專題：打造你的藝術傑作

by 田弘華 **Hung-Hua Tien**

重點複習1：迴圈

電腦很擅長重複做任何事。如果要重複固定次數，我們會用for迴圈。在有for的描述事後面的程式碼要縮排，表示這是要重複的區塊。

- 本例中如果要重複4次，就用for i in range(4)。range(4)裡的4是停止值；根據預設，起始值是0，因此i會取0,1,2和3這些值。
- **可以變更起始值**。如range(1,5)會從1開始數到5，取1,2,3,4這些值。
- **可以變更增加的步數**。例如for i in range(1,10,2)會從1開始數到9且每隔2個，因此會取1,3,5,7,9這些值。

～**巢狀迴圈**：我們要重複的東西本身也在重複。

～如何迫使程式碼停止執行

如果程式碼在執行中，而你需要立即叫它停止，可以把滑鼠移動到Python殼層，並且輸入 **Control + C** 來中斷程式。

- 如果發現海龜程式犯了錯，而不想讓它畫完，就能用這招。
- 建議一個無限迴圈後，程式永遠不會停止時，也可以用這招。

重點複習2：陣列清單

～電腦可以把一套項目儲存在清單(list)。

- 清單是內含多個項目的一種特殊變數，讓我們可以一次存取一個。
- 把清單想像成一個倉庫，裡面有一系列的箱子和櫃子。

～ 在Python建立清單

- 清單是放在方括號裡面。
- 每個項目是以逗號分隔。
- 清單項目是從0開始編號。
- 第i個元素索引是用變數[i]。

1. 海龜圖

海龜圖是學習Python和使用程式碼創作藝術作品的有趣方式，我們將透過輸入指令用虛擬海龜游標在螢幕上畫圖。

1-1 如何在Python使用海龜

- 使用海龜模組

  要開始使用turtle模組，我們首先需要導入它。方法很簡單，直接輸入import，後面則接著我們想要使用的模組。也就是，開頭輸入`import turtle`程式碼，表示要啟動海龜模組、使用海龜圖。你可以將導入視為告訴電腦抓取特定指令的方法，在我們繼續輸入程式碼之前，請電腦將它們準備好。

- 創造一個海龜

一旦導入了turtle模組，在螢幕上仍然看不到任何東西，但在幕後電腦已經準備好了，我們可以開始存取turtle模組不同片段的程式碼。現在要做的事情是先建立一個海龜，並且指派海龜變數名稱。

```
- 如shelly = turtle.Turtle()
- 改變海龜形狀，輸入shelly.shape("turtle")
    - 可以用arrow, circle, classic, square, triangle
- 找出海龜的位置
    - shelly.position()
```

你可以看到一個新視窗套出，海龜就在哪裡。你可以控制這個視窗，並用下面的指令讓海龜游標來畫圖。

~補充：點記法(Dot Notation)

一種顯示某些程式碼區塊彼此相關的方法。例如要告訴電腦，我們想使用專門屬於turtle模組中的Screen物件，就是在它們之間使用點(.)，即`turtle.Screen()`。如果要進一步使用屬於Screen物件的特定函式，例如更改顏色，就在Screen物件和要使用的函式間放一個點（和參數）。如:`turtle.Screen().bgcolor("blue")`，就表示：「請電腦找到turtle模組的Screen物件；執行此動作時，請找到屬於它的bgcolor()函式；最後，按照bgcolor()函式所說，使用我們給它的顏色blue去執行程式。」。記住，我們並沒有為此編寫程式碼，都是已經寫好放在turtle模組裡了，只需要在使用turtle模組之前先導入它即可。電腦會依據我們的要求，找到對應的物件和函式，並且執行已經寫好的程式碼。

~補充：有括弧或沒有括弧

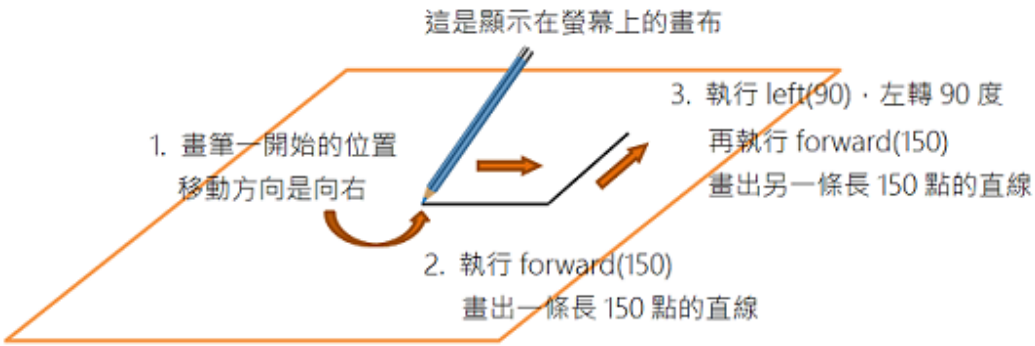
在物件導向的程式設計中，我們必須創立一個實體或副本，方便對物件和函式進行修改，所以有括號。如果是模組，我們不會修改只會直接使用，所以沒有括號。例如，turtle沒有括號，Screen、bgcolor有括號。

