第一次上机作业

18029100040

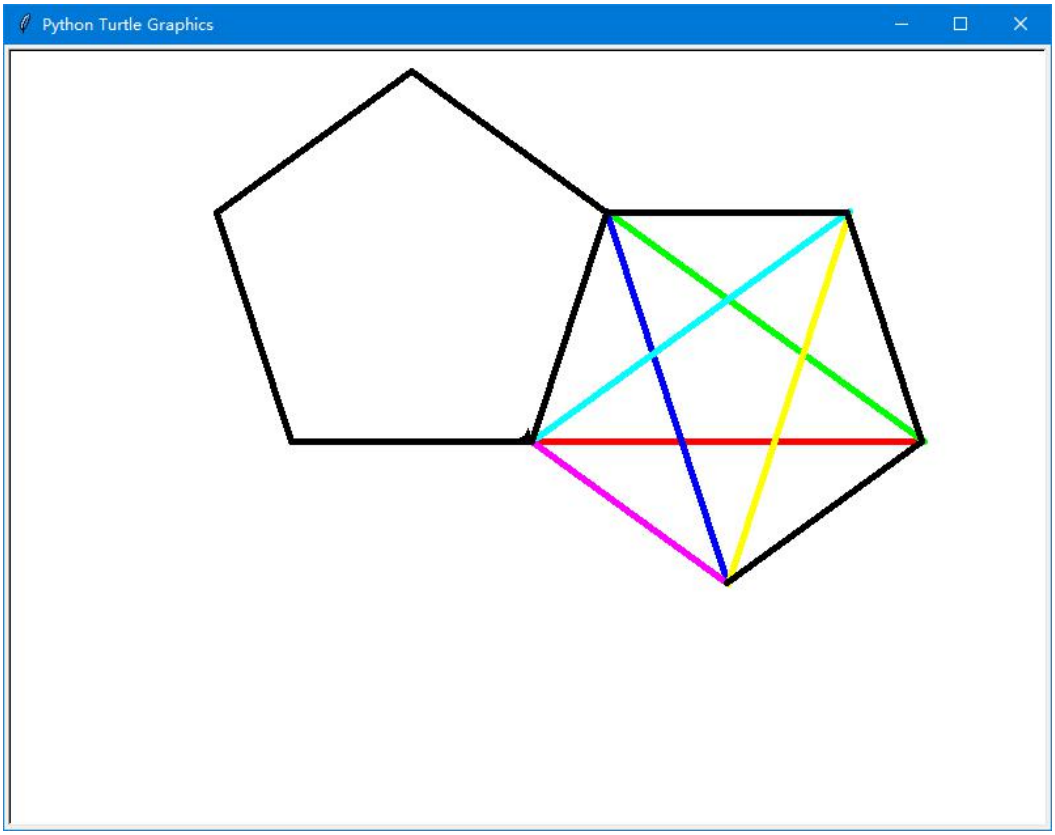
吴程锴

一、作业 11：自动轨迹绘制

1.1 代码

```
#AutoTrace.py
import turtle
turtle.setup(800,600)
turtle.pencolor("red")
turtle.pensize(5)
turtle.speed(5)

data = []
f = open("data.txt","r") #打开文件
for line in f:
    line = line.replace('\n','')
    data.append(list(map(eval,line.split(','))))
f.close()

for i in range(len(data)):
    turtle.pencolor(data[i][3],data[i][4],data[i][5])
    turtle.fd(data[i][0])
    turtle.left(data[i][2]) if data[i][1] == 0 else turtle.right(data[i][2])
turtle.done
```

