**第一课：python编程介绍**

一、环境要求

Python（中文名：蟒蛇）是一种跨平台的计算机程序设计语言。 是一个高层次的结合了解释性、编译性、互动性和面向对象的脚本语言。它的应用领域非常广泛。Python除了极少的涉及不到的开发之外，其他基本上可以说全能：系统运维、图形处理、数学处理、文本处理、数据库编程、网络编程、web编程、多媒体应用、pymo引擎、爬虫编写、机器学习、人工智能等等。

电脑设备要求：win7以上系统，Mac系统，运行内存4g

二、python语言特点

思考：对比其他编程语言而言，你觉得python有哪些语言特点呢？

1．书写简洁

2．有非常强大的第三方库

3．直接使用缩进符对内容进行区分

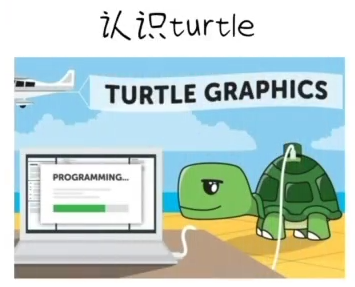
|  |  |
| --- | --- |
| **python语言的特点** | |
| **优点** | |
| **简单** | Python是一种代表简单主义思想的语言。 阅读一个良好的Python程序就感觉像是在读英语一样。它使你能够专注于解决问题而不是去搞明白语言本身。 |
| **易学** | Python极其容易上手，因为Python有极其简单的说明文档 |
| **速度快** | Python 的底层是用 C 语言写的，很多标准库和第三方库也都是用 C 写的，运行速度快 |
| **免费、开源** | 使用者可以自由地发布这个软件的拷贝、阅读它的源代码、对它做改动、把它的一部分用于新的自由软件中 |
| **高层语言** | 用Python语言编写程序的时候无需考虑诸如如何管理你的程序使用的内存一类的底层细节 |
| **可移植性** | 由于它的开源本质，Python已经被移植在许多平台上 |
| **面向对象** | Python既支持面向过程的编程也支持面向对象的编程。 |
| **可嵌入性** | 可以把Python嵌入C/C++程序，从而向程序用户提供脚本功能。 |
| **丰富的库** | Python标准库确实很庞大 |
| **规范的代码** | Python采用强制缩进的方式使得代码具有较好可读性。而Python语言写的程序不需要编译成二进制代码。 |
| **缺点** | |
| **单行语句和命令行输出问题** | 很多时候不能将程序连写成一行 |

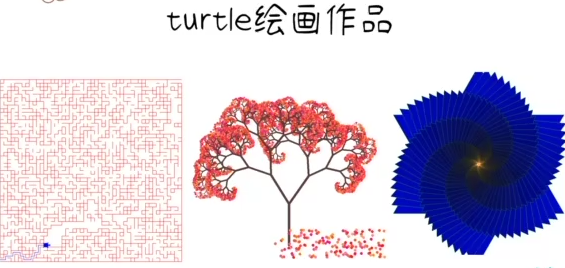
|  |  |
| --- | --- |
| **python第三方库** | |
| **pygame** | Pygame是用于开发2D游戏的python库，也是最受欢迎的库之一。学会使用Pygame库，我们能做出各种好玩有趣的游戏及互动程序。 |
| **easygui** | Easygui是常见的GUI (Graphics User Interface，用户图形界面)库，我们可以使用这个库来编写简单的图形界面。 |
| **pandas** | Pandas库经常被用于数据读取与处理。强大易用的数据分析方法，以及优化的数据处理速度，使得Pandas成为数据科学家最钟爱的Python库之一。 |
| **Tkinter** | 各种图形界面控件,我们就可以设计出实用的图形界面程序。 |
| **wordcloud** | wordcloud意为“词云”，它可以生成词云图，对文本中出现频率较高的“关键词”予以视觉化的展现，直观展示文本关键内容及主旨。 |
| **beautifulsoup4** | BeautifulSoup是-个可以从网站抓取数据并解析的Python库。学会Beautiful Soup库，我们可以用更简洁、高效的方式编写爬虫程序。 |
| **Matplotlib** | Matplotlib是Python最常用的可视化工具，它可以被用来生成直方图、折线图、散点图等数据图表，是数据分析领域常见的小帮手。 |
| **Numpy** | NumPy是一个用于科学计算的基础库，尤其在矩阵运算方面有着明显的优势，还能方便地进行线性代数、傅里叶变换和随机数操作等。 |

**第二课：Python绘图Turtle库初识**

1. 认识turtle库

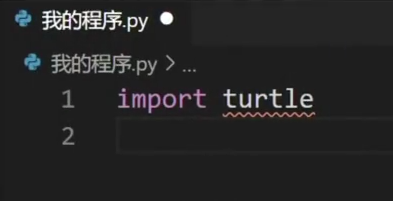
Turtle库是Python语言中一个很流行的绘制图像的函数库，想象一个小乌龟，在一个横轴为x、纵轴为y的坐标系原点，(0,0)位置开始，它根据一组函数指令的控制，在这个平面坐标系中移动，从而在它爬行的路径上绘制了图形。