~補充：顏色

在電腦中，所有的顏色都是由加色原色法(Additive Primary Colors)產生的。原色是指紅色、綠色和藍色，電腦使用加色法，透過添加不同層級的紅色、綠色和藍色來創立顏色。由於電腦螢幕會發出光線，並且只能結合光線來製作顏色。當我們在電腦上選擇顏色時，需要告訴它要抓取多少原色來產生我們想要的顏色，即RGB色彩模型。RGB色彩模型代表紅綠藍模型，是使用三個數字編寫，每個數字表示使用多少的紅色、綠色和藍色。數字最小是0，最大是255。因為在八位元的二進位中，數字0等於00000000，數字255等於11111111。(如果改為十六進位系統十六，則是用0~9A~F表示，如藍色是`#1DA2DB`。)

1-2 常用指令整理

參考網址



(1)運動命令:

- forward(d):向前移動距離d代表距離
- backward(d):向後移動距離d代表距離
- right(degree):向右轉動多少度
- left(degree):向左轉動多少度
- position():位置
- goto(x,y):將畫筆移動到座標為(x,y)的位置
- speed(speed):畫筆繪製的速度範圍[0,10]整數
- stamp():繪製當前圖形

(2)畫筆控制命令:

- penup():畫筆抬起，移動時不繪製圖形
- pendown():畫筆落下，移動時繪製圖形
- pensize(width):畫筆的寬度
- pencolor(colorstring):畫筆的顏色，white,yellow, blue, black, gold, pink, brown, purple, red, gold, seashell. tomato...
- fillcolor(colorstring):繪製圖形的填充顏色。
- color(colorstring):繪製圖形的填充顏色。例：turtle.color(color1, color2)，同時設定pencolor=color1, fillcolor=color2。
- circle(radius, extent):繪製一個圓形，其中radius為半徑，extent為度數，例如若extent為180，則畫一個半圓；畫一個圓形，可不必寫第二個參數
- shape():海龜形狀，turtle, arrow, circle, classic, square, triangle
- setheading(degree):海龜朝向，degree代表角度
- reset():恢復所有設置

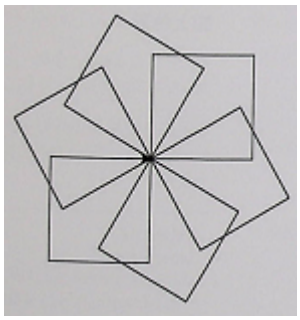
```
# 啟動模組
import turtle  #導入海龜模組
shelly = turtle.Turtle()  # 把模組中的螢幕游標叫出來
shelly.shape("turtle") # 將三角形改為烏龜
shelly.position() # 烏龜的位置

#海龜的外觀
## 海龜的顏色
shelly.color("green", "red") # 綠色
## 海龜外框的顏色
shelly.pencolor("green") # 綠色
## 把海龜變大
shelly.turtlesize(10,10,2) # (上下長度,左右寬度,外框粗細)
## 還原海龜大小
shelly.resizemode("auto") # auto表預設值
shelly.turtlesize(3,3,2)

# 海龜運動命令
## 畫一條線長100
shelly.forward(100)
## 向左轉90度
shelly.left(90)
## 提起筆離開畫布
shelly.penup()
## 放下筆貼近畫布
shelly.pendown()
## 隱藏使用的turtle
shelly.hideturtle()

#畫筆控制
## 要填滿繪製的圖案
shelly.begin_fill()
## 用什麼顏色
shelly.fillcolor("orange")
## 繪製圖形
shelly.circle(50) #半徑50的圓
shelly.circle(50, 180, 30) #半徑50，範圍180度，轉動方向30度
## 停止填滿顏色
shelly.end_fill()
## 蓋章
shelly.stamp()
## 在螢幕寫字
shelly.write("Turtle Rock")
```

例1：尺寸為100的正方形 - for迴圈

```
# 虛擬程式碼
- 向前移動100步
- 向左轉90度

- 向前移動100步
- 向左轉90度

- 向前移動100步
- 向左轉90度

- 向前移動100步
- 向左轉90度
```

```
# 方法（一）：for 迴圈 （請在VS Code中跑）
import turtle
shelly = turtle.Turtle()

for i in range(4):       # 畫正方形 - 1 + 2 做四次 [0,1,2,3]
    shelly.forward(100) # 1.畫直線
    shelly.left(90)      # 2.轉向
    print (i)            # 計數，一共四次

# 方法（二）：for 迴圈 + 色彩（請在VS Code中跑）
import turtle
shelly = turtle.Turtle()
shelly.color("red")  # 顏色 - 線用紅色

for i in range(4):   # 形狀 - 正方形
    shelly.forward(100)
    shelly.left(90)
```

