

turtle 库

turtle 库诞生于 1966 年,是基于 LOGO 编程语言的图形绘制函数库,最初的使用对象主要是中小學生。由于其简单直观、容易掌握,后来被 Python 引入,成为 Python 的一个标准库。

5.1 运行环境设置

使用 turtle 库绘制图形的过程如下:首先设置画布的大小,一只小乌龟(画笔)在其中按照坐标爬行,其爬行轨迹形成了绘制图形。对于小乌龟而言,有前进、后退、旋转等爬行行为,其爬行方向包括前进、后退、左侧、右侧。

刚开始绘制时,小乌龟位于画布的正中央,此处的坐标为(0,0),画布就是 turtle 库中设置的绘图区域,可以定义它的大小和初始位置。具体设置格式有以下两种形式(单独、同时使用均可):

- turtle.screensize(width,height,bg): 参数分别为画布的宽、高、背景颜色。其中,宽和高的单位均为像素。
- turtle.setup(width,height,startx,starty): 参数 width 和 height 表示宽和高,为整数时,表示像素;为小数时,表示占据计算机屏幕的比例。参数 startx 和 starty 表示距离屏幕左上角顶点的横向和纵向距离,如果为空,则窗口位于屏幕中心。

例如:

```
turtle.screensize(800,600,"green")
```

表示设置宽 800 像素、高 600 像素、背景色为绿色的画布,位置在屏幕中央。其中常用的背景色有 white(白色)、black(黑色)、grey(灰色)、darkgreen(深绿色)、gold(金色)、violet(淡紫色)、purple(紫色)、red(红色)、blue(蓝色)等。

又如:

```
turtle.setup(width=400,height=300, startx=200, starty=100)
```

或

```
turtle.setup(400, 300, 200, 100)
```

表示设置宽 400 像素、高 300 像素的画布,其左上角距离屏幕左上角顶点的横向和纵向距离分别为 200 像素和 100 像素。

5.2 画笔设置

5.2.1 画笔基本参数

- `turtle.pensize(width)`: 设置画笔的宽度,单位是像素。
- `turtle.pencolor(color)`: 设置画笔颜色。如果没有参数传入,则返回当前画笔颜色。可以是字符串如 `green`、`red`,也可以是 RGB 三元组。
- `turtle.penup()`: 抬起画笔,之后移动画笔不绘制图形。
- `turtle.pendown()`: 落下画笔,之后移动画笔将绘制图形。
- `turtle.speed(speed)`: 设置画笔移动速度。画笔移动的速度为 $[0,10]$ 范围内的整数,数字越大则移动速度越快。

5.2.2 画笔运动命令

- `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` 角度。
- `turtle.left(degree)`: 逆时针移动 `degree` 角度。
- `turtle.goto(x,y)`: 将画笔移动到坐标为 (x,y) 的位置。
- `turtle.circle()`: 画圆,半径为正(负),表示圆心在画笔的左边(右边)画圆。
- `turtle.setx()`: 将当前 X 轴移动到指定位置。
- `turtle.sety()`: 将当前 Y 轴移动到指定位置。
- `turtle.setheading(angle)`: 设置当前朝向为 `angle` 角度。
- `turtle.home()`: 设置当前画笔位置为原点,朝向东。
- `turtle.dot(r)`: 绘制一个指定直径和颜色的圆点。

需要注意绝对角度和相对角度的区别:绝对角度是相对于画布而言的,因为画布是静止不动的,所以以画布为中心构建的角度坐标系的角度的不会发生变化的,例如 90° 是指正北方向。`turtle.seth(angle)`函数中的角度就是绝对角度;相对角度是以画笔本身的方向为中心建立的角度坐标系的角度的,是时刻在变动的,如 `turtle.left(angle)`、`turtle.right(angle)`中的角度就是相对角度。

5.2.3 画笔控制命令

- `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。

5.3 其他命令

- `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,`font-name`、`font_size` 和 `font_type` 分别为字体名称、大小和类型。`font` 参数为可选项,`font-name`、`font_size` 和 `font_type` 也是可选项。
- `turtle.mainloop()`或 `turtle.done()`: 启动事件循环,调用 `turtle` 库的 `mainloop` 函数,必须是图形程序中的最后一个语句。
- `turtle.mode(mode=None)`: 设置画笔模式为 `standard`、`logo` 或 `world`,并执行重置。如果没有给出模式,则返回当前模式。其中,`standard` 表示向右(东)或逆时针,`logo` 表示向上(北)或顺时针。
- `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。当前的画笔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。当前的画笔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。

5.4 应用实例

【例 5-1】 绘制太阳花,如图 5-1 所示。

编程思路:如图 5-2 所示,从起点开始,默认方向是正东 0° ,第一次使画笔移动 200 像素,以当前角度向左(逆时针)方向偏移 170° ,并沿此方向使画笔第二次移动 200 像素,依此类推,循环 36 次回到原点,填充图形。