1. 如何使用turtle库？

代码：import turtle（见下图）



1. Turtle库绘图指令



1. 画几何图形

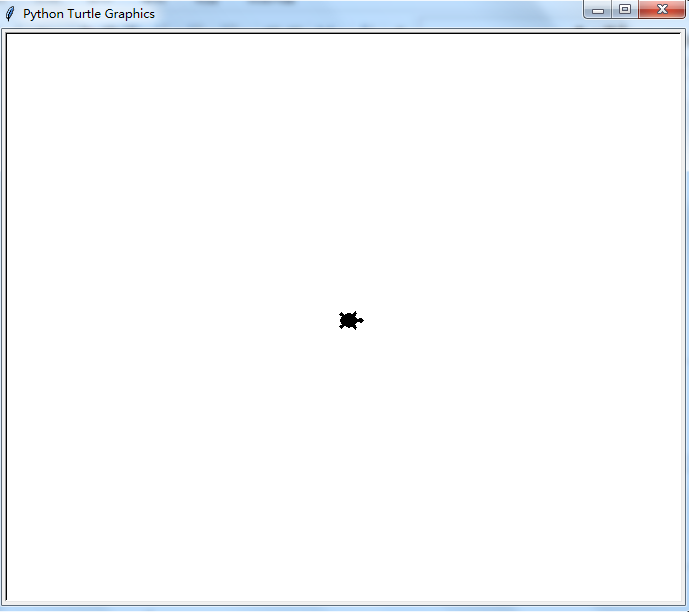
1．编程实例1：显示海龟图形

import turtle

turtle.shape("turtle")

turtle.done()

程序运行结果：



2．编程实例2：海龟画直线

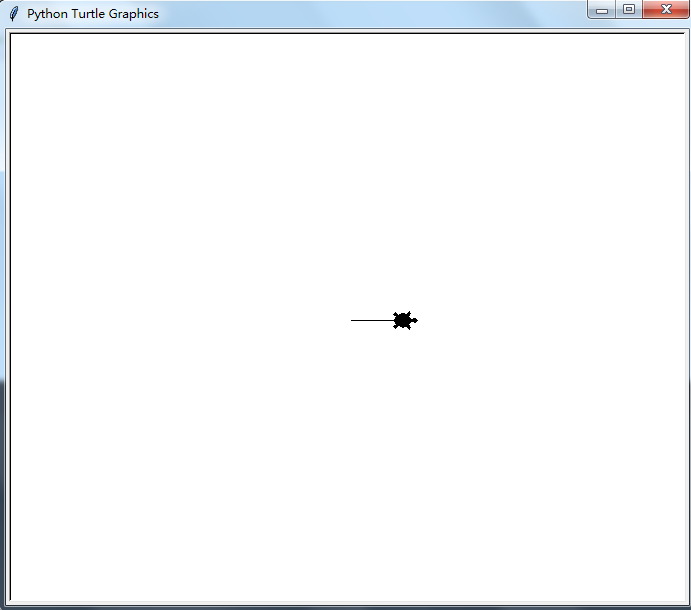
import turtle

turtle.shape("turtle")

turtle.forward(50)

turtle.done()

程序运行结果：



3．编程实例3：海龟画正方形

import turtle

turtle.shape("turtle")

turtle.left(90)

turtle.forward(50)

turtle.left(90)

turtle.forward(50)

turtle.left(90)

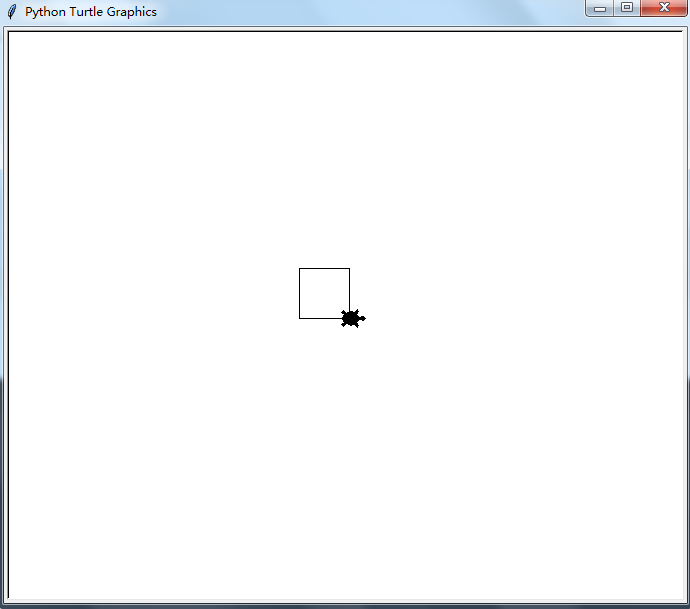
turtle.forward(50)

turtle.left(90)

turtle.forward(50)

turtle.done()

程序运行结果：



思考题：如何画三角形？

1. 简化模块的名称：as

|  |  |
| --- | --- |
| 代码对比 | |
|  |  |

**第三课：Python绘图Turtle库详解**

turtle绘图的基础知识：

1. 画布(canvas)

        画布就是turtle为我们展开用于绘图区域，我们可以设置它的大小和初始位置。

        设置画布大小

         turtle.screensize(canvwidth=None, canvheight=None, bg=None)，参数分别为画布的宽(单位像素), 高, 背景颜色。

        如：turtle.screensize(800,600, "green")

               turtle.screensize() #返回默认大小(400, 300)

        turtle.setup(width=0.5, height=0.75, startx=None, starty=None)，参数：width, height: 输入宽和高为整数时, 表示像素; 为小数时, 表示占据电脑屏幕的比例，(startx, starty): 这一坐标表示矩形窗口左上角顶点的位置, 如果为空,则窗口位于屏幕中心。

        如：turtle.setup(width=0.6,height=0.6)

               turtle.setup(width=800,height=800, startx=100, starty=100)

2. 画笔

2.1 画笔的状态

        在画布上，默认有一个坐标原点为画布中心的坐标轴，坐标原点上有一只面朝x轴正方向小乌龟。这里我们描述小乌龟时使用了两个词语：坐标原点(位置)，面朝x轴正方向(方向)， turtle绘图中，就是使用位置方向描述小乌龟(画笔)的状态。

2.2 画笔的属性

        画笔(画笔的属性，颜色、画线的宽度等)

        1) turtle.pensize()：设置画笔的宽度；

        2) turtle.pencolor()：没有参数传入，返回当前画笔颜色，传入参数设置画笔颜色，可以是字符串如"green", "red",也可以是RGB 3元组。