例2：尺寸為100的正方形 - 巢狀迴圈



```
# 虛擬碼
以下重複六次:
    以下重複四次:
        前進100步
        左轉90度
    右轉60度
```

```
# 方法（一）：巢狀迴圈
# 外迴圈重複正方形6次
for n in range(6):
    #內迴圈重複4次來畫正方形
    for i in range(4):
        shelly.forward(100)
        shelly.left(90)
    shelly.right(60) #畫下一個正方形前轉彎

# 方法（二）：巢狀迴圈 ＋ 加上色彩
import turtle
shelly = turtle.Turtle()
colors = ["red","green","blue","black","purple","yellow"]

# 外迴圈重複正方形6次
for n in range(6):
    shelly.color(colors[n])
    #內迴圈重複4次來畫正方形
    for i in range(4):
        shelly.forward(100)
        shelly.left(90)
    shelly.right(60) #畫下一個正方形前轉彎

# 方法（三） ：巢狀迴圈 ＋填滿色彩
import turtle
shelly = turtle.Turtle()
colors = ["red","green","blue","black","purple","yellow"]

# 外迴圈重複正方形6次
for n in range(6):
    shelly.color(colors[n]) #
    shelly.begin_fill()
    shelly.fillcolor(colors[n])
    #內迴圈重複4次來畫正方形
    for i in range(4):
        shelly.forward(100)
        shelly.left(90)
    shelly.right(60) #畫下一個正方形前轉彎
    shelly.end_fill()
```

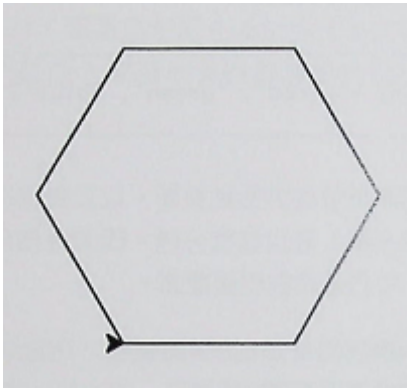
2. 專題：創造幾何藝術

Step 1: 畫出六角型

在任何海龜專題的開頭都要先匯入海龜模組，這樣才可以使用裡面的函數。

- 建立一個名為Art.py的檔案，輸入以下程式碼並執行，確認出現海龜。
- 在頂端加入一行註解來提醒自己這個專題是什麼。
- 修改畫正方形的虛擬程式碼，變成六角形。
    - 數量是6

- 轉彎角度是60



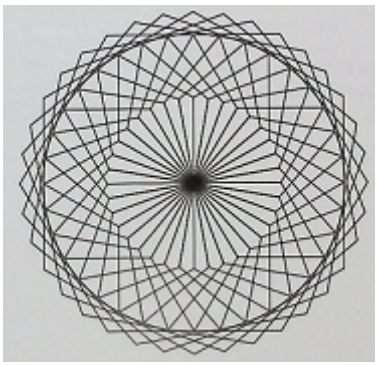
```
# 虛擬碼
以下重複六次
    前進100步
    左轉60度
```

```
# 畫出幾何圖案（一）：六角型
import turtle
shelly = turtle.Turtle()
# 重複6次：前進並轉彎
for i in range(6):
    shelly.forward(100)
    shelly.left(60)
```

Step 2: 用巢狀迴圈重複六角形

用迴圈畫出六角形之後，就能把此六角形放在另一個重複的迴圈裡，畫出一個圓的多個六角形，各個稍微重疊。

- 每個六角形相對於前一個六角形只轉10度。
- 畫一個圓，一共需要360/10=36個六角形



```
# 虛擬碼
以下重複36次
    以下重複6次
        前進100步
        左轉60度
    右轉10度
```

```
# 畫出幾何圖案（二）：重複六角型
import turtle
shelly = turtle.Turtle()
for n in range(36):
# 重複6次：前進並轉彎
    for i in range(6):
        shelly.forward(100)
        shelly.left(60)
    shelly.right(10) #加入轉彎
```

Step 3: 變更背景：加入彩虹顏色

我們可以加入色彩和背景讓這個圖更有趣。

- 背景變黑
    - turtle.bgcolor("black")
- 建立顏色清單，然後用迴圈取顏色。
    - colors = ["red", "yellow", "blue", "orange", "green", "red"]
    - shelly.color(colors[i])

```
## 畫出幾何圖案（三）：彩虹重複六角型
import turtle
shelly = turtle.Turtle()

turtle.bgcolor("black") #把背景變黑

# 畫出36個六角形，各隔10度
for n in range(36):
# 重複6次來畫六角形
    colors = ["red", "yellow", "blue", "orange", "green", "purple"]
 # 選擇六角形顏色順序
    for i in range(6):
        shelly.color(colors[i]) #選擇i位置的顏色
        shelly.forward(100)
        shelly.left(60)
# 在畫下一個六角形前轉彎
    shelly.right(10)
```

Step 4: 在圖案周圍加上小白圓圈

在海龜圖案的邊緣，畫一個小圓圈。

- 因為有36個六角形，所以會有36個小白圓圈。
- 畫一個小白圓圈，然後退回原點，然後轉10度；再出去畫一個小白圓圈，反覆執行。

```
## 畫出幾何圖案（四）：小圓圈彩虹重複六角型
# 畫36個小圓圈
shelly.penup()
shelly.color("white")
# 重複36次，找到對應的六角形
```

```
for i in range(36):
    shelly.forward(220)
    shelly.pendown()
    shelly.circle(5)
    shelly.penup()
    shelly.backward(220)
    shelly.right(10)
# 隱藏海龜
shelly.hideturtle()
```