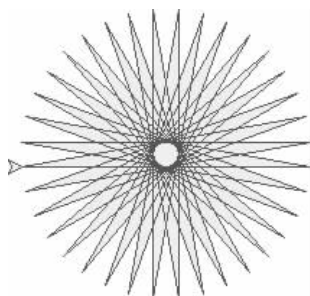


图 5-1 太阳花

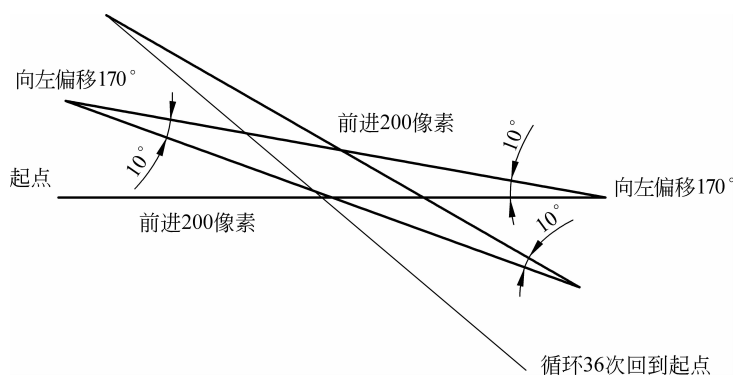


图 5-2 太阳花绘制原理

程序代码如下：

```
import turtle
turtle.color("red", "yellow")           #设置画笔和填充颜色
turtle.begin_fill()                     #开始
for i in range(36):                    #36次循环
    turtle.forward(200)                 #前进 200 像素
    turtle.left(170)                   #向左偏移 170°
turtle.end_fill()                      #填充
turtle.mainloop()                      #启动事件循环
```

【例 5-2】 绘制五角星,如图 5-3 所示。



图 5-3 五角星

程序代码如下：

```
import turtle
turtle.pensize(5)           #设置画笔宽度
turtle.pencolor("yellow")   #设置画笔颜色
turtle.fillcolor("red")     #设置填充颜色
turtle.begin_fill()
for i in range(5):
    turtle.forward(200)
    turtle.right(144)        #向右偏转 144°
turtle.end_fill()
turtle.penup()              #抬起画笔
turtle.goto(70,-140)        #定位
turtle.color("violet")       #设置颜色
turtle.write("Done", font= ('Arial', 20, 'normal')) #写文字
turtle.mainloop()
```

【例 5-3】 绘制旋转的正方形,如图 5-4 所示。

编程思路:如图 5-5 所示,首先从第一个起点开始移动画笔,向左偏转 90° ,循环 4 次,绘制完成第一个正方形。将画笔移动到第二个起点,向左偏转 45° 再绘制第二个正方形。这样循环 8 次即可。

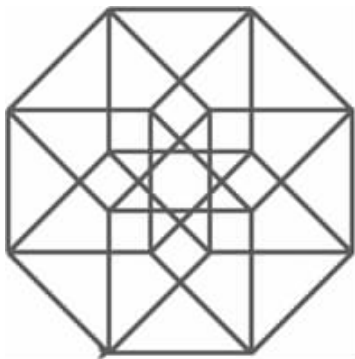


图 5-4 正方形的旋转

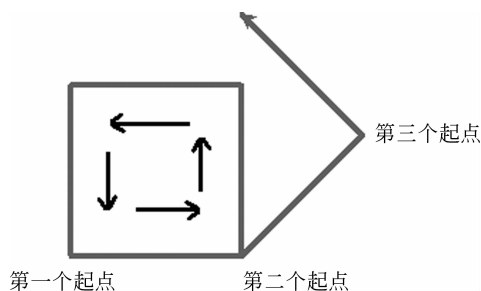


图 5-5 正方形的旋转编程思路

程序代码如下：

```
from turtle import *
d=120
setup(800,600,300,400)      #设置图形边界
color('red')
pensize(4)
for i in range(8):          #大循环 8 次
    for j in range(4):      #小循环 4 次
        forward(d)
        left(90)
    penup()
    forward(d)
    pendown()
    left(45)
```

【例 5-4】 绘制空心五角星,如图 5-6 所示。

编程思路:如图 5-7 所示,首先绘制第一个五角星,从第一个起点开始,初始角度 $A=0^\circ$,移动 $d=100$,再向左偏转 108° ,移动 $d_1=1.618d$,再向左偏转 144° ,移动 $d_1=1.618d$,完成第一个三角形的绘制,并填充。角度不变,抬起笔,沿着上次的角度前进 $d=100$,落下笔,第二个起点确定,使 $A=A+72^\circ$,重复第一次的画法。这样循环 5 次,完成图形的绘制。