        3) turtle.speed(speed)：设置画笔移动速度，画笔绘制的速度范围[0,10]整数，数字越大越快。

2.3 绘图命令

         操纵海龟绘图有着许多的命令，这些命令可以划分为3种：一种为运动命令，一种为画笔控制命令，还有一种是全局控制命令。

(1)    画笔运动命令

|  |  |
| --- | --- |
| 命令 | 说明 |
| turtle.forward(distance) | 向当前画笔方向移动distance像素长度 |
| turtle.backward(distance) | 向当前画笔相反方向移动distance像素长度 |
| turtle.right(degree) | 顺时针移动degree° |
| turtle.left(degree) | 逆时针移动degree° |
| turtle.pendown() | 移动时绘制图形，缺省时也为绘制 |
| turtle.goto(x,y) | 将画笔移动到坐标为x,y的位置 |
| turtle.penup() | 提起笔移动，不绘制图形，用于另起一个地方绘制 |
| turtle.circle() | 画圆，半径为正(负)，表示圆心在画笔的左边(右边)画圆 |
| setx( ) | 将当前x轴移动到指定位置 |
| sety( ) | 将当前y轴移动到指定位置 |
| setheading(angle) | 设置当前朝向为angle角度 |
| home() | 设置当前画笔位置为原点，朝向东。 |
| dot(r) | 绘制一个指定直径和颜色的圆点 |

 (2)     画笔控制命令

|  |  |
| --- | --- |
| 命令 | 说明 |
| turtle.fillcolor(colorstring) | 绘制图形的填充颜色 |
| turtle.color(color1, color2) | 同时设置pencolor=color1, fillcolor=color2 |
| turtle.filling() | 返回当前是否在填充状态 |
| turtle.begin\_fill() | 准备开始填充图形 |
| turtle.end\_fill() | 填充完成 |
| turtle.hideturtle() | 隐藏画笔的turtle形状 |
| turtle.showturtle() | 显示画笔的turtle形状 |

 (3)    全局控制命令

|  |  |
| --- | --- |
| 命令 | 说明 |
| turtle.clear() | 清空turtle窗口，但是turtle的位置和状态不会改变 |
| turtle.reset() | 清空窗口，重置turtle状态为起始状态 |
| turtle.undo() | 撤销上一个turtle动作 |
| turtle.isvisible() | 返回当前turtle是否可见 |
| stamp() | 复制当前图形 |
| turtle.write(s [,font=("font-name",font\_size,"font\_type")]) | 写文本，s为文本内容，font是字体的参数，分别为字体名称，大小和类型；font为可选项，font参数也是可选项 |

(4)    其他命令

|  |  |
| --- | --- |
| 命令 | 说明 |
| turtle.mainloop()或turtle.done() | 启动事件循环 -调用Tkinter的mainloop函数。  必须是乌龟图形程序中的最后一个语句。 |
| turtle.mode(mode=None) | 设置乌龟模式（“standard”，“logo”或“world”）并执行重置。如果没有给出模式，则返回当前模式。   |  |  |  | | --- | --- | --- | | 模式 | 初始龟标题 | 正角度 | | standard | 向右（东） | 逆时针 | | logo | 向上（北） | 顺时针 | |
| turtle.delay(delay=None) | 设置或返回以毫秒为单位的绘图延迟。 |
| turtle.begin\_poly() | 开始记录多边形的顶点。当前乌龟位置是多边形的第一个顶点。 |
| turtle.end\_poly() | 停止记录多边形的顶点。当前的乌龟位置是多边形的最后一个顶点。将与第一个顶点相连。 |
| turtle.get\_poly() | 返回最后记录的多边形。 |

3. 命令详解

        3.1 turtle.circle(radius, extent=None, steps=None)

        描述：以给定半径画圆

        参数：

        radius(半径)：半径为正(负)，表示圆心在画笔的左边(右边)画圆； extent(弧度) (optional)；

        steps (optional) (做半径为radius的圆的内切正多边形，多边形边数为steps)。

举例:

|  |  |
| --- | --- |
| 代码 | 程序运行结果 |
| import turtle as t  t.circle(50) # 整圆  t.circle(50, steps=3) #三角形  t.circle(120, 180) # 半圆  t.done() |  |

4．编程练习

（1）画一个半径为100的圆

|  |  |
| --- | --- |
| 代码 | 程序运行结果 |
| import turtle as t  t.circle(100)  t.done() |  |

（2）画一个半径为100的半圆

|  |  |
| --- | --- |
| 代码 | 程序运行结果 |
| import turtle as t  t.circle(100,180)  t.left(90)  t.forward(200)  t.done() |  |

**第四课：Python绘制五星红旗（一）**

1．画红色的长方形

import turtle as t

t.penup() #抬笔

t.goto(-200,200) #移动到（-200,200）位置

t.pendown() #落笔

t.begin\_fill() #开起填充模式

t.fillcolor("red") #设置填充色

t.pencolor("red") #设置画笔色

t.forward(400)

t.right(90)

t.forward(300)

t.right(90)

t.forward(400)

t.right(90)

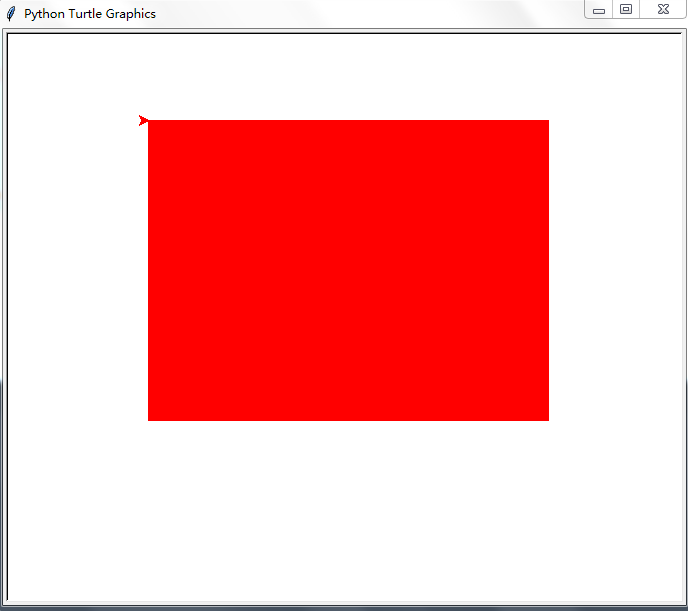
t.forward(300)

t.right(90)

t.end\_fill() # 结束填充

t.done()