Summary

```
# 完整的英文程式碼
# make a geometric rainbow pattern
import turtle
# pick order of colors for the hexagon
colors = ['red', 'yellow', 'blue', 'orange', \
'green', 'red']
shelly = turtle.Turtle()
turtle.bgcolor('black') # turn background black
# make 36 hexagons, each 10 degrees apart
for n in range(36):
# make hexagon by repeating 6 times
    for i in range(6):
        shelly.color(colors[i]) # pick color at position i
        shelly.forward(100)
        shelly.left(60)
    # add a turn before the next hexagon
    shelly.right(10)

# get ready to draw 36 circles
shelly.penup()
shelly.color('white')
# repeat 36 times to match the 36 hexagons
for i in range(36):
    shelly.forward(220)
    shelly.pendown()
    shelly.circle(5)
    shelly.penup()
    shelly.backward(220)
    shelly.right(10)
# hide turtle to finish the drawing
shelly.hideturtle()
```

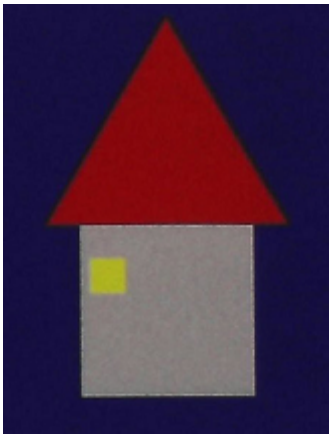
3. 熟能生巧

3-1 圖案1：畫出一排彩色正方形

```
# 虛擬碼
以下重複6次
    從清單設定顏色
    以下重複4次
        前進25
        左轉90
    提筆
    前進30
    下筆
隱藏海龜
```

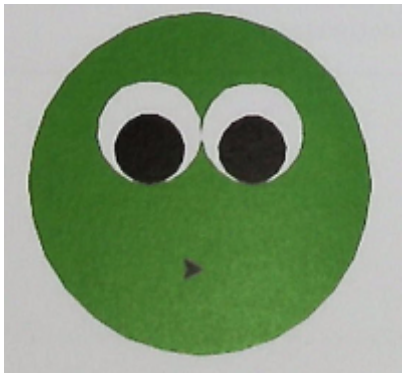
```
## 圖案1：畫出幾何彩虹圖案，畫出一排彩色正方形
# (1) 準備工作（色彩）
import turtle
shelly = turtle.Turtle()
turtle.bgcolor("black") #把背景變黑

# (2) 畫出6個正方形，各差30步
for n in range(6):     # 畫六個正方形
    colors = ["red", "green","blue", "gold", "purple",  "yellow"]
    shelly.color(colors[n]) # A. 選擇第n個正方形的顏色
    for i in range(4):       # B. 畫一個尺寸為25的正方形
        shelly.forward(25)
        shelly.left(90)
    shelly.penup()          # C. 畫下一個正方形，前進30步（25+5=30）
    shelly.forward(30)
    shelly.pendown()
# (3) 隱藏海龜
shelly.hideturtle()
```

3-2 圖案2：畫出房子



```
## 圖案2：畫出幾何彩虹圖案，畫出房子
import time
import turtle
turtle.bgcolor("blue") #把背景變藍
shelly = turtle.Turtle()

# (1)畫出房子的第一個大正方形
shelly.begin_fill() # 開始填入顏色
shelly.color("gray")
for i in range(4): # 重複4次來畫尺寸為100正方形
    shelly.forward(100)
    shelly.left(90)
shelly.end_fill() # 停止填入顏色
shelly.penup()
shelly.goto(-20,100) #將海龜一到下一個三角形的起點
shelly.pendown()
time.sleep(3)

# (2) 畫出房子的屋頂，一個大三角形
shelly.begin_fill()
shelly.color("red")
shelly.left(60)
shelly.forward(140)
shelly.right(120)
shelly.forward(140)
shelly.right(120)
shelly.forward(140)
shelly.end_fill()
time.sleep(3)

# (3)畫出窗戶
shelly.penup()
shelly.goto(25,80)
shelly.pendown()
shelly.begin_fill()
shelly.color("yellow")
for j in range(4):
    shelly.forward(20)
    shelly.left(90)
shelly.end_fill()
time.sleep(3)

# (4)隱藏海龜
shelly.hideturtle()
```

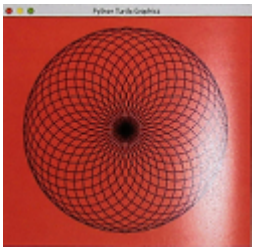
3-3 圖案 3 ：用圓圈畫出綠色臉



```
## 圖案3：畫出幾何彩虹圖案，畫出綠色臉
import time
import turtle
shelly = turtle.Turtle()

# (1)畫出背景綠色圓
shelly.begin_fill()
shelly.color("green")
shelly.circle(120)
shelly.end_fill()

# (2)畫出左右眼睛
shelly.penup()
shelly.goto(-30,100) #眼睛
shelly.pendown()
shelly.begin_fill() # 開始填入顏色
shelly.color("white")
shelly.circle(30)
shelly.end_fill()
shelly.begin_fill() # 眼球
shelly.color("black")
shelly.circle(20)
shelly.end_fill() # 停止填入顏色
time.sleep(3)

shelly.penup()
shelly.goto(30,100) #眼睛
shelly.pendown()
shelly.begin_fill() # 開始填入顏色
shelly.color("white")
shelly.circle(30)
shelly.end_fill()
shelly.begin_fill() # 眼球
shelly.color("black")
shelly.circle(20)
shelly.end_fill() # 停止填入顏色
time.sleep(3)

# (3)畫出嘴巴
shelly.penup()
shelly.goto(0,45) #嘴巴
shelly.pendown()
```

```
shelly.color("black")
shelly.shape("classic")
shelly.stamp()
time.sleep(3)

# (4)隱藏海龜
shelly.hideturtle()
```

