Python 语言画图教程

——少儿计算机和几何入门课程

[第1课：了解turtle画图程序 2](#_Toc25396662)

[用乌龟程序画一个正方形 2](#_Toc25396663)

[第2课：了解turtle画图程序 3](#_Toc25396664)

[用乌龟程序画一个三角形 3](#_Toc25396665)

[第3课：画圆、半圆、弧、正n边形 4](#_Toc25396666)

[第4课：循环画线段 6](#_Toc25396667)

[第5课：延迟画五角星 7](#_Toc25396668)

[第6课：改变画图速度 8](#_Toc25396669)

[第7课：顺时针画五角星 9](#_Toc25396670)

[第8课：画给定边数的轮轴 10](#_Toc25396671)

[第9课：循环画四边形（正方形） 11](#_Toc25396672)

[第10课：画正多边形 12](#_Toc25396673)

[第11课：画类似螺线的正n边形 13](#_Toc25396674)

[第12课：循环画圆 14](#_Toc25396675)

[第13课：循环画半圆 15](#_Toc25396676)

[第14课：循环画部分圆和平移旋转结合 16](#_Toc25396677)

[第15课：平移、画多边形、旋转相结合 17](#_Toc25396678)

[第16课：迭代生成多边形和螺线 18](#_Toc25396679)

[第17课：递归得到分形图（三角形） 19](#_Toc25396680)

[第18课：循环画折线，绕圆心一周。 21](#_Toc25396681)

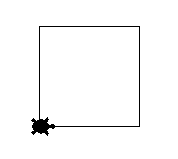
[第19课：对封闭的多边形进行填充颜色 22](#_Toc25396682)

[Python Turtle Graphics 25](#_Toc25396683)

# 第1课：了解turtle画图程序

## 用乌龟程序画一个正方形

**重复4次执行两条语句：1.向前走100, 2. 左转90°（逆时针旋转90°）。**

****

# -\*- coding: utf-8 -\*-

"""

Lesson 1 : Draw a square 第一课：画一个正方形

"""

import turtle  # 调用 乌龟turtle库

win = turtle.Screen()  # Create a graphics windows 创建绘画窗口

Yang = turtle.Pen()    # 初始化乌龟程序，调出图形框，准备好画笔，给画师命名

Yang.shape("turtle")   # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

for i in range(4):  # 重复4次

    Yang.fd(100)  # 简写向前走100步 forward(100)

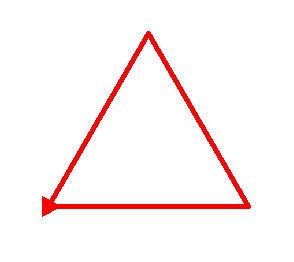
    Yang.lt(90)  # 左转90度 left(90)

# 其他简写 penup()→up(), pendown()→down(),right()→rt()

win.exitonclick()

# 第2课：了解turtle画图程序

## 用乌龟程序画一个三角形



# -\*- coding: utf-8 -\*-

"""

Lesson 2 : Draw a triangle 画一个等边三角形

"""

import turtle

win = turtle.Screen()  # Create a graphics windows 创建绘画窗口

t = turtle.Pen()     # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("triangle")  # 改变画笔形状为triangle，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

t.width(5)          # 改变线宽度

t.color("red")      # 改变画笔颜色, 还有green,blue,black,white,pink,...,或者(r,g,b)

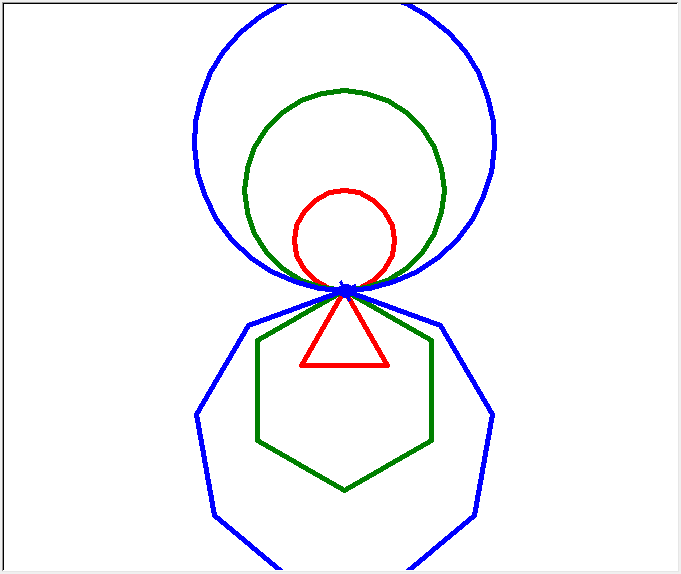
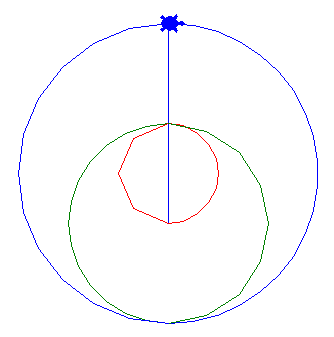
for i in range(3):  # 重复三次

    t.fd(200)  # 向前走200

    t.lt(120)  # 逆时针旋转120度，左转120°

win.exitonclick()