1.2 结果

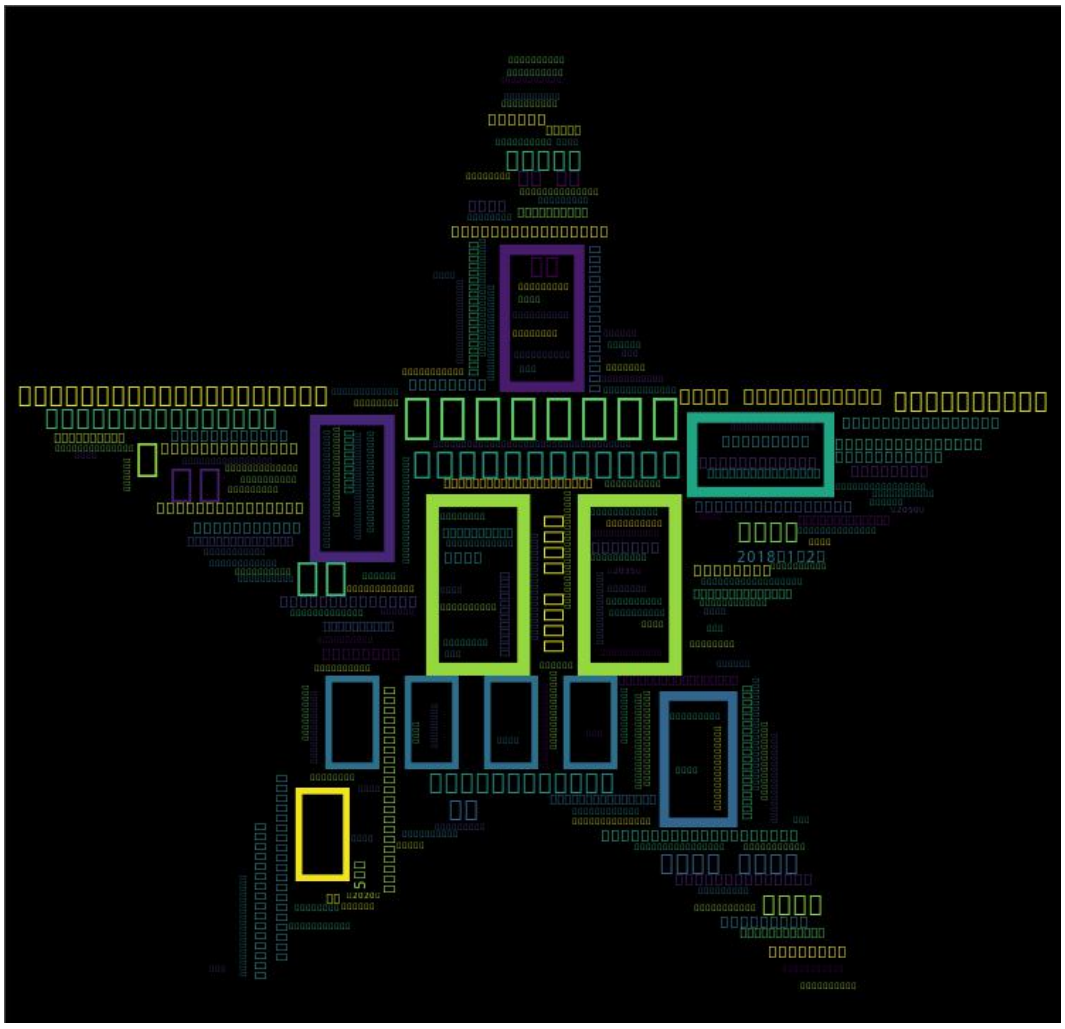


二、作业 12：词云处理

2.1 代码

```
    from wordcloud import WordCloud
import PIL.Image as image
import numpy

with open("关于实施乡村振兴战略的意见.txt",encoding='utf-8') as fp:
    text = fp.read()
    mask = numpy.array(image.open("fivestart.png"))
    wordcloud = WordCloud(mask = mask).generate(text)
    image_produce = wordcloud.to_image()
        image_produce.show()
```

2.2 结果



三、作业 13：模拟比赛

3.1 代码

```
from random import random
import math

rate_A = eval(input("A 的胜率(%):"))
rate_B = eval(input("B 的胜率(%):"))
times = eval(input("次数:"))
win_A = 0
win_B = 0

for i in range(times):
    rate = random() * 100
    if rate < rate_A:
```

```
        win_A = win_A + 1
    else:
        win_B = win_B + 1
    print("A 获胜次数:{0}     B 获胜次数:{1}    A 与 B 获胜比值:{2}".format(win_A,win_B,win_A/win_B))
```

