Python程式設計：海龜繪圖

**目錄**

[Python程式設計：海龜繪圖 1](#_Toc206000498)

[1. 建立畫布與海龜 1](#_Toc206000499)

[1.1 範例：畫正方形 2](#_Toc206000500)

[1.2 範例：畫五邊形 2](#_Toc206000501)

[2. 控制畫筆顏色與線條粗細 2](#_Toc206000502)

[2.1 範例：繪製一個由不同顏色和粗細線條組成的同心圓 2](#_Toc206000503)

[2.2 範例： 3](#_Toc206000504)

[3. 填滿顏色 4](#_Toc206000505)

[3.1 範例：繪製一個填滿顏色的五角形星星 4](#_Toc206000506)

[3.2 繪製圓弧形 5](#_Toc206000507)

[3.3 範例： 5](#_Toc206000508)

[4. 螢幕與海龜控制 6](#_Toc206000509)

[4.1 範例：繪製一個藍色天空下的五角形星星 6](#_Toc206000510)

[5. 停止追蹤 6](#_Toc206000511)

[5.1 範例： 7](#_Toc206000512)

[小專題：謝爾賓斯基三角形（Sierpinski Triangle） 8](#_Toc206000513)

[小專題：科赫雪花（Koch Snowflake） 9](#_Toc206000514)

海龜繪圖模組 (turtle) 是 Python 內建的，不需要額外安裝，可以直接使用 import turtle 來匯入。海龜繪圖的核心是「海龜」，它有三個主要特性：

* **位置與坐標**：以 (0, 0) 為中心，向右為 x 軸正向，向上為 y 軸正向 。
* **畫筆屬性**：可以設定畫筆的顏色、粗細、填充顏色等 。
* **方向與狀態**：海龜有方向性，可以前進、後退、轉彎，畫筆可以放下或抬起

# 建立畫布與海龜

t = turtle.Pen() 可以建立一個海龜，並自動生成一個畫布。

以下是海龜繪圖的一些基本指令：

|  |  |
| --- | --- |
| **指令** | **說明** |
| left(angle) | 向左轉動指定角度 |
| right(angle) | 向右轉動指定角度 |
| forward(number) | 向前移動指定距離 |
| backward(number) | 向後移動指定距離 |
| penup() | 抬起畫筆，移動時不繪圖 |
| pendown() | 放下畫筆，開始繪圖 |
| goto(x, y) | 移動到指定座標 (x, y) |
| speed(n) | 設定海龜速度，0 為最快 |

## 範例：畫正方形

import turtle

t = turtle.Pen()

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

t.forward(100)

t.left(90)

turtle.done() # 結束後停住

## 範例：畫五邊形

import turtle

t = turtle.Pen()

sides = 5

for x in range(sides):

    t.forward(100)

    t.right(360/sides)

turtle.done()  # 保持視窗開啟

# 控制畫筆顏色與線條粗細

|  |  |
| --- | --- |
| pencolor(color\_string) | 選擇畫筆顏色，例如 "red" 或 "green" |
| pencolor(r, g, b) | 使用 RGB 值設定顏色 |
| pensize(size) / width(size) | 設定畫筆粗細 |

## 範例：繪製一個由不同顏色和粗細線條組成的同心圓

import turtle

t = turtle.Pen()

colors = ['red', 'orange', 'yellow', 'green', 'blue', 'cyan', 'purple', 'violet']

t.width(1)# 初始畫筆寬度設為 1

t.speed(10)    # 設定畫筆速度為 10（接近最快）

for x in range(41):

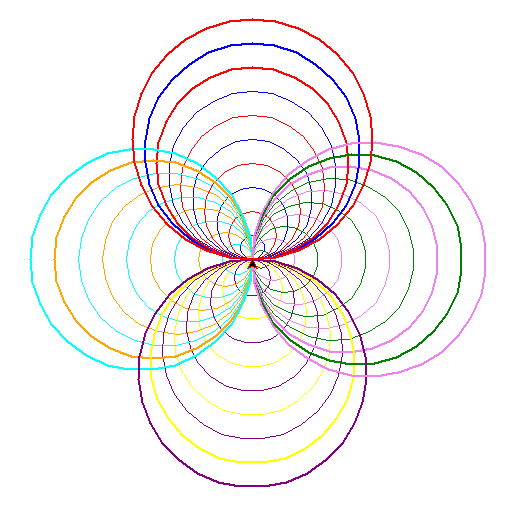
    t.pencolor(colors[x % 8])     # 循環選擇顏色（x % 8 讓顏色重複使用）

    t.circle(x \* 3)               # 畫一個半徑為 x\*3 的圓

    t.width(x \* 0.05)             # 隨著迴圈增加畫筆寬度，讓線條漸粗

    t.left(90)                    # 每次畫完圓向左轉 90 度，產生旋轉效果

turtle.done()  # 保持視窗開啟直到手動關閉



## 範例：

import turtle

t = turtle.Pen()

t.pensize(5)

colorValue = 1.0

colorStep = colorValue / 36

for x in range(1, 40):