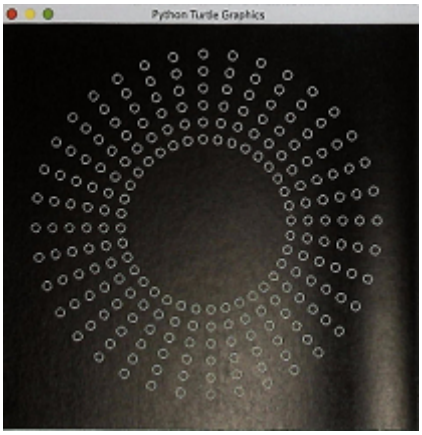
3-4 圖案4：重疊圓圈



```
# 虛擬碼
重複以下36次：
    畫出尺寸100的圓
    右轉10度
```

```
## 圖案4：畫出幾何彩虹圖案，重疊圓圈
import turtle
shelly = turtle.Turtle()
turtle.bgcolor("orange")
for n in range(36):
    shelly.circle(100)
    shelly.right(10) #加入轉彎
```

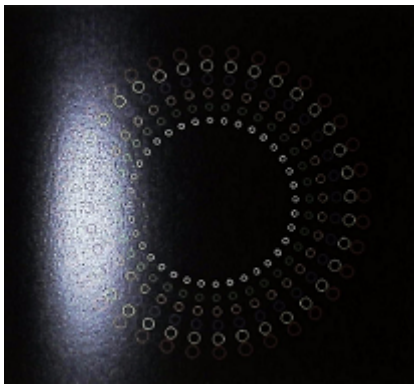
3-5 圖案5：圈中有圈

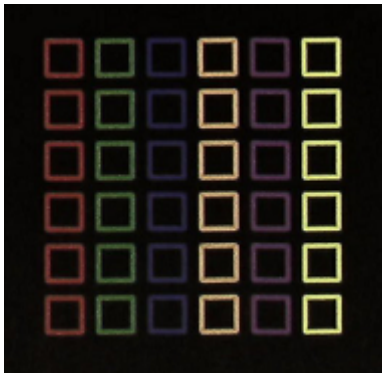


```
# 虛擬程式碼
重複以下36次
    提筆
    前進200
    重複六次
        下筆
        畫尺寸為5的圓
        提筆
        後退20
    返回中心點80
    右轉10度
隱藏海龜
```

```
# 圖案5：畫出幾何彩虹圖案，圈中有圈
import turtle
turtle.bgcolor("black")
shelly = turtle.Turtle()
shelly.color("white")
for n in range(36):
    shelly.penup()
    shelly.forward(200)
    for i in range(6):
        shelly.pendown()
        shelly.circle(5)
        shelly.penup()
        shelly.backward(20)
    shelly.backward(80) # 6*20 + 80 = 200
    shelly.right(10) #加入轉彎
shelly.hideturtle()
```

3-6 圖案6：加分題





～想想看有沒有其他可以試(Optional)

4. 補充練習

4-1 圖案: Turtle Rock

```
import turtle
import random
shelly = turtle.Turtle()
shelly.shape("turtle")

turtle.colormode(255)
paces = 20
random_red = 50
random_green = 50
random_blue = 50

shelly.penup()

for i in range(50):
    random_red = random.randint(0,255)
    random_green = random.randint(0,255)
    random_blue = random.randint(0,255)
    shelly.color(random_red, random_green, random_blue)
    shelly.stamp()
    paces += 3
    shelly.forward(paces)
    shelly.right(25)

    shelly.write("Turtle rock!")
```

4-2 圖案: Star畫星星

```
import turtle
turtle.colormode(255)
shelly = turtle.Turtle()
shelly.color(255, 215, 0)
shelly.pensize(5)

# 將筆向前移動100個單位，向右轉144度，做五次畫五角星。
for i in range(5):
    shelly.forward(100)
    shelly.right(144)
```

4-3 練習：圓中圓

```
import turtle
pen = turtle.Turtle()

pen.color('purple')
pen.begin_fill()
pen.circle(100)
pen.end_fill()

pen.color('blue')
pen.begin_fill()
pen.circle(50)
pen.end_fill()

pen.color('red')
pen.begin_fill()
pen.circle(20)
pen.end_fill()
```