3.2 结果

```
A的胜率(%):40
B的胜率(%):60
次数:100
A获胜次数:41      B获胜次数:59    A与B获胜比值:0.6949152542372882
```

四、作业 14：第三方库安装脚本

4.1 代码

```
import os
list = ["numpy","matplotlib","pillow","sklearn","requests","jieba","beautifulsoup4"]

try:
    for lib in list:
        os.system("pip install" + lib)
    print("Successful!")
except:
        print("Failed!")
```

4.2 结果

```
Successful!
```

五、作业 15：玫瑰花绘制

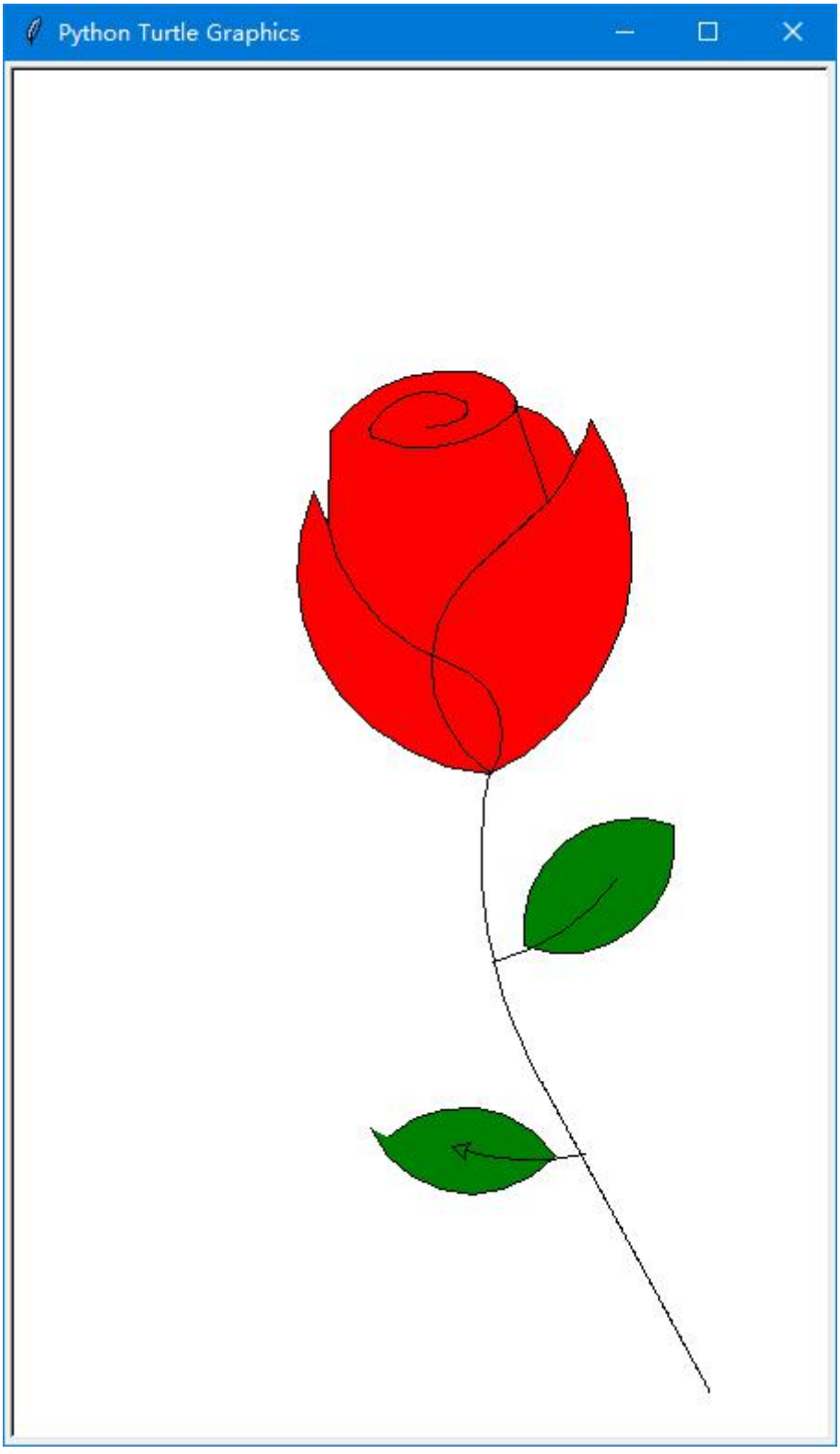
5.1 代码

```
#RoseDraw.py
import turtle as t
# 定义一个曲线绘制函数
def DegreeCurve(n, r, d=1):
    for i in range(n):
```

```
        t.left(d)
        t.circle(r, abs(d))
# 初始位置设定
s = 0.2 # size
t.setup(450*5*s, 750*5*s)
t.pencolor("black")
t.fillcolor("red")
t.speed(100)
t.penup()
t.goto(0, 900*s)
t.pendown()
# 绘制花朵形状
t.begin_fill()
t.circle(200*s,30)
DegreeCurve(60, 50*s)
t.circle(200*s,30)
DegreeCurve(4, 100*s)
t.circle(200*s,50)
DegreeCurve(50, 50*s)
t.circle(350*s,65)
DegreeCurve(40, 70*s)
t.circle(150*s,50)
DegreeCurve(20, 50*s, -1)
t.circle(400*s,60)