# 第3课：画圆、半圆、弧、正n边形

前半部分后半部分

# -\*- coding: utf-8 -\*-

"""

Lesson 3 : Draw a circle 画圆、半圆、弧、正多边形

"""

import turtle

win = turtle.Screen()  # Create a graphics windows 创建绘画窗口

t = turtle.Pen()    # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle")   # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

t.width(5)          # 改变线宽度

t.color("red")      # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

mycolor = ["red","green","blue"]

for i in range(3):

    t.color(mycolor[i])

    t.circle(50\*(i+1))

"""call: circle(radius)                  # 画整个圆

    |--or: circle(radius, extent)        # 画弧--部分圆

    |--or: circle(radius, extent, steps)

    |--or: circle(radius, steps=6)       # 画六边形

"""

for i in range(3):

    t.color(mycolor[i])

    t.circle(-50\*(i+1),steps=(i+1)\*3)  # 画正n边形

t.reset()  # 清屏，重来

for i in range(3):

    t.color(mycolor[i])

    t.circle(50\*(i+1),180)  # 画半圆

t.setposition(0,0)  # setpos(x,y) 移动圆点（0,0）

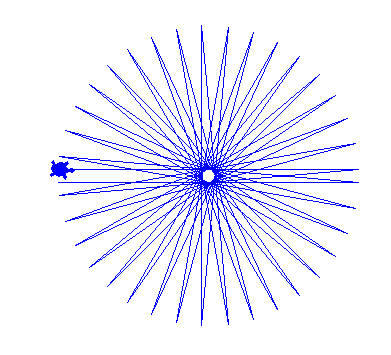
for i in range(3):

    t.color(mycolor[i])

    t.circle(-50\*(i+1),180,(i+1)\*4)  # 画正n边形 代表 半圆

win.exitonclick()

# 第4课：循环画线段



# -\*- coding: utf-8 -\*-

"""

Lesson 4 : Draw a segment loop （循环画线段），就两个动作：平移与旋转

"""

import turtle

win = turtle.Screen()  # Create a graphics windows 创建绘画窗口

t = turtle.Pen()    # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle")   # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

t.width(1)          # 改变线宽度

t.color("blue")     # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

for i in range(1,37):  # 循环次数可以更改

    t.fd(300)

    t.lt(175)  # 逆时针旋转角度175°，也可以改变

win.exitonclick()

# 第5课：一笔画五角星（奇数角）



# -\*- coding: utf-8 -\*-

"""

Lesson 5 : Draw Star Delay 以红色为背景，带延迟一笔画五角星(n=5)或n角♥

"""

import turtle

wn = turtle.Screen()

t = turtle.Pen()  # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("arrow")  # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

wn.bgcolor("red")  # lightgreen

t.pensize(3)       # 改变画笔线宽度

t.color("yellow")  # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

wn.delay(10)  # 延迟100毫秒

**n = 7  # n为奇数**

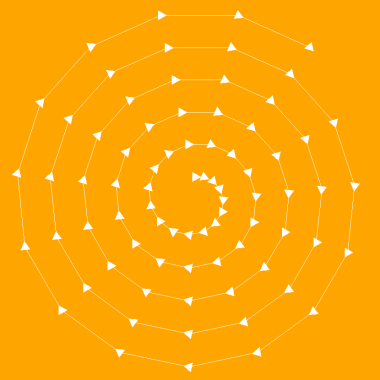
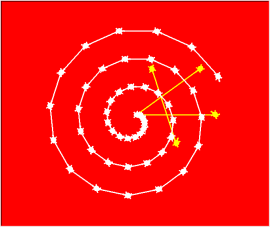
for \_ in range(n):

    t.fd(307)  # prime

    t.rt(180-180/n)  # 五角星的内角为36°=180°/5，也可以是left(144)

wn.exitonclick()  # 鼠标点击就退出来

# 第6课：改变速度画螺线图



# -\*- coding: utf-8 -\*-

"""