程序运行结果：



2．画星星

import turtle as t

t.penup()

t.goto(-170, 145)

t.pendown()

t.begin\_fill()

t.fillcolor("yellow")

t.pencolor("yellow")

t.forward(50)

t.right(144)

t.forward(50)

t.right(144)

t.forward(50)

t.right(144)

t.forward(50)

t.right(144)

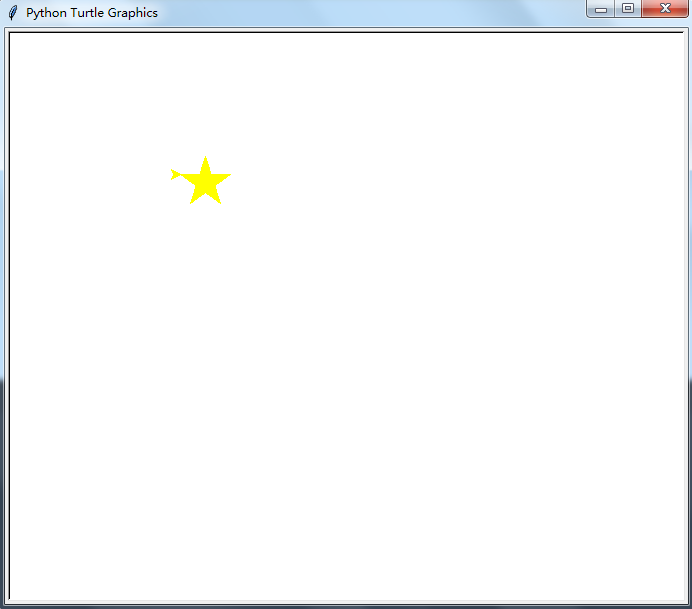
t.forward(50)

t.right(144)

t.end\_fill()

t.done()

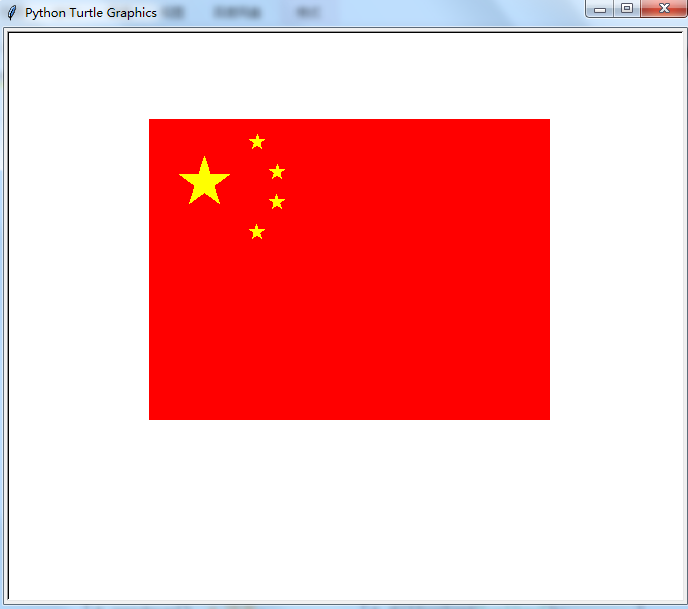
程序运行结果：



**第五课：Python绘制五星红旗（二）**

Python绘制五星红旗的四颗小星星

（1）程序运行结果：



（2）参考程序：

|  |  |
| --- | --- |
| import turtle as t  #画长方形  t.penup()  t.goto(-200, 200) # 移动到（-200,200）位置  t.pendown() # 落笔  t.begin\_fill() # 开起填充模式  t.fillcolor("red") # 设置填充色  t.pencolor("red") # 设置画笔色  t.forward(400)  t.right(90)  t.forward(300)  t.right(90)  t.forward(400)  t.right(90)  t.forward(300)  t.right(90)  t.end\_fill() # 结束填充 | #画大星星  t.penup()  t.goto(-170, 145)  t.pendown()  t.begin\_fill()  t.fillcolor("yellow")  t.pencolor("yellow")  t.forward(50)  t.right(144)  t.forward(50)  t.right(144)  t.forward(50)  t.right(144)  t.forward(50)  t.right(144)  t.forward(50)  t.right(144)  t.end\_fill() |

|  |  |
| --- | --- |
| #第一颗小星星  t.penup()  t.goto(-100, 180)  t.pendown()  t.begin\_fill()  t.fillcolor("yellow")  t.pencolor("yellow")  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.end\_fill() | #第二颗小星星  t.penup()  t.goto(-80, 150)  t.pendown()  t.begin\_fill()  t.fillcolor("yellow")  t.pencolor("yellow")  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.end\_fill() |
| #第三颗小星星  t.penup()  t.goto(-80, 120)  t.pendown()  t.begin\_fill()  t.fillcolor("yellow")  t.pencolor("yellow")  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.end\_fill() | #第四颗小星星  t.penup()  t.goto(-100, 90)  t.pendown()  t.begin\_fill()  t.fillcolor("yellow")  t.pencolor("yellow")  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.forward(15)  t.right(144)  t.end\_fill()  t.hideturtle()  t.done() |