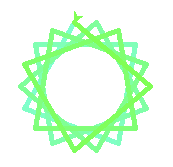
    colorValue -= colorStep

    t.color(0.5, 1, colorValue)

    t.forward(100)

    t.left(100)

turtle.done()



import turtle

t = turtle.Pen()

colorsList = ['red', 'orange', 'yellow', 'green', 'blue', 'cyan', 'purple', 'violet']

tWidth = 1      # 最初畫筆寬度

for x in range(1, 41):

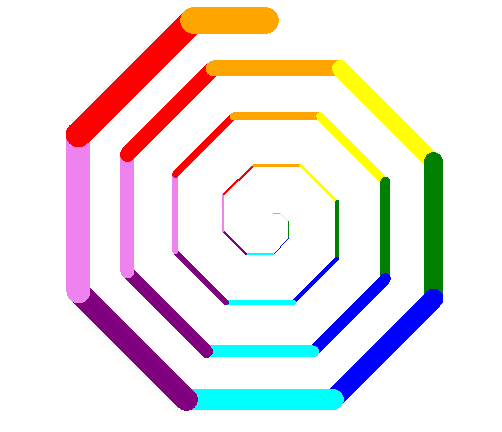
    t.color(colorsList[x % 8])  # 選擇畫筆顏色

    t.forward(2 + x \* 5)        # 每次移動距離

    t.right(45)                 # 每次旋轉角度

    tWidth += x \* 0.05          # 每次畫筆寬度遞增

    t.width(tWidth)



# 填滿顏色

|  |  |
| --- | --- |
| begin\_fill() | 開始記錄繪製路徑以準備填色 |
| end\_fill() | 填滿路徑內的區域 |
| fillcolor(color\_string) | 設定填滿的顏色，例如 "red" 或 "green" |
| fillcolor(r, g, b) | 使用 RGB 值設定填滿顏色 |

## 範例：繪製一個填滿顏色的五角形星星

import turtle

t = turtle.Pen()

t.color('blue', 'yellow') # 設定畫筆和填色顏色

t.begin\_fill()

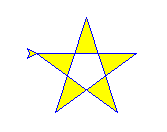
for i in range(5):

    t.forward(100)

    t.right(144)

t.end\_fill()

turtle.done()



## 繪製圓弧形

|  |  |
| --- | --- |
| circle(radius, extend=None) | radius 為半徑，extend 為繪製圓弧的角度 |

## 範例：

import turtle

t = turtle.Pen()

t.color('blue')

t.penup()

t.setheading(180)

t.forward(150)

t.setheading(0)

t.pendown()

t.circle(50)

t.circle(-50)

t.forward(100)

t.circle(50)

t.circle(-50)

t.penup()

t.forward(100)

t.pendown()

t.setheading(0)

step = 5

for r in range(10, 100+step, step):

    t.penup()

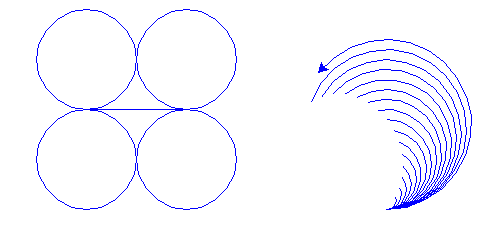
    t.setpos(150, -100)

    t.setheading(0)

    t.pendown()

    t.circle(r, 90 + r\*2)

turtle.done()



# 螢幕與海龜控制

|  |  |
| --- | --- |
| 方法 | 說明 |
| screen.title(string) | 設定視窗標題 |
| screen.bgcolor(color\_string) | 設定背景顏色 |
| screen.setup(width, height) | 設定視窗寬度和高度 |
| t.hideturtle() | 隱藏海龜 |
| t.showturtle() | 顯示海龜 |

## 範例：繪製一個藍色天空下的五角形星星

import turtle

t = turtle.Pen()

turtle.Screen().bgcolor('blue') # 設定背景顏色為藍色

t.color('yellow')

t.begin\_fill()

for i in range(5):

    t.forward(100)

    t.right(144)

t.end\_fill()

turtle.done()



# 停止追蹤

turtle.tracer(0,0) 可以停止動畫，一次顯現出來

## 範例：

import turtle

turtle.tracer(0,0)

t = turtle.Pen()

colorsList = ['red', 'green', 'blue']

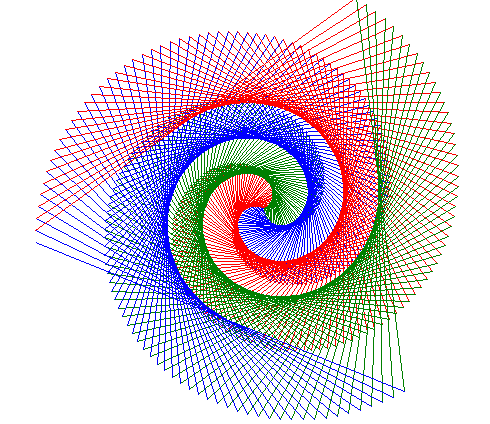
for line in range(400):

    t.color(colorsList[line % 3])

    t.forward(line)

    t.right(119)

turtle.done()