Lesson 6 : 给定速度画螺线图

"""

import turtle

wn = turtle.Screen()

monkey = turtle.Pen()    # 初始化乌龟程序，调出图形框，准备好画笔

wn.bgcolor('orange')

monkey.color("white")  # 改变画笔颜色

monkey.speed(7)

monkey.pensize(1)

# 画类似螺线或蜘蛛网

monkey.shape('triangle')

for size in range(5, 173, 2):  # start with size = 5 and grow by 2

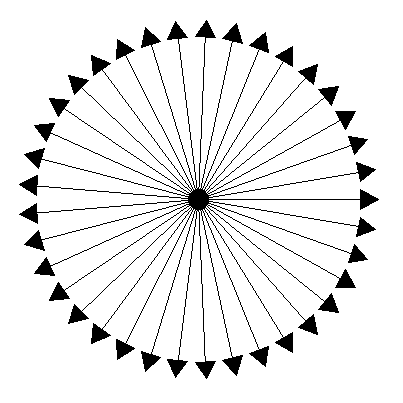
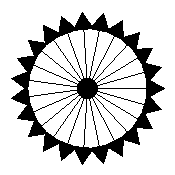
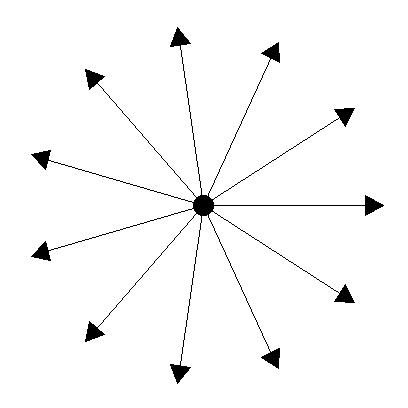
    monkey.stamp()        # leave an impression on the canvas

    monkey.forward(size)  # move turtle along

    monkey.right(24)      # and turn her

wn.exitonclick()  # 鼠标点击就退出

# 第7课：画给定边数（n边形）的轴



# -\*- coding: utf-8 -\*-

"""

Lesson 7 : 给定n值，画正n边形的轴

"""

import turtle

wn = turtle.Screen()

xpen = turtle.Turtle()

xpen.shape("triangle")

n = int(input("How many legs should this sprite have? "))

angle = 360 / n

len = 168

for \_ in range(n):

    # draw the leg

    xpen.right(angle)

    xpen.forward(len)

    xpen.stamp()

    # go back to the middle and turn back around

    xpen.right(180)

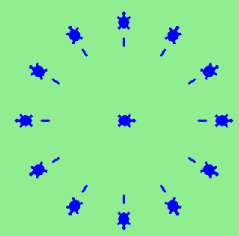
    xpen.forward(len)

    xpen.right(180)

xpen.shape("circle")

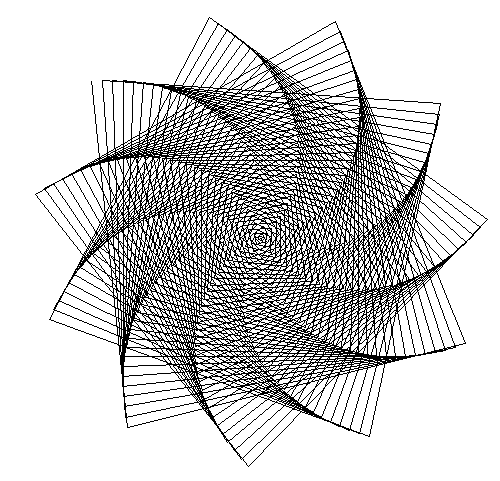
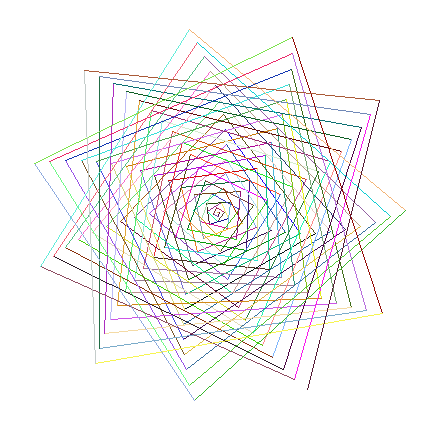
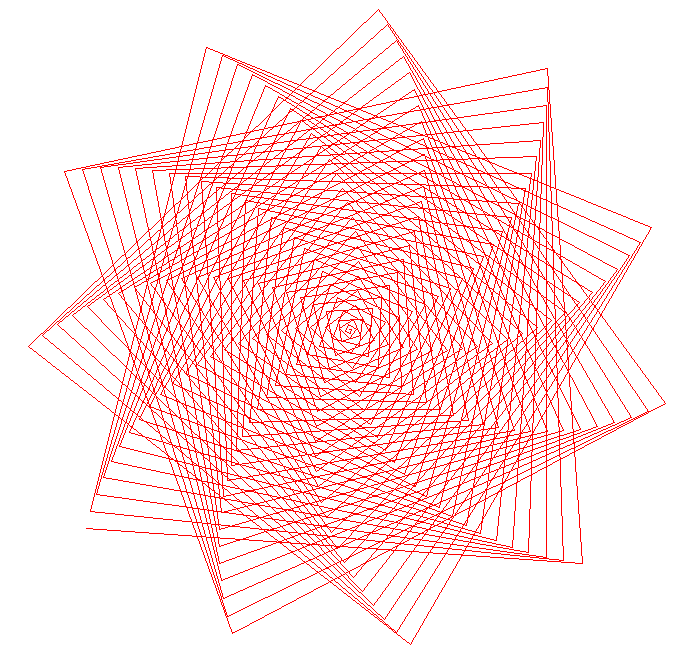
wn.exitonclick()