**第六课：Python绘制螺旋图形（一）**

1. 正方形螺旋图

（1）代码：

import turtle as t

#for循环，x从0增加到200

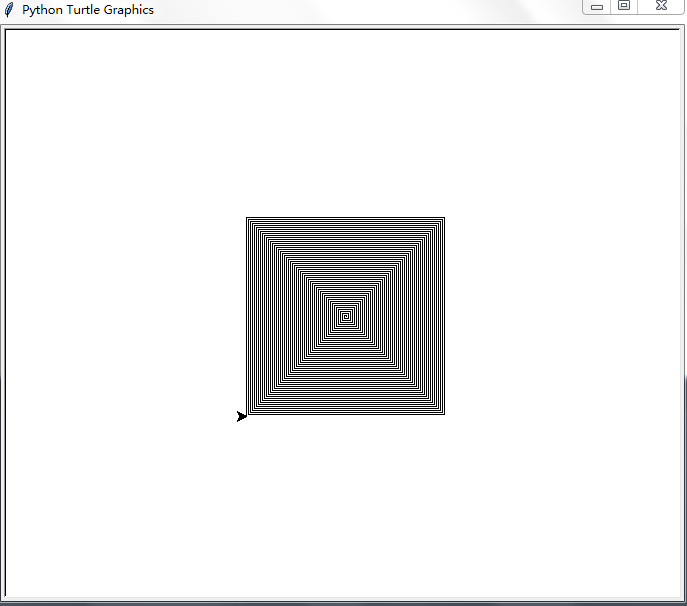
for x in range(200):

t.forward(x)

t.left(90)

t.done()

（2）程序运行结果



1. 螺旋线楼梯图

（1）代码：

import turtle as t

#for循环，x从0增加到200

for x in range(200):

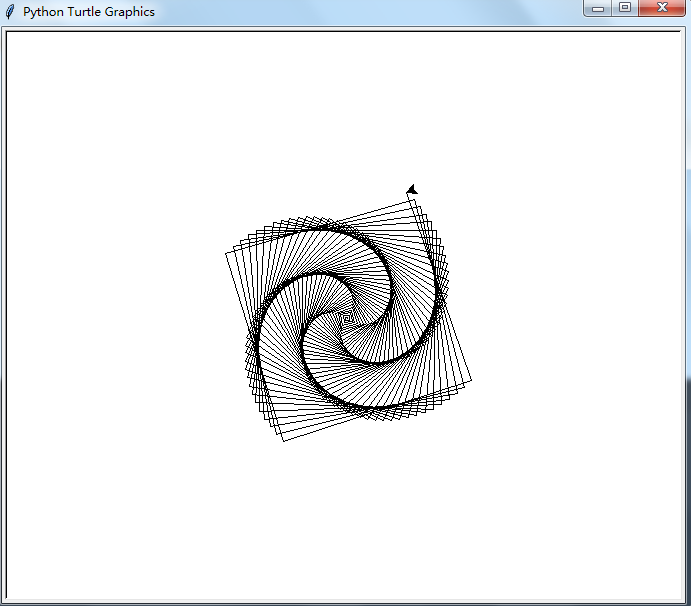
t.forward(x)

#参数修改，角度变化

t.left(91)

t.done()

（2）程序运行结果



1. 6个螺旋线的圆

（1）代码：

import turtle as t

#for循环，x从0增加到200

for x in range(100):

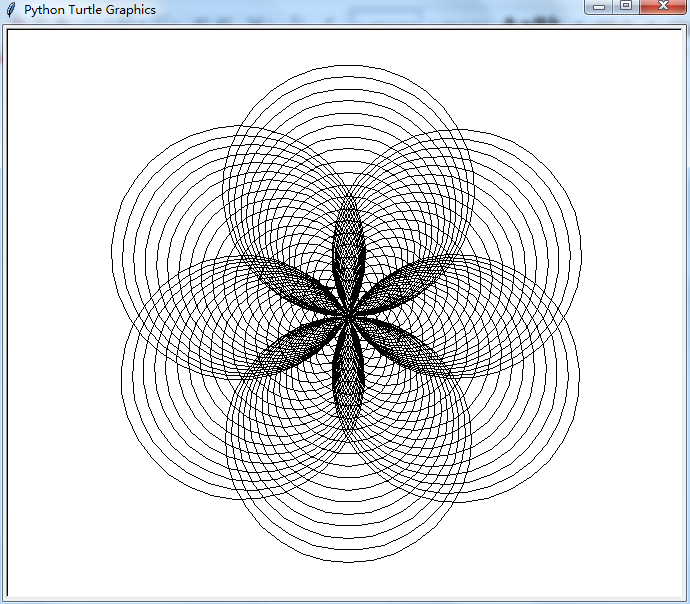
t.circle(x)

#参数修改，角度变化

t.left(60)

t.done()

（2）程序运行结果



**第七课：Python绘制螺旋图形（二）**

1. 彩色的正方形螺旋图

（1）代码：

import turtle as t

#添加颜色

t.pencolor("red")

#for循环，x从0增加到200

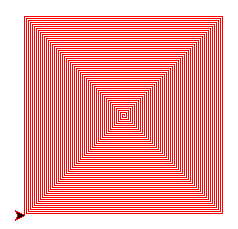
for x in range(200):

t.forward(x)

t.left(90)

t.done()

（2）程序运行结果



1. 彩色的螺旋线楼梯图

（1）代码：

import turtle as t

#创建一个名为colors的列表变量，并在列表中放置四个颜色

colors=["red","yellow","blue","green"]

#for循环，x从0增加到200

for x in range(100):

t.pencolor(colors[x%4])

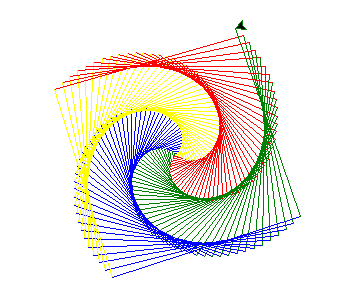
t.forward(x)