4-4 一排連續三個正方形

```
# 程式碼（一）：一般函式
## 用海龜畫一個正方形
import turtle
shelly = turtle.Turtle()
for i in range(4):
    shelly.forward(100)
    shelly.left(90)

# 程式碼（二）：一般函式，讓海龜畫一個正方形
# 海龜總共畫一排連續三個正方形

## 我的函式
import turtle
shelly = turtle.Turtle()

# square函式產生尺寸為100的正方形
def square():
    for i in range(4):
```

```
        shelly.forward(100)
        shelly.left(90)

square() # 呼叫函數第一次，畫一個正方形
shelly.forward(100) #前進100
square() # 呼叫函式第二次，畫另一個正方形
shelly.forward(100) #前進100
square() # 呼叫函式第三次，再畫另一個正方形
```

```
# 程式碼（三）：有參數的函式，讓海龜畫一個正方形
# 海龜總共畫三個越來越大的正方形

## 我的函式，參數名稱為s
import turtle
shelly = turtle.Turtle()

# square函式能畫出任何尺寸的尺寸正方形
def square(s):
    for i in range(4):
        shelly.forward(s) # 正方形每邊為變數s
        shelly.left(90)

square(100) # 呼叫函數第一次，畫一個尺寸為100正方形
shelly.forward(100) #前進100
square(200) # 呼叫函式第二次，畫另一個尺寸為200正方形
shelly.forward(100) #前進100
square(300) # 呼叫函式第三次，再畫另一個尺寸為300正方形
```

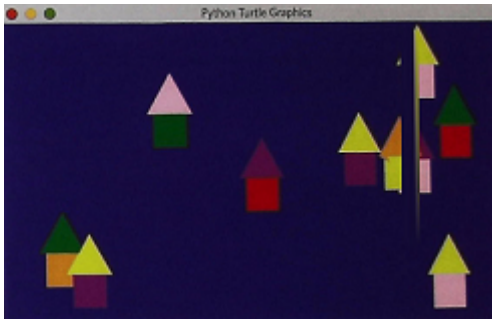
4-5 隨機實驗1:創造抽象藝術

用隨機顏色、隨機尺寸的圓圈和隨機尺寸的正方形，來創造每次執行程式都不同的抽象藝術。

```
# 虛擬程式碼
- 匯入turtle模組
- 匯入random模組
- 建立海龜
- 建立顏色清單
- 以下執行100次
    - 向前移動海龜0和360之間的隨機距離
    - 開始填入顏色
    - 設尺寸為10和50之間的隨機數
    - 以square函式和尺寸畫正方形
    - 結束填入正方形顏色
    - 向前移動海龜20和100之間的隨機距離
    - 以0和360之間的隨機角度旋轉海龜
    - 開始填入顏色
    - 設定隨機填入顏色
    - 以5和30之間的隨機數畫圓
    - 結束填入圓形顏色
```

4-6 隨機實驗2:創造有變化的鄉村風景

使用下面的house函式，在風景畫中創作各種不同顏色尺寸的隨機小房子，每次執行程式時都會有不同的結果。




```
# 虛擬程式碼
- 匯入turtle模組
- 匯入random模組
- 建立海龜
- 建立顏色清單
- 從下方複製函式程式碼
- 設背景為藍色
- 以下重複10次
    - 設x為-200到200之間的隨機數。
    - 設y為-200到200之間的隨機數。
    - 設wall_color為清單中的隨機顏色。
    - 設roof_color為清單中的隨機顏色。
    - 以x,y,wall_color,roof_color為參數呼叫house函式。
```

```
def house(x,y,wallColor, roofColor):
    shelly.penup()
    shelly.goto(x,y)
    shelly.setheading(0) # 讓海龜指向右邊
    shelly.pendown()
    shelly.color(wallColor)
    for i in range(4):
        shelly,right(90)
        shelly.forward(30)
    shelly.end_fill()
    shelly.backward(35) # 返回，準備好畫屋頂
    shelly.begin_fill()
    shelly.color(roofColor)
    shelly.left(60)
    shelly.forward(40)
    shelly.right(120)
    shelly.forward(40)
    shelly.right(120)
    shelly.forward(40)
    shelly.end_fill()
```

在海龜作圖中，我們可以編寫指令讓一個虛擬的（想象中的）海龜在螢幕上來回移動。這個海龜帶著一隻筆，我們可以讓海龜無論移動到哪都使用這隻筆來繪製線條。通過編寫程式碼，以各種很酷的模式移動海龜，我們可以繪製出令人驚奇的圖片。使用海龜作圖，我們不僅能夠只用幾行程式碼就創建出令人印象深刻的視覺效果，而且還可以跟隨海龜看看每行程式碼如何影響到它的移動。這能夠幫助我們理解程式碼的邏輯。所以海龜作圖也常被用作新手學習 Python 的一種方式。

4-7 紅與黃多個星星，烏龜太陽花

```
# turtleStar.py
import turtle
screen = turtle.Screen()
star = turtle.Turtle()
star.color('red', 'yellow')
star.begin_fill()
while True:
 star.forward(200)
 star.left(170)
 if abs(star.pos()) < 1:
    break;
star.end_fill()
screen.exitonclick()
```

4-8 三個三角形塔

```
# turtleSTriangle.py
# (Sierpinsky triangle algorithm)
import turtle
def striangle(myTurtle, depth, base):
    myTurtle.down()
    if depth == 0:
        myTurtle.begin_fill()
        for i in 0,1,2:
            myTurtle.forward(base)
            myTurtle.left(120)
            myTurtle.end_fill()
    else:
        for i in 0,1,2:
            striangle(myTurtle, depth-1, base)
            myTurtle.up()
             myTurtle.forward(base*2**depth)
             myTurtle.left(120)
            myTurtle.down()

screen = turtle.Screen()
triangle = turtle.Turtle()
triangle.speed(0)
triangle.reset()
striangle(triangle, 4, 10)
screen.exitonclick()
```

