第 3 章

函　　数

Chapter 3

【学习目标】

(1) 理解函数基本概念。

(2) 掌握函数定义与调用方法。

(3) 掌握函数参数。

(4) 掌握变量作用域用法。

(5) 学会使用模块组织程序。

(6) 掌握 lambda 函数。

(7) 掌握递归函数。

任务 3.1　定义函数与调用函数

【任务描述】

针对第 1 章中的中国结动画综合案例,将绘制单个中国结的功能进行模块化设计,定义相关的函数与接口,并进行调用。

【任务分析】

(1) 理解模块化程序设计方法。

(2) 掌握函数的定义。

(3) 掌握函数的调用。

3.1.1　函数定义与调用基础

函数是组织好的、可重复使用的、用来实现单一或相关联功能的代码段。

函数能提高应用的模块化和代码的重复利用率。用户可以根据自己的需要创建函数,这样的函数称作用户自定义函数,函数的功能需要编程实现。

1. 函数定义

函数定义的语法如下:

```
def 函数名(形式参数列表):
    封装的功能代码段
    [return 表达式]
```

(1) 函数定义包括函数头部和函数体两部分,函数头部用来定义调用接口,函数体用来封装功能代码段及返回值。

(2) 函数头部以关键字 def 开始,冒号结束,主要用来指明函数名与函数形参两部分。

(3) 函数名称一般用与函数功能相符合的词表示,同时符合标识符的命名规则。

(4) 封装的功能代码段要使用缩进格式,前面有 4 个空格。

(5) 形式参数列表是在调用函数时实际参数值的入口,一般以变量形式体现,多个形参之间用逗号分隔,其值在调用时由实际参数的值确定。

(6) 如果函数有返回值,则使用 return 语句返回。一个函数内部可以有多个 return,但只有运行时遇到的第一个 return 起作用。

2. 函数调用

函数调用的语法如下:

```
函数名(实际参数列表)
```

调用发生时,程序的运行控制权转到被调用的函数,实际参数的值传递给函数对应的形参。被调函数运行结束,控制权返回到调用处,如果函数有返回值,则可当作一个值来使用。调用的函数可以是内建函数、开源库中的函数以及自定义函数。

【代码 3-1】 已知一等腰直角三角形的斜边长度,定义一个函数,用来求其直角边长度,并调用该函数,求斜边长度为 10 的等腰直角三角形直角边长度。

```
1    def leg(len):                    # 函数头部,斜边长度作为形式参数
2        value=pow(len,0.5)
3        return value
4    result=leg(10)                   # 函数调用
5    print(result)
```

代码说明:

(1) 代码第 1 行为定义的函数头部,函数名为 leg,形式参数名称为 len,用来接收三角形斜边值。

(2) 代码第 2、3 行为函数内的功能代码,完成直角边长度的求解,并使用 return 返回结果,这两行代码需要保持缩进 4 个空格的格式。

(3) 代码第 4 行是对函数的调用,10 为实际参数,值被传递给 len。

运行结果如下:

```
14.142135623730951
```

如图 3-1 所示,将单个中国结根据绘制特点划分为五部分,分别为中心 center、边缘拱形 arc、两侧的圆耳朵 ear、顶部的丝线 topthread 以及底部的穗子 tassel。需要定义五个函数来绘制各个部分。因为中国结的大小和位置是可以变化的,所以需要将中国结位置、尺寸设置为形式参数,在这里设置为(x,y,size),分别代表结心顶端横、纵坐标和结心边线的长度。

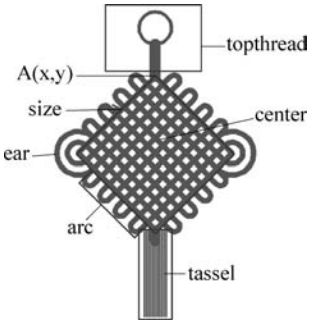

图 3-1 单个中国结分解示意图

【代码 3-2】 定义绘制单个中国结的相关函数，并调用输出。

```
1    def center(x,y,size):
2        # 绘制中国结结心的函数
3        print("中国结的结心")
4    def arc(x,y,size):
5        # 绘制中国结边缘拱形的函数
6        pass                    # 空函数体
7    def ear(x,y,size):
8        # 绘制中国结两侧圆耳朵的函数
9        pass
10   def topthread(x,y,size):
11       # 绘制中国结顶部丝线的函数
12       pass
13   def tassel(x,y,size):
14       # 绘制中国结底部穗子的函数
15       pass
16   center(0,0,200)
```

