jpg)，兩個檔案圖形的尺寸皆為 48X48，應使用這兩個檔案圖形取代 Pygame 畫的圓形及方形。

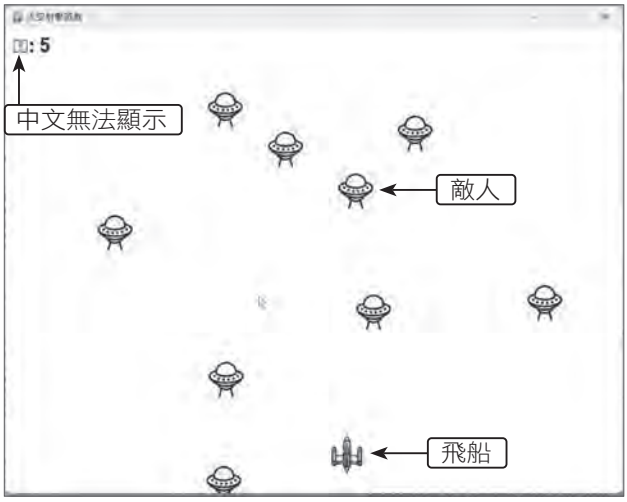
依照下圖對話修正。

```
請修改下面程式碼：
將飛船圖形改為 ship.jpg 檔案，敵人圖形改為 enermy.jpg 檔案，尺寸皆為 48x48

###
import pygame
import sys
```

之前產生的程式碼

將程式碼儲存成 <gametest6.py> 檔，執行結果為：



顯示中文

最後，<gametest6.py> 的執行結果畫面「分數：」無法顯示中文，嘗試了多次以自然語言指示 ChatGPT 修正都沒有成功，所以使用手動修正程式碼方式完成。<gametest7.py>

將 19 列程式「 font = pygame.font.Font(None, 36) 」修改為：

```
font = pygame.font.Font("TaipeiSansTCBeta-Regular.ttf", 24)
```

16.3.4 太空射擊遊戲完整程式碼

程式碼：**gametest7.py**

```
import pygame
import sys
import random

# 初始化 Pygame
pygame.init()

# 設定遊戲視窗大小
width, height = 800, 600
screen = pygame.display.set_mode((width, height))
pygame.display.set_caption(" 太空射擊遊戲 ")

# 定義顏色
black = (0, 0, 0)
white = (255, 255, 255)

# 設定分數
score = 0
font = pygame.font.Font("TaipeiSansTCBeta-Regular.ttf", 24)

class Player(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        self.image = pygame.image.load("ship.jpg").convert()
            # 更改為 ship.jpg
        self.image = pygame.transform.scale(self.image, (48, 48))
            # 調整尺寸為 48x48
        self.rect = self.image.get_rect()
        self.rect.center = (width // 2, height - 2 * 25)
        self.speed = 5

    def update(self):
        self.rect.x, _ = pygame.mouse.get_pos()
        self.rect.x -= self.rect.width // 2

class Enemy(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
```