4-9 留下烏龜印記

```
#留下印記
## 烏龜在行進中可以留下印記(Stamp)
# turtleStamp.py
import turtle
screen = turtle.Screen()
screen.setup(600,600)
screen.bgcolor("lightgreen")
myStamp = turtle.Turtle(visible=False)
myStamp.shape("turtle")
myStamp.color("blue")
# myStamp.speed(8)
myStamp.penup() # Do not draw the path
stepLen = 20
for i in range(31):
 myStamp.stamp() # Leave an impression on the canvas
 stepLen = stepLen + 3 # Increase the step length on every
iteration
 myStamp.forward(stepLen) # Move along
 myStamp.right(24) # and turn
myStamp.penup() # Do not draw the path
myStamp.goto(0, 260) # Move
myStamp.color('red')
myStamp.write('Done!', align='center', font=('Arial', 20, 'bold'))
screen.exitonclick()
```

4-10 計算烏龜之間兩點的距離

```
# 利用 distance(x)方法指令可計算烏龜位置與 x 的距離，x 可為一個位置向量(x,
y)、或另一隻烏龜
# turtleDistance.py
import turtle
screen = turtle.Screen()
turtleA, turtleB = turtle.Turtle(), turtle.Turtle()
turtleB.goto(10, 20)
print(turtleA.distance((30,40)), ',', turtleA.distance(turtleB))
screen.exitonclick()
```

4-11 烏龜鐘錶

```
##### Applicaiton
# coding=utf-8

import turtle
from datetime import *

# 擡起畫筆，向前運動一段距離放下
def Skip(step):
    turtle.penup()
    turtle.forward(step)
```

```
    turtle.pendown()

def mkHand(name, length):
    # 註冊Turtle形狀，建立錶針Turtle
    turtle.reset()
    Skip(-length * 0.1)
    # 開始記錄多邊形的頂點。當前的烏龜位置是多邊形的第一個頂點。
    turtle.begin_poly()
    turtle.forward(length * 1.1)
    # 停止記錄多邊形的頂點。當前的烏龜位置是多邊形的最後一個頂點。將與第一個頂
點相連。
    turtle.end_poly()
    # 返回最後記錄的多邊形。
    handForm = turtle.get_poly()
    turtle.register_shape(name, handForm)

def Init():
    global secHand, minHand, hurHand, printer
    # 重置Turtle指向北
    turtle.mode("logo")
    # 建立三個錶針Turtle並初始化
    mkHand("secHand", 135)
    mkHand("minHand", 125)
    mkHand("hurHand", 90)
    secHand = turtle.Turtle()
    secHand.shape("secHand")
    minHand = turtle.Turtle()
    minHand.shape("minHand")
    hurHand = turtle.Turtle()
    hurHand.shape("hurHand")

    for hand in secHand, minHand, hurHand:
        hand.shapesize(1, 1, 3)
        hand.speed(0)

    # 建立輸出文字Turtle
    printer = turtle.Turtle()
    # 隱藏畫筆的turtle形狀
    printer.hideturtle()
    printer.penup()

def SetupClock(radius):
    # 建立表的外框
    turtle.reset()
    turtle.pensize(7)
    for i in range(60):
        Skip(radius)
        if i % 5 == 0:
            turtle.forward(20)
            Skip(-radius - 20)

            Skip(radius + 20)
```

```
            if i == 0:
                turtle.write(int(12), align="center", font=
("Courier", 14, "bold"))
            elif i == 30:
                Skip(25)
                turtle.write(int(i/5), align="center", font=
("Courier", 14, "bold"))
                Skip(-25)
            elif (i == 25 or i == 35):
                Skip(20)
                turtle.write(int(i/5), align="center", font=
("Courier", 14, "bold"))
                Skip(-20)
            else:
                turtle.write(int(i/5), align="center", font=
("Courier", 14, "bold"))
            Skip(-radius - 20)
        else:
            turtle.dot(5)
            Skip(-radius)
        turtle.right(6)

def Week(t):
    week = ["星期一", "星期二", "星期三",
            "星期四", "星期五", "星期六", "星期日"]
    return week[t.weekday()]

def Date(t):
    y = t.year
    m = t.month
    d = t.day
    return "%s %d%d" % (y, m, d)

def Tick():
    # 繪製錶針的動態顯示
    t = datetime.today()
    second = t.second + t.microsecond * 0.000001
    minute = t.minute + second / 60.0
    hour = t.hour + minute / 60.0
    secHand.setheading(6 * second)
    minHand.setheading(6 * minute)
    hurHand.setheading(30 * hour)

    turtle.tracer(False)
    printer.forward(65)
    printer.write(Week(t), align="center",
                  font=("Courier", 14, "bold"))
    printer.back(130)
    printer.write(Date(t), align="center",
                  font=("Courier", 14, "bold"))
    printer.home()
    turtle.tracer(True)
```

```
    # 100ms後繼續呼叫tick
    turtle.ontimer(Tick, 100)

def main():
    # 開啟/關閉龜動畫，併為更新圖紙設定延遲。
    turtle.tracer(False)
    Init()
    SetupClock(160)
    turtle.tracer(True)
    Tick()
    turtle.mainloop()

if __name__ == "__main__":
    main()
```