代码说明：

(1) 分别对应如图 3-1 所示的五个部分，每个部分定义为一个函数，每个函数负责完成对应部分的图形绘制。

(2) 代码第 1～15 行分别定义了一个中国结的五个部分，center()函数功能部分暂时用 print()函数来打印说明，其他函数使用 pass 关键字表示函数体内为空。随着所学知识的扩展，所有函数体将用真正的图形绘制代码来替代。

(3) 代码第 16 行是一个函数调用语句，调用 center()，传入实际参数 0，0，200。

运行结果如下：

```
0 0 200 中国结的结心
```

【代码 3-3 】 定义一个函数 jiemain()，能够调用例 3-2 中定义的五个函数，并传入正确的参数。

```
1    def jiemain(x,y,size):
2        center(x,y,size)
3        arc(x,y,size):
4        ear(x,y,size):
5        topthread(x,y,size):
6        tassel(x,y,size):
7    jiemain(0,0,200)
```

代码说明：

(1) 代码第 1～6 行定义了 jiemain()函数，其参数是如图 3-1 所示的 x、y、size。函数体部分分别调用了五部分的函数，传入的参数也是以如图 3-1 所示的 A 点为基础。进入函数内部需要按绘制图形以 A 点为基准重新计算起点坐标和绘制线条的长度。

(2) 代码第 7 行以实际参数 0、0、200 调用 jiemain()函数。应该注意的是，只有 center()函数有内容输出，其他函数没有功能代码。

运行结果如下：

```
0 0 200 中国结的结心
```

3.1.2 Turtle 库函数

Python 提供了很多标准库，为实现图形的绘制，本节将重点介绍 Turtle 库。在使用库函数之前需要导入库模块，语法格式如下：

```
import 库名                #方法(1)
import  库名  as 别名      #方法(2)
from  库名  import  *      #方法(3)
```

使用方法(1)导入库后，在后续调用函数时，会使用“库名.函数()”的方式加以调用；使用方法(2)导入库后，在后续调用函数时，会使用“别名.函数()”的方式加以调用；使用方法(3)导入库后，在后续可直接使用“函数()”的方式进行调用。

Turtle 库是 Python 语言的标准库之一，是入门级的图形绘制函数库，实现了一只小海龟在屏幕上游走，而行走的轨迹形成了图像。其坐标系如图 3-2 所示，海龟的头初始朝向 x 轴正方向。

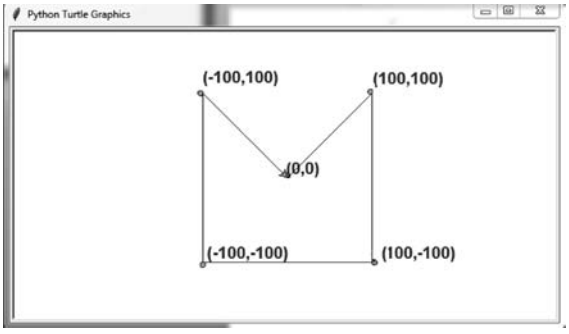

图 3-2 Turtle 绘图坐标系示意图

可以通过函数改变海龟头的方向，角度坐标体系如图 3-3 所示。

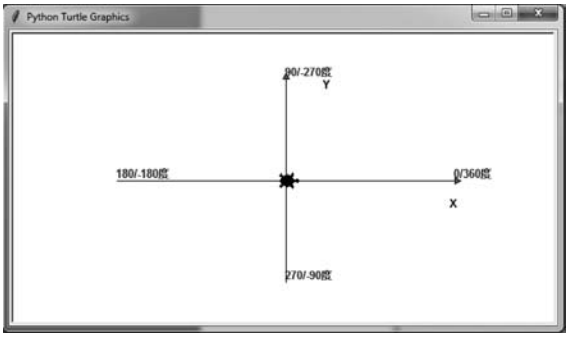

图 3-3 Turtle 库的角度坐标体系

在这两个坐标基础上可以使用 Turtle 库函数完成图形的绘制功能。常用库函数见表 3-1。

表 3-1 Turtle 库中常用的库函数

函 数	功 能 说 明	示 例
setup(width,height,x,y)	在屏幕(x,y)的位置绘制一个宽 width、高 height 的窗口	setup(800,600,0,0)
screensize(width,height,color)	创建一个宽 width、高 height、背景色 color 的画布	screensize(800,600,'red')

