9.3 实例："自动轨迹绘制"

问题分析

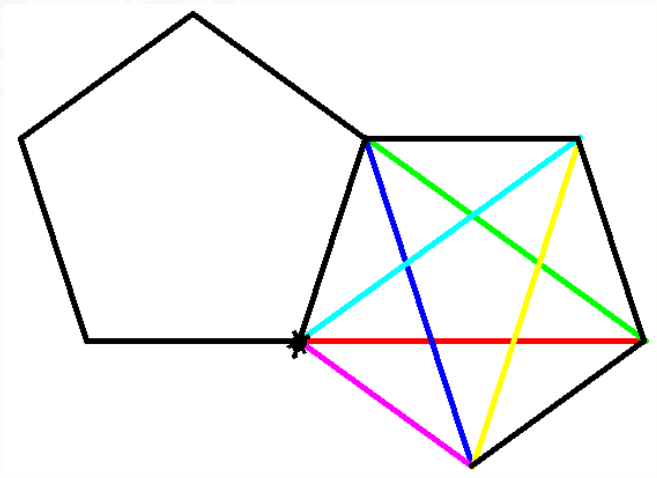
自动轨迹绘制

- 需求：根据脚本来绘制图形?
- 不通过写代码而通过写数据绘制轨迹
- 数据脚本是自动化最重要的第一步

问题分析

自动轨迹绘制

```
300,0,144,1,0,0
300,0,144,0,1,0
300,0,144,0,0,1
300,0,144,1,1,0
300,0,108,0,1,1
184,0,72,1,0,1
```



"自动轨迹绘制"实例讲解

自动轨迹绘制

基本思路

- 步骤1：定义数据文件格式（接口）
- 步骤2：编写程序，根据文件接口解析参数绘制图形
- 步骤3：编制数据文件

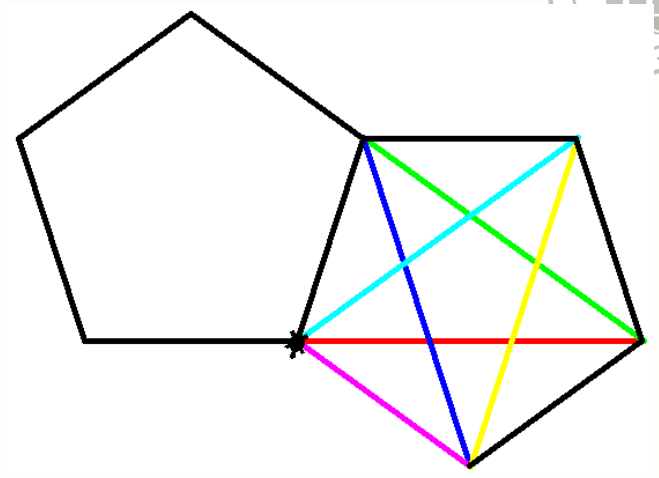
数据接口定义

非常具有个性色彩

300,0,144,1,0,0

300,1,144,0,1,0

```
#AutoTraceDraw.py
import turtle as t
t.title('自动轨迹绘制')
t.setup(800, 600, 0, 0)
t.pencolor("red")
t.pensize(5)
#数据读取
datals = []
f = open("data.txt")
for line in f:
    line = line.replace("\n","")
    datals.append(list(map(eval, line.split(","))))
f.close()
#自动绘制
for i in range(len(datals)):
    t.pencolor(datals[i][3],datals[i][4],datals[i][5])
    t.fd(datals[i][0])
    if datals[i][1]:
        t.right(datals[i][2])
    else:
        t.left(datals[i][2])
```



PY01B32 重民食

数据文件

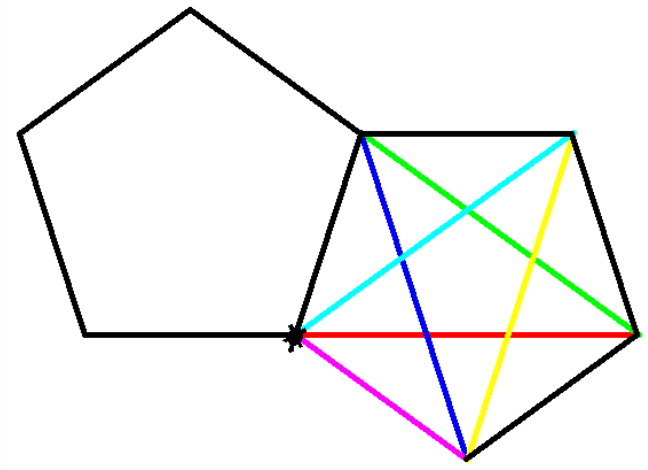
```
300,0,144,1,0,0
300,0,144,0,1,0
300,0,144,0,0,1
300,0,144,1,1,0
300,0,108,0,1,1
184,0,72,1,0,1
184,0,72,0,0,0
```

```
184,0,72,0,0,0
184,0,72,0,0,0
184,1,72,1,0,1
184,1,72,0,0,0
184,1,72,0,0,0
184,1,72,0,0,0
184,1,72,0,0,0
184,1,720,0,0,0
```

data.txt

"自动轨迹绘制"举一反三

```
import turtle as t
t.title('自动轨迹绘制')
t.setup(800, 600, 0, 0)
t.pencolor("red")
t.pensize(5)
datals = []
f = open("data.txt")
for line in f:
    line = line.replace("\n","")
    datals.append(list(map(eval, line.split(","))))
f.close()
for i in range(len(datals)):
    t.pencolor(datals[i][3],datals[i][4],datals[i][5])
    t.fd(datals[i][0])
    if datals[i][1]:
        t.right(datals[i][2])
    else:
        t.left(datals[i][2])
```



举一反三

理解方法思维

- 自动化思维：数据和功能分离，数据驱动的自动运行
- 接口化设计：格式化设计接口，清晰明了
- 二维数据应用：应用维度组织数据，二维数据最常用

举一反三

应用问题的扩展

- 扩展接口设计，增加更多控制接口
- 扩展功能设计，增加弧形等更多功能
- 扩展应用需求，发展自动轨迹绘制到动画绘制