# 练习：画下列钟表图形



# 第8课：画给定边数的轮轴

# 第9课：循环画四边形（正方形）

# -\*- coding: utf-8 -\*-

"""

Lesson 9 : 循环画四边形

"""

import turtle

import random

mt = turtle.Turtle()

mt.hideturtle()

mt.speed(20)

scr = turtle.Screen()

for i in range(1,150): # 150可以改成350

mt.color(random.random(),random.random(),random.random()) # 随机颜色

mt.forward(i\*2)

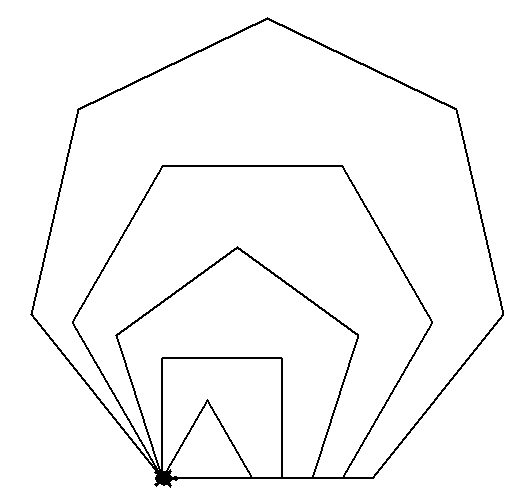
mt.right(98)

scr.exitonclick()

# 第10课：画正多边形

# -\*- coding: utf-8 -\*-

"""

Lesson 10 : 画正n边形 

"""

import turtle

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(2) # 改变线宽度

t.color("black") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.penup()

t.setpos(-100,-200)

t.pendown()

# print("画正多边形例子")

for n in range(3,8):

for i in range(n):

t.fd(n\*30)

t.lt(360.0/n) # 逆时针旋转 2pi/n 度

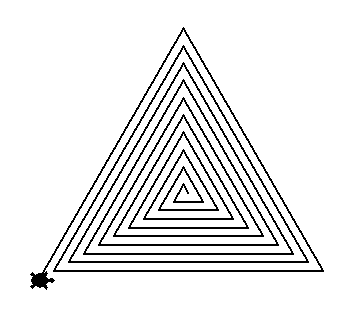
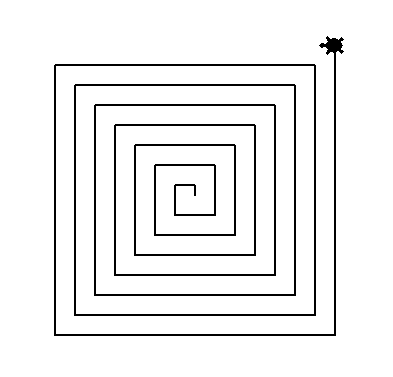
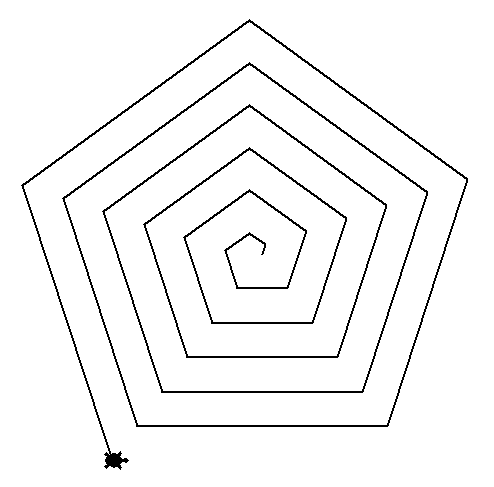
scr.exitonclick()

# 第11课：画类似螺线的正n边形

# -\*- coding: utf-8 -\*-

"""

Lesson 11 : 画类似螺线的正n边形

"""

import turtle

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(2) # 改变线宽度

t.color("black") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

for n in range(30):

t.fd(n\*10)

t.lt(72) # 改变角度试试，90,120，72，...