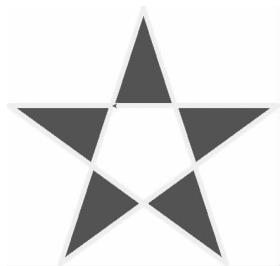


图 5-6 空心五角星

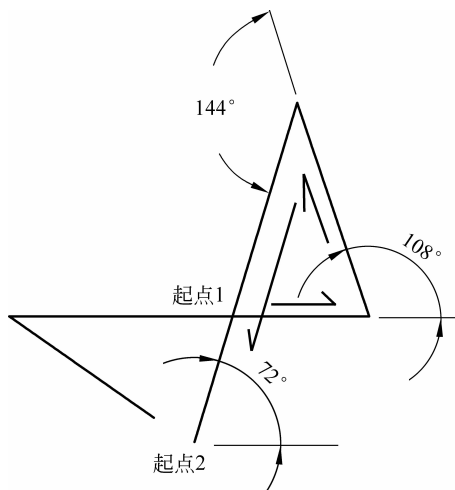


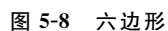
图 5-7 空心五角星的编程思路

程序代码如下:

```
from turtle import *
d=100
d1=d*1.618
setup(800,600,300,200)
pensize(8)
color('yellow','red')
A=0
for i in range(5):
    seth(A)          #设置绝对角度
    begin_fill()
    forward(d)        #前进 100 像素
    left(108)         #左偏转 108°
    forward(d1)       #前进 168 像素
    left(144)         #左偏转 144°
    forward(d1)       #前进 168 像素
    end_fill()        #填充
    up()              #抬笔
    forward(d)         #沿着最后的角度前进 100 像素
    down()             #笔落下
    A=A+72            #改变初始角度,5 次刚好 360°
```

【例 5-5】 绘制六边形,如图 5-8 所示。

编程思路:分为两个步骤,第 1 步参照例 5-4 五角星的代码画出 6 个三角形并填充,



```
from turtle import *
d=100
setup(800,600,300,400)
pensize(8)
color('yellow','red')
A=0
for i in range(6):
    seth(A)
    begin_fill()
    forward(d)
    left(120)
    forward(d)
    left(120)
    forward(d)
    end_fill()
    up()
    forward(d)
    down()
    A=A+60
#移动到中心
up()
left(120)
forward(d)
#调整方向
color('yellow','green')
down()
A=0
for i in range(6):
    seth(A)
    left(60)
    begin_fill()
```

```
forward(d)
left(120)
forward(d)
left(120)
forward(d)
end_fill()
A=A+60
```

习题 5

一、简答题

1. turtle 库有什么作用?
2. turtle 库的画布如何设置?
3. turtle 库的直线和圆如何绘制?
4. turtle 库如何进行图形填充?
5. turtle 库中如何书写文字?

二、选择题

1. 画笔抬起函数是()。
A. penup() B. pendown() C. pentop() D. pensize()
2. 画笔落下函数是()。
A. penup() B. pendown() C. pentop() D. pensize()
3. 画笔前进函数 forward 内的距离参数单位是()。
A. 厘米 B. 毫米 C. 英寸 D. 像素
4. 画布的默认原点(0,0)在画布的()。
A. 左上角 B. 右下角 C. 中心 D. 左下角
5. 画笔宽度设置函数是()。
A. penup() B. pensize() C. setup() D. pencolor
6. turtle.setheading(30)表示该点()。
A. 左前上方 30° B. 右前上方 30°
C. 左前下方 30° D. 右前下方 30°
7. turtle.left(30)表示相对当前方向()。
A. 顺时针改变 30° B. 逆时针改变 30°
C. 顺时针改变 60° D. 逆时针改变 60°
8. turtle.fillcolor(colorstring)表示()。
A. 绘制图形的边框颜色 B. 画布颜色
C. 画笔颜色 D. 绘制图形的填充颜色
9. turtle.color(color1,color2)中的 color1 表示()。
A. 画笔颜色 B. 填充颜色 C. 画布颜色 D. 文字颜色
10. turtle.color(color1,color2)中的 color2 表示()。

A. 画笔颜色

B. 填充颜色

C. 画布颜色

D. 文字颜色

三、填空题

1. 画布尺寸设置函数是_____。
2. 画笔抬起函数是_____。
3. 画笔尺寸设置函数是_____。
4. 画笔前进函数是_____。
5. 绝对角度设置函数是_____。
6. 画布的角度坐标系以_____为原点。
7. 画布内部的距离单位是_____。
8. 隐藏画笔的 turtle 形状函数是_____。
9. 显示画笔的 turtle 形状函数是_____。
10. 画圆命令是_____。

四、编程题

1. 编写程序绘制太极图,如图 5-9 所示。

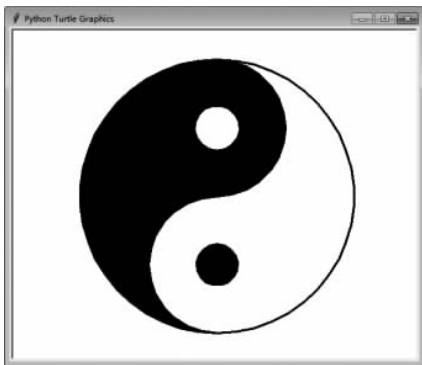


图 5-9 太极图

2. 编写程序绘制爱心祝福图形,如图 5-10 所示。



图 5-10 爱心祝福图形