DegreeCurve(18, 50*s)
t.fd(250*s)
t.right(150)
t.circle(-500*s,12)
t.left(140)
t.circle(550*s,110)
t.left(27)
t.circle(650*s,100)
t.left(130)
t.circle(-300*s,20)
t.right(123)
t.circle(220*s,57)
t.end_fill()
# 绘制花枝形状
t.left(120)
t.fd(280*s)
t.left(115)
t.circle(300*s,33)
t.left(180)
t.circle(-300*s,33)
DegreeCurve(70, 225*s, -1)
```

```
t.circle(350*s,104)
t.left(90)
t.circle(200*s,105)
t.circle(-500*s,63)
t.penup()
t.goto(170*s,-30*s)
t.pendown()
t.left(160)
DegreeCurve(20, 2500*s)
DegreeCurve(220, 250*s, -1)
# 绘制一个绿色叶子
t.fillcolor('green')
t.penup()
t.goto(670*s,-180*s)
t.pendown()
t.right(140)
t.begin_fill()
t.circle(300*s,120)
t.left(60)
t.circle(300*s,120)
t.end_fill()
t.penup()
t.goto(180*s,-550*s)
t.pendown()
t.right(85)
t.circle(600*s,40)
# 绘制另一个绿色叶子
t.penup()
t.goto(-150*s,-1000*s)
t.pendown()
t.begin_fill()
t.rt(120)
t.circle(300*s,115)
t.left(75)
t.circle(300*s,100)
t.end_fill()
t.penup()
t.goto(430*s,-1070*s)
t.pendown()
t.right(30)
t.circle(-600*s,35)
    t.done()
```