#参数修改，角度变化

t.left(91)

t.done()

（2）程序运行结果



1. 6个彩色的螺旋线的圆

（1）代码：

import turtle as t

#创建一个名为colors的列表变量，并在列表中放置六个颜色

colors = ["red", "yellow", "blue", "green","pink","black"]

#for循环，x从0增加到100

for x in range(100):

t.pencolor(colors[x%6])

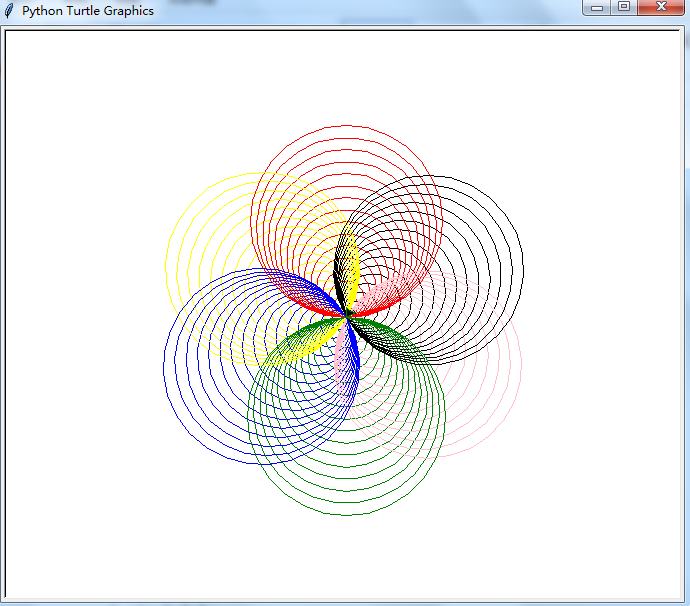
t.circle(x)

#参数修改，角度变化

t.left(60)

t.done()

（2）程序运行结果



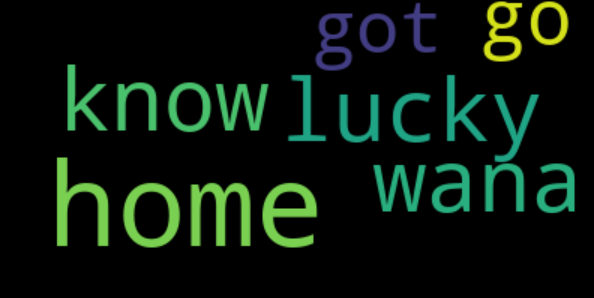
**第八课：python简单的词云图**

1. 什么是词云图

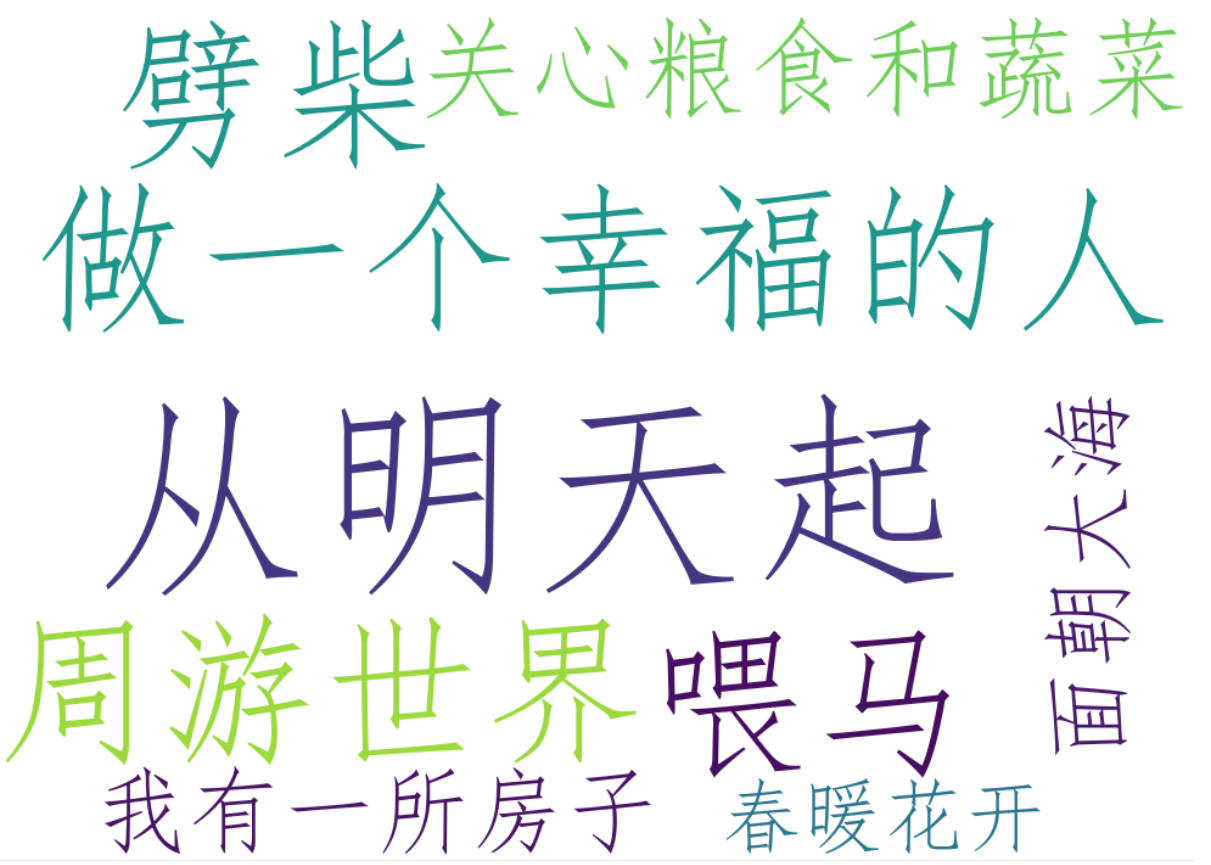
词云，是数据可视化的一种形式。

“词云”就是对网络文本中出现频率较高的“关键词”予以视觉上的突出，形成“关键词云层”或“关键词渲染”，从而过滤掉大量的文本信息，使浏览网页者只要一眼扫过文本就可以领略文本的主旨。