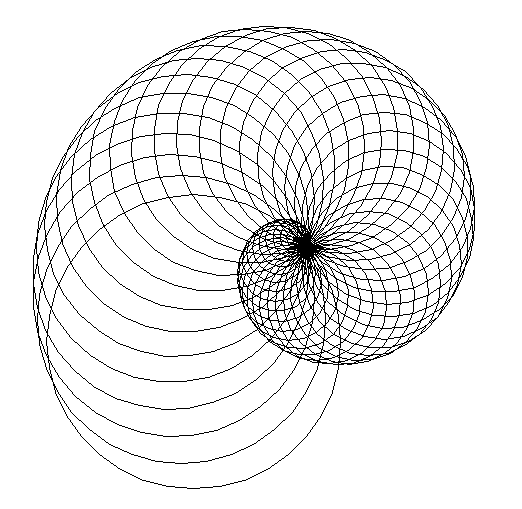
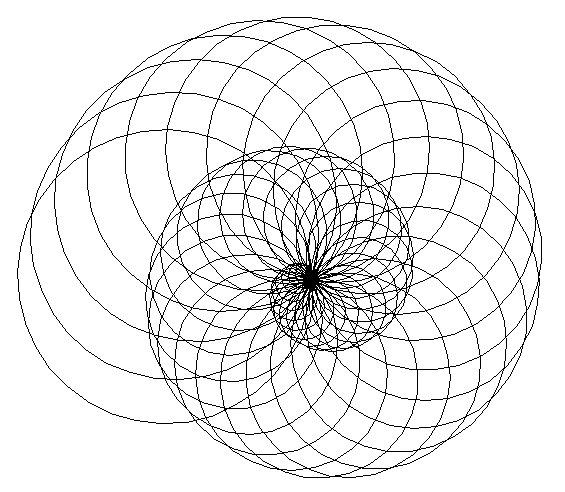
scr.exitonclick()

# 第12课：循环画圆

# -\*- coding: utf-8 -\*-

"""

Lesson 12 : 循环画圆

"""

import turtle

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(1) # 改变线宽度

t.color("black") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

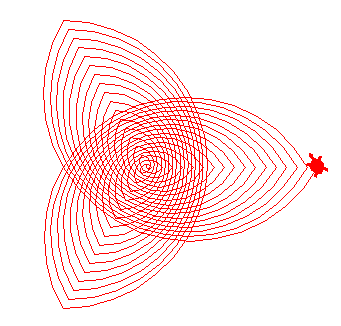
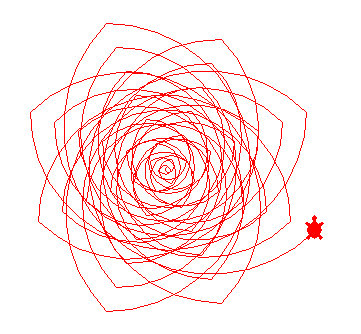
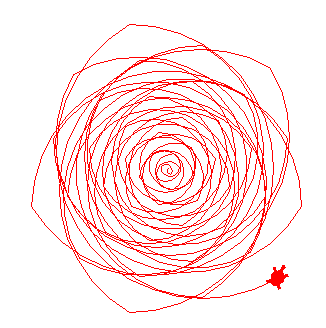
for n in range(50):

t.circle(n\*3)

t.lt(10) # 改变角度试试，90,120，72，...

scr.exitonclick()

# 第13课：循环画半圆

# -\*- coding: utf-8 -\*-

"""

Lesson 13 : 循环画半圆

"""

import turtle

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(1) # 改变线宽度

t.color("red") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

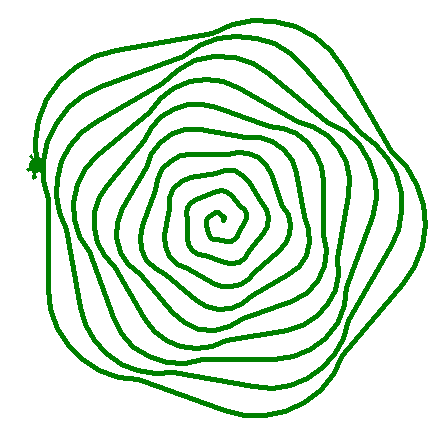
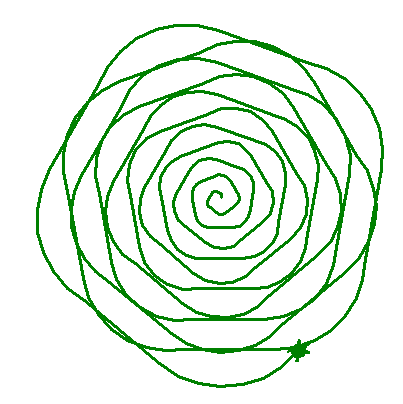
for n in range(50):