续表

函　数	功能说明	示　例
tracer(bool)	画笔追踪开关	True为展示绘制过程,False为隐藏
hideturtle()	隐藏小海龟	hideturtle()
pensize(数值)	画笔线条粗细	pensize(10)
pencolor(颜色)	画笔颜色设置	pencolor('blue')
penup()	笔抬离画布不再绘制	ponup()
goto(x,y)	将笔移动至(x,y)处	goto(100,100)　若笔在界面上,则移动中绘制
pendown()	将笔放下,置于画布上	pendown()
seth(角度)	改变笔的方向	seth(90)　竖直向上
left(角度)	当前方向向左转向一指定的角度	left(90)
right(角度)	当前方向向右转向一指定的角度	right(135)
fd(长度) 或 forward(长度)	当前方向绘制指定长度的直线	fd(100)
circle(半径[,圆弧角度])	绘制指定半径、指定圆周角的弧,半径的正负可确定按逆时针或顺时针画	circle(50,180)　逆时针 circle(−50,180)　顺时针
begin_fill() end_fill()	填充区域开始; 填充区域结束	begin_fill()　#绘制一个填充圆 circle(50) end_fill()
done()	结束绘图,所有绘制结束后调用	done()
listen()	启动键盘事件监听	listen()
onkey('key',lambda 函数)	定义键 key 对应的处理函数	onkey('a',lambda:fun(−5))
ontimer(函数,t 毫秒)	定时器,每 t 毫秒调用指定函数	ontimer(main,58)


代码 3-4

【代码 3-4】 定义一个函数,初始化一个宽 800、长 600 的窗口,位置定位在屏幕左上角。设置绘制速度为 14,隐藏乌龟图片及绘制过程。调用函数初始化后,在窗口中心绘制各种情况的圆弧。

```
1    import turtle as t                    # 导入 turtle 库,别名为 t
2    def init():
3        t.screensize(800,600,'white')   # 创建一个宽度 800,高度 600 的窗口
4        t.speed(14)
5        t.hideturtle()
6    def upgoto(x,y):              # 第 6～9 行定义一个函数 upgoto(),用来将画线移至(x,y)处
7        t.penup()
8        t.goto(x,y)
9        t.pendown()
10   def drawArc(x,y,radius,angle):
11       upgoto(x,y)                       # 第 11～13 行用于绘制圆弧起点标志,为蓝色圆
12       t.pencolor('blue')
13       t.circle(2,360)
14       t.pencolor('red')  # 第 4～16 行用于绘制方向 angle 角度起始的一个 270°的红色圆弧
15       t.seth(angle)
16       t.circle(radius,270)
17   init()
18   t.pensize(10)
```

```
19   drawArc(-200,100,50,0)              # 逆时针绘制一个半径为 50 的 270°圆弧
20   drawArc(-50,100,50,135)             # 逆时针绘制一个半径为 50 的 270°圆弧
21   drawArc(0,100,50,-45)               # 逆时针绘制一个半径为 50 的 270°圆弧
22   drawArc(150,100,-50,45)             # 顺时针绘制一个半径为 50 的 270°圆弧
23   t.done()
```

代码说明：

(1) 代码第 1 行，导入 turtle 库，并使用别名 t。

(2) 代码第 2～5 行定义一个函数 init()初始化窗口及画笔的初始状态，即窗口宽为 800 像素、高为 600 像素，而背景色为白色。

(3) 代码第 6～9 行定义一个 upgoto()函数，用于将画笔抬起来移动到(x,y)处并放下准备绘制，后面即可直接调用该函数，实现代码的可重用性。

(4) 代码第 10～16 行定义一个 drawArc()函数，用于绘制一个 270°圆弧，参数 x、y 代表圆弧起点位置，radius 为圆弧半径，angle 圆弧初始角度。