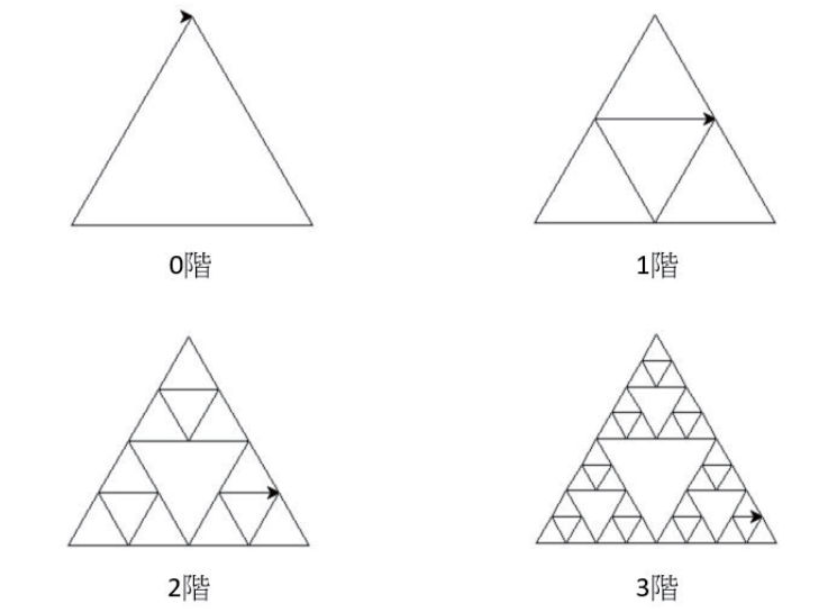
小專題：謝爾賓斯基三角形（Sierpinski Triangle）

謝爾賓斯基三角形是一個經典的碎形（Fractal）圖案，由波蘭數學家謝爾賓斯基在 1915 年提出。它的生成規則與科赫雪花類似，都是基於遞迴（Recursive）的概念，可以從一個簡單的三角形開始，不斷地進行迭代。

* 初始狀態（0 階）：一個實心的等邊三角形。
* 迭代規則：
  + 找到目前三角形三條邊的中點。
  + 將這三個中點相連，形成一個新的、倒立的等邊三角形。
  + 將這個新形成的倒立三角形從原三角形中移除。
  + 這個操作將原三角形分割成三個較小的、實心的正立三角形。

重複這個步驟，你就能得到更高階的謝爾賓斯基三角形。當這個過程無限次地進行下去時，最終形成的圖案就是謝爾賓斯基三角形。

* **0 階**：一個實心的大三角形。
* **1 階**：大三角形中間被挖空，留下三個較小的大三角形。
* **2 階**：三個小三角形的中間再次被挖空，留下九個更小的三角形。
* **3 階**：……以此類推。



畫面呈現：

|  |  |  |
| --- | --- | --- |
|  |  |  |

此專案這個小專題是給你練習燒腦看看，

當然google/YT上其實都有一堆現成的程式碼

你當然可以查，就是利用這份檔案+網路上資料+gpt，去練習把程式碼寫出來

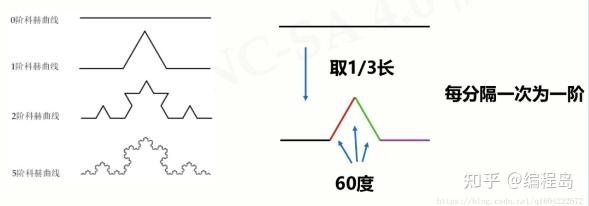
當然，也要看懂程式碼在幹嘛呦

小專題：科赫雪花（Koch Snowflake）

科赫雪花是由三條科赫曲線（Koch Curve）組成的。所以，我們首先需要了解科赫曲線的生成規則。

科赫曲線是一種遞迴（Recursive）圖案，其生成規則非常簡單：

* 初始狀態（0 階）：一條直線段。
* 迭代規則：將每一條直線段分割成四個長度為原線段三分之一的小線段。
  + 將中間三分之一的線段移除。
  + 在這個空缺處，向外凸出一個等邊三角形。



重複這個步驟，你就能得到更高階的科赫曲線。當這個過程無限次地進行下去時，最終形成的曲線就是科赫曲線。

畫面呈現：

|  |  |  |
| --- | --- | --- |
|  |  |  |

此專案這個小專題是給你練習燒腦看看，

當然google/YT上其實都有一堆現成的程式碼

你當然可以查，就是利用這份檔案+網路上資料+gpt，去練習把程式碼寫出來

當然，也要看懂程式碼在幹嘛呦