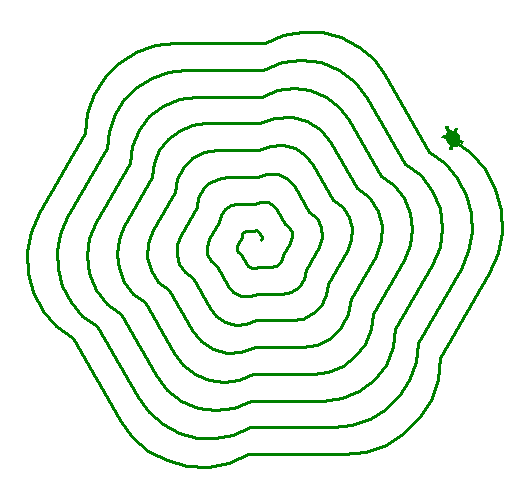
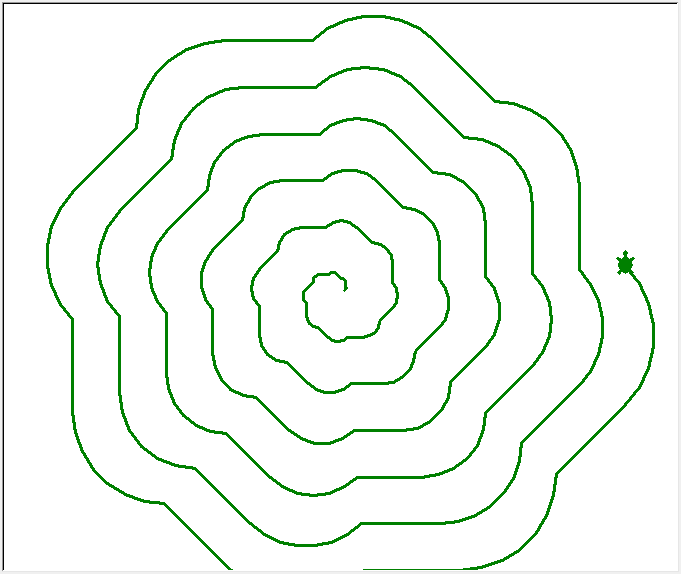
t.circle(n\*3,180)

t.lt(60) # 改变角度试试，60,45,30

scr.exitonclick()

# 第14课：循环画部分圆和平移旋转结合

# -\*- coding: utf-8 -\*-

"""

Lesson 14 : 循环画部分圆和平移旋转结合

"""

import turtle

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(3) # 改变线宽度

t.color("green") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

for n in range(50):

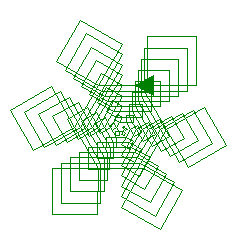
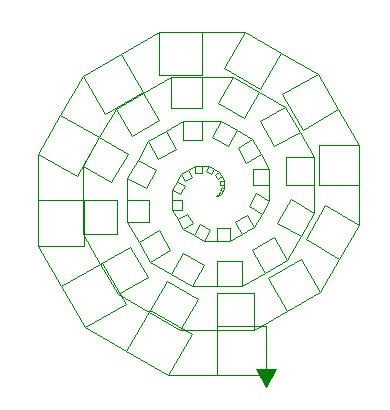
t.fd(n\*2)

t.circle(n\*2,90)

t.rt(45) # 改变角度试试，20,10,30,45

scr.exitonclick()

# 第15课：平移、画多边形、旋转相结合



# -\*- coding: utf-8 -\*-

"""Lesson 15 : 平移、画多边形、旋转结合

"""

import turtle

def draw\_poly(turtle,length,n): # draw n-polygon 画正n边形

for i in range(n):

turtle.fd(length)

turtle.left(360/n)

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("triangle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(1) # 改变线宽度

t.color("green") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

for n in range(1,50):

t.fd(n) # 也可以隐藏之，原地旋转。

draw\_poly(t,n,4) # 改变 为正 5,4,3边形

t.right(60) # 改变角度试试，5,10,15

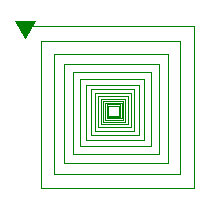
scr.exitonclick()