5.2 结果

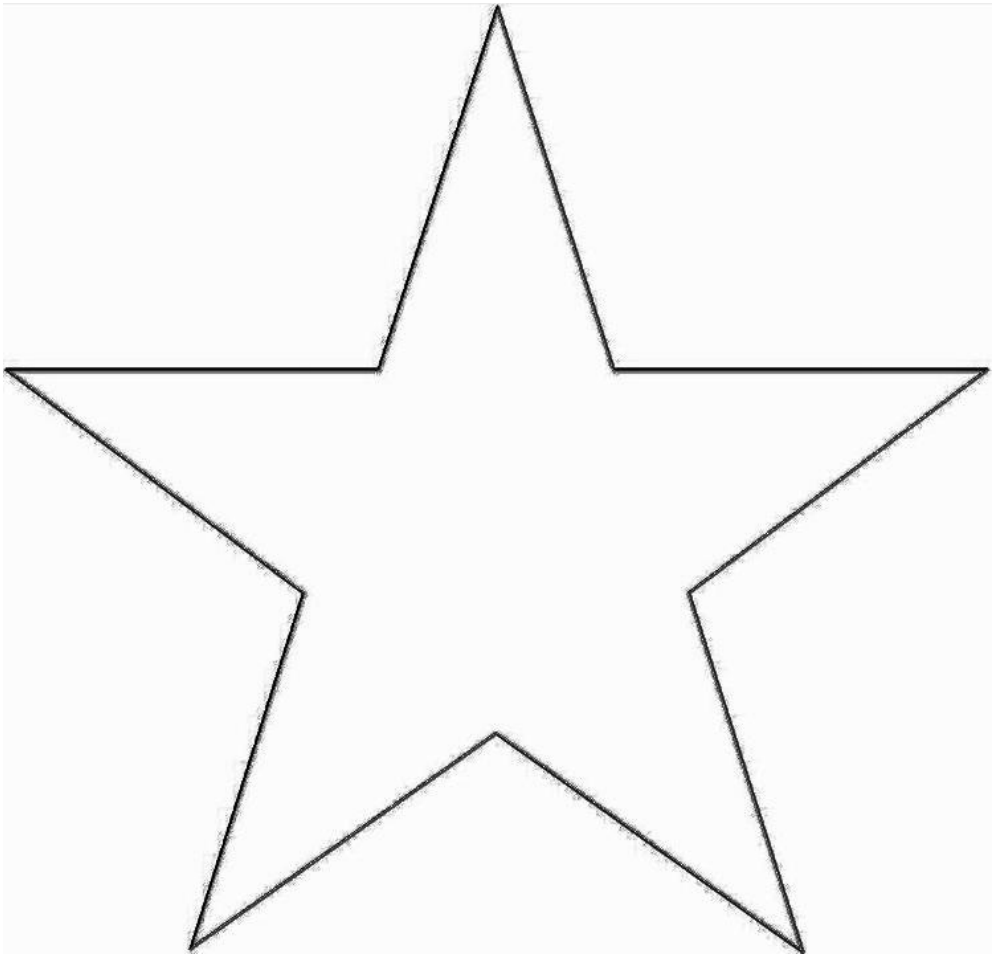


六、作业 16：图像转手绘效果

6.1 代码

```
from PIL import Image
import numpy as np

a = np.asarray(Image.open(r"fivestart.png").convert('L')).astype('float')

depth = 10.
grad = np.gradient(a)  # 梯度值，
grad_x, grad_y = grad
grad_x = grad_x * depth / 100.  # 列梯度值*0.1
grad_y = grad_y * depth / 100.

A = np.sqrt(grad_x ** 2 + grad_y ** 2 + 1.)  # 相当于 grad_z=1
uni_x = grad_x / A
uni_y = grad_y / A
uni_z = 1. / A  # 梯度归一化

vec_el = np.pi / 2.2
vec_az = np.pi / 4.

dx = np.cos(vec_el) * np.cos(vec_az)
dy = np.cos(vec_el) * np.sin(vec_az)
dz = np.sin(vec_el)  # 长度为 1，投影 x,y,z 长度

b = 255 * (dx * uni_x + dy * uni_y + dz * uni_z)
b = b.clip(0, 255)

im = Image.fromarray(b.astype('uint8'))
im.save(r"fivestart_Draw.jpg")
```