4-12 烏龜皮卡秋

```
# 皮卡秋

import turtle as t

def nose():
    t.penup()
    t.seth(90)
    t.fd(100)
    t.pendown()
    t.begin_fill()
    t.fillcolor("black")
    t.seth(45)
    t.fd(25)
    t.seth(135)
    t.circle(25,90)
    t.seth(315)
    t.fd(25)
    t.end_fill()

def eyes(seth,fd,c):
    t.penup()
    t.seth(seth)
    t.fd(fd)
    t.pendown()
    t.begin_fill()
    t.fillcolor('black')
    t.circle(50)
    t.end_fill()
    t.penup()
    t.circle(50,c)
    t.pendown()
    t.begin_fill()
    t.fillcolor('white')
```

```
    t.circle(20)
    t.end_fill()

def face(seth,fd):
    t.penup()
    t.seth(seth)
    t.fd(fd)
    t.pendown()
    t.begin_fill()
    t.fillcolor('red')
    t.circle(70)
    t.end_fill()

def lip():
    t.penup()
    t.seth(135)
    t.fd(250)
    t.pendown()
    t.seth(-300)
    t.circle(30,-65)
    t.begin_fill()
    t.fillcolor('Firebrick')
    t.seth(165)
    t.fd(140)
    t.seth(195)
    t.fd(140)
    t.seth(-360)
    t.circle(30,-65)
    t.penup()
    t.seth(-60)
    t.circle(30,65)
    t.pendown()
    t.seth(-70)
    t.fd(240)
    t.circle(55,140)
    t.seth(70)
    t.fd(240)
    t.end_fill()
    t.seth(-110)
    t.fd(80)
    t.begin_fill()
    t.fillcolor('Firebrick')
    t.seth(120)
    t.circle(120,123)
    t.seth(-70)
    t.fd(165)
    t.circle(55,140)
    t.seth(72)
    t.fd(165)
    t.end_fill()

def setting():
```

```
    t.pensize(4)
    t.hideturtle()
    t.setup(1000,600)
    t.speed(10)
    t.screensize(bg='yellow')

def main():
    setting()
    nose()
    eyes(160,250,60)
    eyes(-9.5,530,230)
    face(195,600)
    face(-11,720)
    lip()
    t.done()

if __name__ == '__main__':
    main()
```

4-13 用直線和圓畫風景畫

```
# 用直線和圓畫風景畫

"""
星星、樹葉都是randrange隨機產生的，因此每次執行程式，
星星、樹葉都長的不一樣。寫程式儘量用def定義函式庫。
主程式不要太大，程式結構才能調理分明，修改程式也容易。
本程式把房屋、月亮、樹木都def乘函式庫，
要重覆在不同位置，畫很多次就很方便。
"""

"""
# 主程式架構

def line(x1, x2, y2, color, size):
    畫直線
def tree(xa, ya):
    畫樹
def moon(xa, ya):
    畫月亮和雲
def house(xq, ya):
    用line函式畫房屋

for n in range(80):
    畫80個星星
moon(-100, 300) 月亮
tree(-500, 0) 樹
tree(-300, -40) 樹
tree(300, -70) 樹
house(100, 70)房屋
turtle.goto(0, -1580)
```

```
turtle.dot(2800, "green")綠地
"""

import turtle
import random

turtle.bgcolor("black")
turtle.speed(11)
turtle.up()
turtle.ht()

# 定義畫線的函式，要用來畫房子
def line(x1, y1, x2, y2, color, size):
    turtle.pencolor(color)
    turtle.pensize(size)
    turtle.goto(x1, y1)
    turtle.down()
    turtle.goto(x2, y2)
    turtle.up()

# 隨機30個綠色小圓組合成樹葉
def tree(xa, ya):
    line(xa + 170, ya + 60, xa + 170, ya - 160, 'brown', 30)
    for n in range(30):
        x = random.randrange(xa + 100, xa + 250)
        y = random.randrange(ya + 0, ya + 100)
        d = random.randrange(30, 60)
        turtle.goto(x, y)
        turtle.dot(d, "green")

# 畫月亮函式
def moon(xa, ya):
    turtle.goto(xa, ya)
    turtle.dot(90, "white")
    turtle.goto(xa - 30, ya + 30)
    turtle.dot(120, "black")

# 畫房屋函式
def house(xa, ya):
    line(xa, ya - 100, xa, ya - 300, "yellow", 150)
    line(xa, ya, xa - 100, ya - 100, "red", 50)
    line(xa, ya, xa + 100, ya - 100, "red", 50)

# 隨機畫80個星星
for n in range(80):
    x = random.randrange(-800, 800)
    y = random.randrange(0, 500)
    turtle.goto(x, y)
    turtle.dot(3, "white")

# 主程式
moon(300, 200)
```

```
tree(300, -90)
tree(-450, 0)
tree(-750, -60)
house(100, 70)
turtle.goto(0, -1580)
turtle.dot(3000, "green")
```