1. 词云图案例展示
2. 最简单的中英文词云图



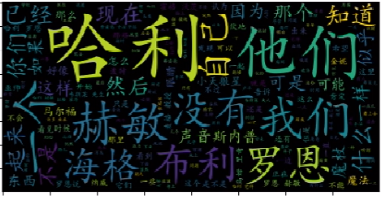
1. 最简单的中英文词云图



1. 读取英文文档中的词云图

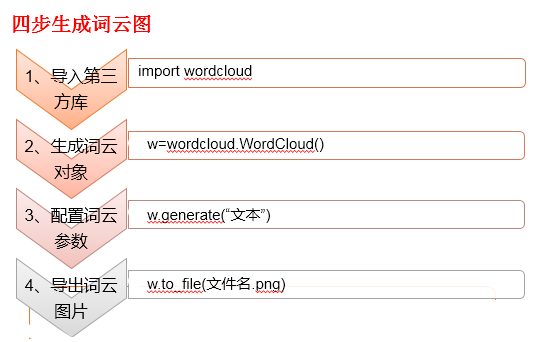


1. 读取中文文档中的词云图



1. 简单的词云图案例分析

**编程思路：**



**Wordcloud库：**

wordcloud意为“词云”，它可以生成词云图，对文本中出现频率较高的“关键词”予以视觉化的展现，直观展示文本关键内容及主旨。在wordcloud模块中，WordCloud对象，也就是“词云”对象。

1. 简单的英文词云图



**程序代码：**

#导入词云库

import wordcloud

#创建词云对象

w = wordcloud.WordCloud()

#设置词云图参数

w.generate('hello,welcome to python world')

#导出词云图片

w.to\_file('txt.png')

1. 简单的中文词云图



程序准备：simfang.ttf字体 

**程序代码：**

#导入词云库

import wordcloud

#创建词云对象，设置背景颜色、宽高、字体参数

w = wordcloud.WordCloud(width=1000,height=600,background\_color="pink",font\_path="simfang.ttf")

#设置词云图参数

w.generate('大家好，欢迎来到python世界')

#导出词云图片

w.to\_file('中文.png')

**第九课：python复杂的词云图（一）**

读取英文文档中的词云图

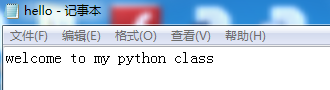
**Matplotlib库：**

Matplotlib是Python最常用的可视化工具，它可以被用来生成直方图、折线图、散点图等数据图表，是数据分析领域常见的小帮手。使用Matplotlib库中的pyplot 子模块

图形由plot() 函数绘制。图形由 show() 函数显示。

1．课前准备资料：

（1）Txt文档：hello.txt

（2）文档内容：

2．程序代码：

**#导入词云库**

import wordcloud

**#导入绘图库的pyplot子模块**

from matplotlib import pyplot as plt

**#创建词云图对象**

w = wordcloud.WordCloud(height=700, width=1000, background\_color="skyblue")

**#读取文件路径**

path = "hello.txt"

**#读取文件内容**

txt = open(path, 'r').read()

**#设置词云图参数**

nw = w.generate(txt)

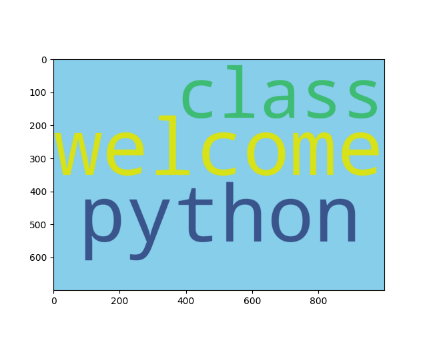
**#使用plt的imshow方法将词云图生成热图**

plt.imshow(nw)

**#显示词云图**

plt.show()

1. 程序运行结果：



**第十课：python复杂的词云图（二）**

读取中文文档中的词云图

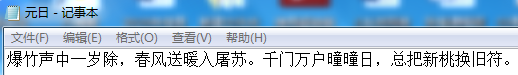
**Matplotlib库：**

Matplotlib是Python最常用的可视化工具，它可以被用来生成直方图、折线图、散点图等数据图表，是数据分析领域常见的小帮手。使用Matplotlib库中的pyplot 子模块

图形由plot() 函数绘制。图形由 show() 函数显示。

1．课前准备资料：

（1）Txt文档：元日.txt

（2）文档内容：

2．程序代码：

#导入词云库

import wordcloud

#导入绘图库的pyplot子模块

from matplotlib import pyplot as plt

#创建词云图对象，设置字体属性

w=wordcloud.WordCloud(height=700, width=1000, background\_color="skyblue",font\_path='simfang.ttf')

#读取文件路径

path = "元日.txt"

#读取文件内容

txt = open(path, 'r', encoding="utf-8").read()

#设置词云图参数

nw = w.generate(txt)

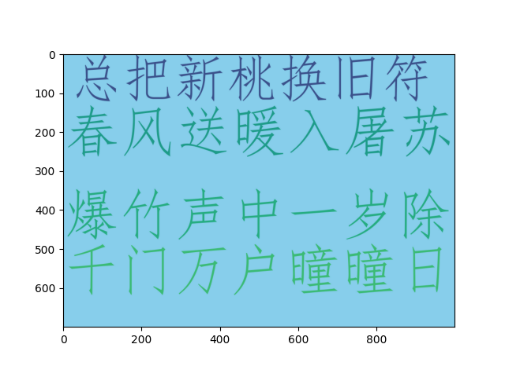
#使用plt的imshow方法将词云图生成热图

plt.imshow(nw)