6.2 结果



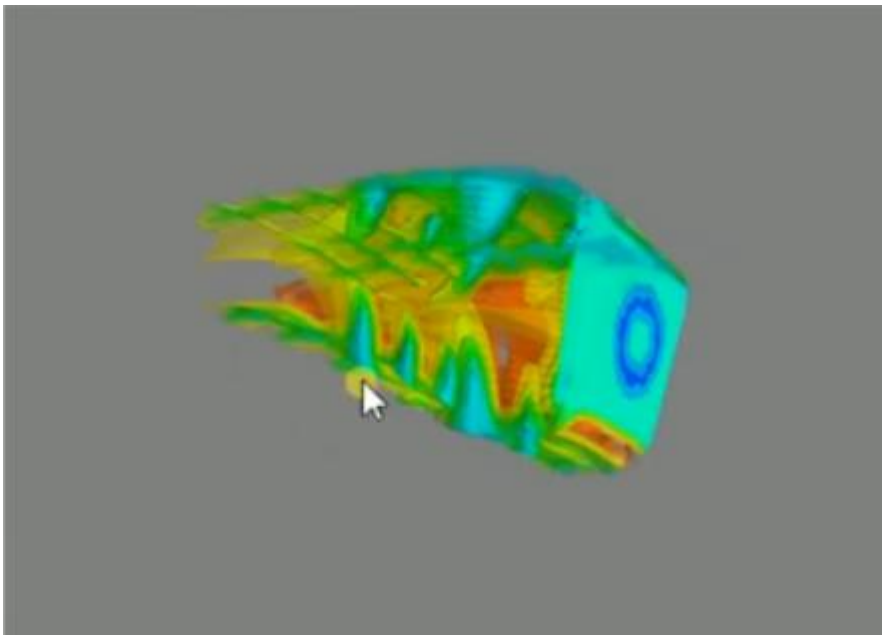
七、作业 17：标量数据绘制

7.1 代码

```
    from tvtk.api import tvtk
from tvtkfunc import ivtk_scene,event_loop

plot3d = tvtk.MultiblockPLOT3DReader(
    xyz_file_name = "combxyz.bin",
    q_file_name = "combq.bin",
    scalar_function_number = 100,
    vector_function_number = 200
)
plot3d.update()
grid = plot3d.output.get_block(0)

con = tvtk.ContourFilter()
con.set_input_data(grid)
```

```
con.generate_values(10,grid.point_data.scalars.range)
m = tvtk.PolyDataMapper(scalar_range = grid.point_data.scalars.range,
                        input_connection = con.output_port)
a = tvtk.Actor(mapper = m)
a.property.opacity = 0.5
win = ivtk_scene(a)
win.scene.isometric_view()
    event_loop()
```

7.2 结果



八、作业 18：矢量数据可视化

8.1 代码

```
    from tvtk.api import tvtk
from Tvtkfunc import ivtk_scene,event_loop

def read_data():    #导入数据
    plot3d = tvtk.MultiBlockPLOT3DReader(
        xyz_file_name="comxyz.bin", #网格文件
        q_file_name="combq.bin", #开启动力学结果文件
        scalar_function_number = 100, #设置标量数据数量
        vector_function_number=200, #设置矢量数据数量
    )   #读入 Plot3D 数据
    plot3d.update() #让 plot3D 计算器输出数据
    return plot3d
```

```
13.
14.
15. plot3d = read_data()
16. grid = plot3d.output.get_block(0)   #获取读入的数据集对象
17.
18. #对数据集中的数据进行随机选取，每 50 个点选择一个点,是对数据进行降采样
19. mask = tvtk.MaskPoints(random_mode=True,on_ratio=50)
20. mask.set_input_data(grid)   #将 grid 和 mask 相连
21. #创建表示箭头的 PolyData 数据集
22. glyph_source = tvtk.ArrowSource()
23. #在 Mask 采样后的 PolyData 数据集每个点上放置一个箭头
24. #箭头的方向（速度方向），长度<箭头越大，表示标量越大>和颜色<也表示标量大小，红色小，蓝色大>
    （两个都表示密度）由于点对应的矢量和标量数据决定
25.
26. #将上面的降采样数据与箭头符号化相关联
27. glyph = tvtk.Glyph3D(input_connection=mask.output_port,
28.                       scale_factor=4)    #scale_factor 符号的共同放缩系数
29. glyph.set_source_connection(glyph_source.output_port)
30.
31. m = tvtk.PolyDataMapper(scalar_range=grid.point_data.scalars.range, #设置映射器的变量
    范围属性
32.                         input_connection=glyph.output_port)
33. a = tvtk.Actor(mapper=m)
34. a.property.opacity = 0.5    #设置透明度为 0.5
35.
36. win = ivtk_scene(a)
37. win.scene.isometric_view()
38.   event_loop()
```

8.2 结果