# 第16课：迭代生成多边形和螺线

# -\*- coding: utf-8 -\*-

"""

Lesson 16 : 迭代生成多边形和螺线



"""

import turtle

def draw\_spiral(turtle,length,angle,scale,n):

for i in range(1,n):

turtle.forward(length)

turtle.left(angle)

length \*= scale

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("triangle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(1) # 改变线宽度

t.color("green") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

draw\_spiral(t,10,90,1.05,60)

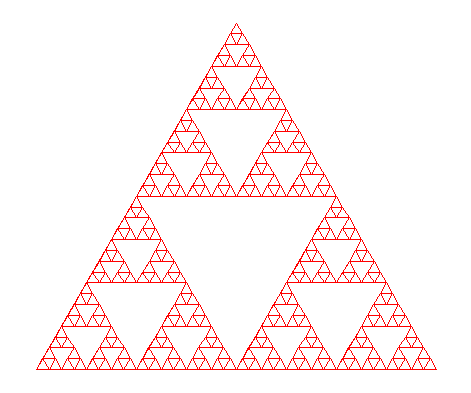
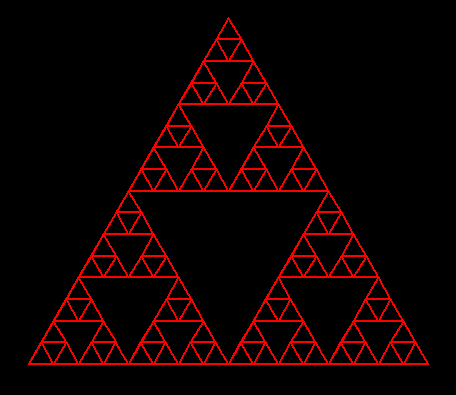
scr.exitonclick()

# 第17课：递归得到分形图（三角形）

# -\*- coding: utf-8 -\*-

"""

Lesson 17 : 利用递归算法，生成分形图形



"""

import turtle

# 引入函数，过程，将重复动作组合在一个函数中。

def sierpinski(turtle,length,level):

if level == 0:

triangle(turtle,length)

else:

#for i in range(3):重复三次画三角形，分别在三个顶点上画三角形

curpos = t.position() # 保留起始点坐标

length \*= 0.5

sierpinski(turtle,length,level-1)

turtle.penup()

turtle.forward(length)

turtle.pendown()

sierpinski(turtle,length,level-1)

turtle.penup()

turtle.left(120)

turtle.forward(length)

turtle.right(120)

turtle.pendown()

sierpinski(turtle,length,level-1)

t.setposition(curpos[0],curpos[1]) # 回到起始点，方向也复原

def triangle(t,length):

for i in range(3):

t.forward(length)

t.left(120)

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("turtle") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("black") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(2) # 改变线宽度

t.color("red") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

t.penup()

t.setposition(-200,-100)

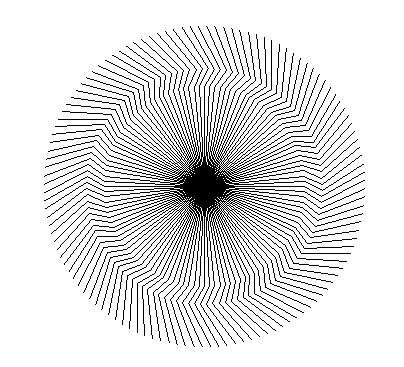
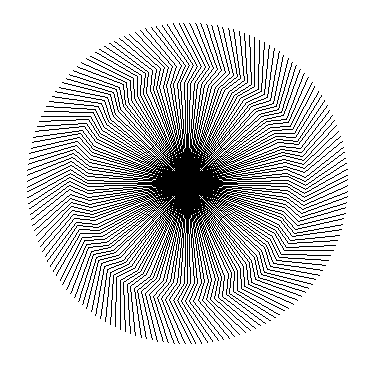
t.pendown()

sierpinski(t,400,4)

t.hideturtle() #隐藏乌龟图形

scr.exitonclick()

# 第18课：循环画折线，绕圆心一周。



# -\*- coding: utf-8 -\*-

"""

Lesson 18 : 利用递归算法，生成分形图形

"""

import turtle

powell = turtle.Turtle()

powell.speed(10)

for i in range(180):

powell.forward(100)

powell.right(30)

powell.forward(20)

powell.left(60)

powell.forward(50)

powell.right(30)

powell.penup()