(5) 代码第 11～13 行为在(x,y)处绘制一个蓝色小圆，表示当前圆弧的起笔位置，便于了解 circle 函数中的参数含义。

(6) 代码第 14～16 设置绘制圆弧的画笔颜色为红色，起始角度为 angle，绘制一个角度为 270°的半径为 radius 的圆弧。

(7) 代码第 17 行调用 init 进行窗口初始化，第 18 行设置笔的粗细，第 19～22 行分别用不同的实际参数值调用 drawArc()函数。

(8) 代码第 23 行用以结束 turtle 的绘制过程，从而允许窗口关闭。

运行结果如图 3-4 所示。在看到运行结果后，可以尝试注释掉第 5 行的代码，观察运行过程，这样有助于了解 hideturtle()函数的作用效果。可以通过修改圆弧的方向、坐标及顺时针、逆时针方向，从而深入学习 circle()函数的用法。

图 3-4 代码 3-4 运行结果

【拓展思考 3-1】 根据 Turtle 函数，思考如何绘制夜空中的一轮黄色月牙。

【代码 3-5】 使用 Turtle 库完善代码 3-2 中的函数 center，绘制最上方的两条相交粗红实线。

代码 3-5

```
1    import turtle as t
2    def center(x,y,size):
3        t.pensize(4)                    # 设置画笔的粗细值为 4
4        upgoto(x,y)
5        t.seth(-45)                     # 向右下方转 45°,角度值见图 3-3
```

```
6        t.forward(size)              # 画一条长度为 size 的直线
7        upgoto(x,y)
8        t.seth(-135)                 # 向左下方转 45°
9        t.forward(size)              # 画一条长度为 size 的直线
10   init()                           # 窗口初始化,来自代码 3-4
11   t.pencolor('red')                # 画笔的颜色设置为红色
12   center(0,100,200)
```

代码说明：

(1) 代码第 10 行调用 init()函数,来自代码 3-4,用以初始化窗口。

(2) 代码如图 3-1 所示 A 点处两条相交的结心红线,在画直线前需要调整角度、画笔粗细以及画笔颜色。

(3) 代码第 6～8 行为从(x,y)处绘制一条朝向右下方 45°的粗红实线。

(4) 代码第 9～11 行为从(x,y)处绘制一条朝向左下方 45°的粗红实线。

运行结果如图 3-5 所示。

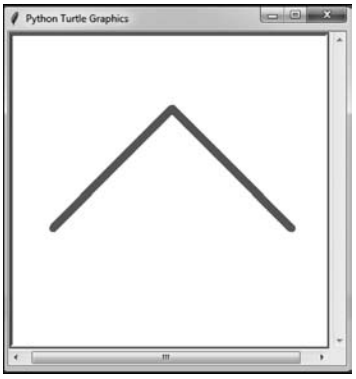

图 3-5　中国结结心的基本元素绘制结果

【拓展思考 3-2】 根据代码 3-5,如何在夜空中绘制白色五角星。

【代码 3-6】 完善代码 3-2 中的函数 topthread,绘制中国结上方的绳结。

代码 3-6

```
1    def topthread(x,y,size):
2        step=size/10
3        upgoto(x,y)
4        t.pensize(size/10)
5        t.seth(90)                   # 画笔垂直向上
6        t.fd(step* 3)                # 绘制一条向上的直线
7        upgoto(x,y+step* 6)          # 将画笔放置到最高点
8        t.pensize(6)                 # 将笔的粗细设置为 6
9        t.seth(180)                  # 向上转到最左方
10       t.circle(step*3/2,360)       # 逆时针画圆,半径为 step* 3/2
11   def hanzi():
12       upgoto(-90,200)              # 将笔放到坐标(-90,200)处
13       t.write("幸福中国结",font=("Arial",20,"normal"))     # 输出汉字
14   def main():
15       t.clear()
16       jiemain(-100,100,100)        # 来自代码 3-2
17       hanzi()
```

```
18   init()
19   main()
20   t.done()
```