#显示词云图

plt.show()

3．程序运行结果：



**第十一课：python复杂的词云图（三）**

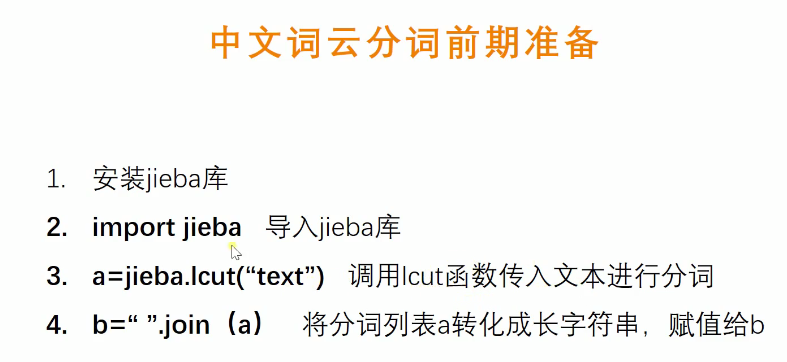
思考：以下两张图片有什么不同？



图1 图2

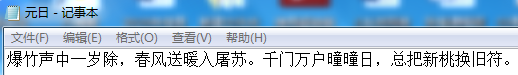
**jieba库：**

jieba是一个中文分词库，它可以把中文句子划分成词组，以便进一步进行自然语言处理。



1．课前准备资料：

（1）Txt文档：元日.txt

（2）文档内容：

2．程序代码：

#导入分词库

import jieba

#导入词云库

import wordcloud

#导入绘图库的pyplot子模块

from matplotlib import pyplot as plt

#创建词云图对象，设置字体属性

w = wordcloud.WordCloud(height=700, width=1000,background\_color="skyblue", font\_path='simfang.ttf')

#读取文件路径

path = "元日.txt"

#读取文件内容

txt=open(path, 'r', encoding="utf-8").read()

#对text文本进行分词，并且保存到text中。

txt=jieba.lcut(txt)

txt=' '.join(txt)

print(txt)

#设置词云图参数

nw = w.generate(txt)

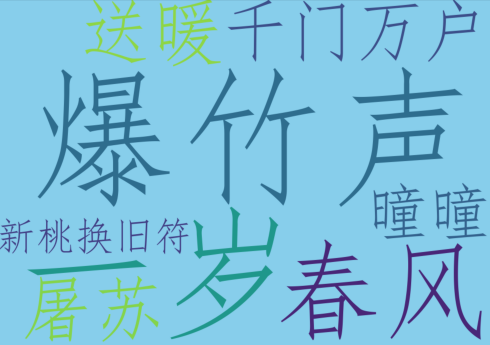
#使用plt的imshow方法将词云图生成热图

plt.imshow(nw)

#显示词云图

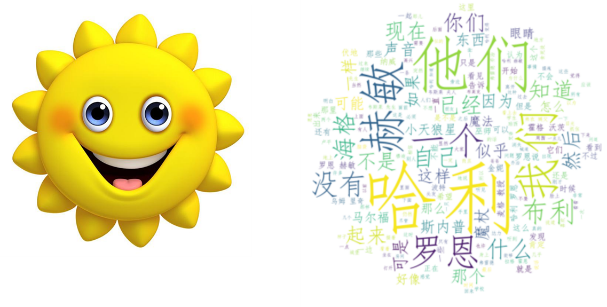
plt.show()

3．程序运行结果：

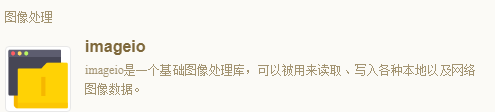


**第十二课：python作品创作**

1. 做出有形状的词云图

1. 如何做出有形状的词云？



Io就是input和output的意思，专门用来处理图像的输入输出。照片一定是白底才能够被程序识别，读取这个图片形状。

1. 代码

#导入分词库

import jieba

#导入imageio库

import imageio

#使用imread()方法读取图片形状

mk=imageio.imread("2.jpg")

#导入词云库

import wordcloud

#导入绘图库的pyplot子模块

from matplotlib import pyplot as plt

#创建词云图对象，设置字体属性

w = wordcloud.WordCloud(mask=mk,background\_color="white", font\_path='simfang.ttf')

#读取文件路径

path = "名言.txt"

#读取文件内容

txt=open(path, 'r', encoding="utf-8").read()

#对text文本进行分词，并且保存到text中。

txt=jieba.lcut(txt)

txt=' '.join(txt)

print(txt)

#设置词云图参数

nw = w.generate(txt)

#使用plt的imshow方法将词云图生成热图

plt.imshow(nw)

#显示词云图

plt.show()

1. 程序运行结果



1. 学生创意编程

**Python作品类型**

* 数字艺术类：通过程序生成和展示视觉艺术，具备创意、美感和互动性。
* 互动游戏类：各种竞技类、探险类、角色扮演类、球类、棋牌类游戏等。
* 实用工具类：有实用价值、能解决学习生活中的实际问题，提高学习工作效率的程序应用工具。
* 科学探索类：数学对象可视化、现实过程模拟仿真、科学实验等各学科的趣味性展示与探究。

**Python作品要求**

* 作品原创。作品可借鉴已有程序作品，但必须体现创作者的思考和创新。
* 艺术展现。作品充分展现计算机图形与计算机艺术特色，创意巧妙独特，表现形式丰富。作品合理运用图形与色彩，创造愉悦审美感受。
* 程序技术。作品的绘制过程流畅，富有创意。作品的交互设计简单明了，体验良好。作品内容主题清晰，易于理解。
* 交互体验。程序能够正常运行，运行过程稳定、流畅、高效，无明显错误；程序结构划分合理，代码编写规范，清晰易读；巧妙利用计算思维与算法，创造独特创意体验。