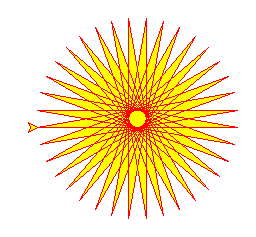
powell.setposition(0, 0)

powell.pendown()

powell.right(2)

turtle.done()

# 第19课：对封闭的多边形进行填充颜色



# -\*- coding: utf-8 -\*-

""" Lesson 19 : 填充颜色

"""

import turtle

powell = turtle.Turtle()

picture = turtle.Screen()

powell.speed(5)

powell.color("red","yellow")

powell.begin\_fill()

#for i in range(120):

while True:

powell.forward(200)

powell.left(170)

if abs(powell.pos()) < 1: # turtle.pos() 返回的值是turtle.Vec2D,重新回到了起始点

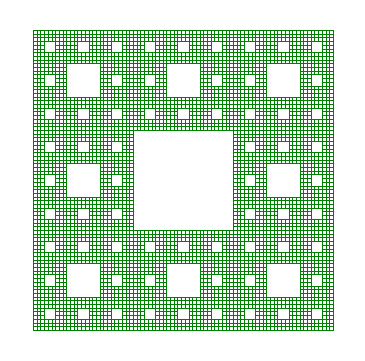
break

powell.end\_fill()

# turtle.done()

picture.exitonclick() # 不能同时使用 turtle.done() 和 screen.exitonclick()

第20课：递归生成Sierpinski正方形地毯分形图案



# -\*- coding: utf-8 -\*-

"""Lesson 20 : 利用递归算法，生成分形图形（Sierpinski正方形地毯）"""

import turtle

# 引入函数，过程，将重复动作组合在一个函数中。

def sierpinski2(turtle,length,level):

if level == 0:

square(turtle,length) # 画正方形的过程或函数

else:

curpos = t.position() # 保留起始点坐标

length /= 3.0 # 长度变化因子，缩小为1/3

sierpinski2(turtle,length,level-1) # 第1个正方形递归

for i in range(2): # 第2,3个正方形递归

turtle.penup()

turtle.forward(length)

turtle.pendown()

sierpinski2(turtle,length,level-1)

for i in range(2): # 第4,5个正方形递归

turtle.penup()

turtle.left(90)

turtle.forward(length)

turtle.right(90) # 注意调整方向

turtle.pendown()

sierpinski2(turtle,length,level-1)

turtle.penup()

turtle.backward(2\*length)

turtle.pendown()

sierpinski2(turtle,length,level-1)

turtle.penup()

turtle.forward(length)

turtle.pendown()

sierpinski2(turtle,length,level-1)

turtle.penup()

turtle.backward(length)

turtle.right(90)

turtle.forward(length)

turtle.left(90)

turtle.pendown()

sierpinski2(turtle,length,level-1)

t.goto(curpos[0],curpos[1]) # 回到起始点，方向也复原

def square(turtle,length):

for i in range(4):

turtle.forward(length)

turtle.left(90)

scr = turtle.Screen()

t = turtle.Pen() # 初始化乌龟程序，调出图形框，准备好画笔

t.shape("classic") # 改变画笔形状为一只乌龟，缺省是箭头arrow，

# 还可以为 'circle'-圆, 'square'-正方形, 'triangle'-三角形, 'classic'.

scr.bgcolor("white") # (0.1,0.51,0.3) # red,green,blue 取值在0和1之间。1代表255,

t.pensize(1) # 改变线宽度

t.color("green") # 改变画笔颜色,还有green,blue,black,white,pink,...,或者(r,g,b)

t.speed(0)

t.penup()

t.setposition(-200,-100)

t.pendown()

sierpinski2(t,300,3)

t.hideturtle() # 隐藏乌龟图形

scr.exitonclick()

# Python Turtle Graphics

* [4.1. Hello Little Turtles!](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/intro-HelloLittleTurtles.html)
* [4.2. Our First Turtle Program](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/OurFirstTurtleProgram.html)
* [4.3. Instances — A Herd of Turtles](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/InstancesAHerdofTurtles.html)
* [4.4. The for Loop](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/TheforLoop.html)
* [4.5. Flow of Execution of the for Loop](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/FlowofExecutionoftheforLoop.html)
* [4.6. Iteration Simplifies our Turtle Program](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/IterationSimplifiesourTurtleProgram.html)
* [4.7. The range Function](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/TherangeFunction.html)
* [4.8. A Few More turtle Methods and Observations](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/AFewMoreturtleMethodsandObservations.html)
* [4.9. Summary of Turtle Methods](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/SummaryofTurtleMethods.html)
* [4.10. Glossary](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/Glossary.html)
* [4.11. Exercises](http://www.interactivepython.org/runestone/static/thinkcspy/PythonTurtle/Exercises.html)