代码说明：

(1) 代码第 1～10 行是定义一个函数 topthread()，用来绘制如图 3-1 所示的 topthread() 部分，绘制一个圆及一条垂直的粗红线。

(2) 代码第 3～6 行是从坐标(x,y)起，将画笔转为垂直向上，绘制一条垂直的长度为 3 * step 的直线。

(3) 代码第 7～10 行是将画笔带到最高点并将头转向最左方并绘制圆环。

(4) 代码第 11～13 行在指定位置绘制“幸福中国结”。

(5) 代码第 14～17 行定义一个 main()函数来调用绘制结心和绳结的函数。

(6) 代码第 19～21 行用来初始化和调用 main()函数及结束绘制。

运行结果如图 3-6 所示。

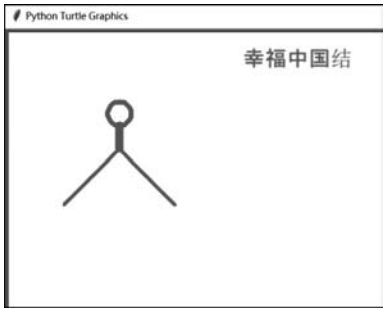

图 3-6 增加顶部丝线的中国结程序运行结果

【代码 3-7】 完善代码 3-2 中的函数 arc()，绘制中国结结心四面的拱形，每条边绘制一个。

```
1    def arc(x,y,size):
2        t.pensize(4)                          # 设定画笔的粗细为 4
3        step=size/10                          # 结心长度十等分
4        m=x-step/pow(2,0.5)                   # 确定左上方拱形的位置
5        n=y-step/pow(2,0.5)
6        upgoto(m,n)
7        t.seth(135)                           # 设置画笔方向为左上方 45°
8        t.fd(step)                            # 画拱形的一条线段
9        t.circle(step/2,180)                  # 逆时针绘制半圆
10       t.fd(step)                            # 画拱形的另一条线段
11       m=x+step/pow(2,0.5)                   # 确定右上方拱形的位置
12       n=y-step/pow(2,0.5)
13       upgoto(m,n)
14       t.seth(45)                            # 画笔向右上方 45°
15       t.fd(step)                            # 绘制一条长度为 step 的线段
16       t.circle(-step/2,180)                 # 顺时针绘制一个半圆
17       t.fd(step)                            # 绘制另一条线段
18       m=x-step/pow(2,0.5)                   # 确定左下方拱形的位置
19       n=y-size* pow(2,0.5)+step/pow(2,0.5)
20       upgoto(m,n)
```


代码 3-7

```
21      t.seth(-135)                    # 画笔向左下方 45°
22      t.fd(step)                      # 绘制一条线段
23      t.circle(-step/2,180)           # 顺时针绘制一个半圆
24      t.fd(step)                      # 绘制一条线段
25      m=x+step/pow(2,0.5)             # 确定右下方拱形的位置
26      n=y-size* pow(2,0.5)+step/pow(2,0.5)
27      upgoto(m,n)
28      t.seth(-45)                     # 画笔向右下方 45°
29      t.fd(step)                      # 向右下方绘制一条线段
30      t.circle(step/2,180)            # 逆时针绘制一个半圆
31      t.fd(step)                      # 绘制另一条线段
```

代码说明：

(1) 代码用来绘制如图 3-1 所示的函数 arc()拱形中的一部分，每条边上绘制一个拱形。

(2) 代码第 2、3 行是设定画笔粗细及绘制拱形圆的半径大小。圆的半径为拱形直线的一半。

(3) 拱形代码的要点是绘制两条线段及一个半圆，需要确定起始位置及直线方向，半圆需要指明是顺时针方向还是逆时针方向。

(4) 代码第 4～10 行绘制如图 3-7 所示，左上角的一个拱形，其起点位置在坐标(x,y)的左下方，所以(x,y)可以按以 step 为斜边长度的直角三角形的直角边长度 step/pow(2,0.5)进行递减。方向为 135°，即北偏西 45°方向。在画完一条线段后逆时针绘制一个半圆，之后绘制偏下一点的那条直线，从而完成一个拱形的绘制。可以尝试注释掉第 10 行代码，观察运行结果，这样可以更好地了解绘制过程。

(5) 代码第 11～17 行绘制右上角的拱形，此处的起点坐标纵坐标与第一个拱形相同，横坐标比起点的 x 坐标略向右，所以是 m＝x＋step/pow(2,0.5)，方向是 45°，按此起点位置方向绘制一条长度为 step 的直线后顺时针绘制半圆，绘制函数 t.circle(-step/2,180)，第 1 个参数使用负数，表示顺时针。最后绘制另一条直线。

(6) 代码第 18～24 行绘制左下角的拱形，参数可以参考第 1、2 个拱形。

(7) 代码第 25～31 行绘制右下方的拱形，参数可以参考前 3 个拱形，也可以把这部分内容注释掉，观察运行结果。

用代码 3-7 替换之前的 arc()函数，整体运行结果如图 3-7 所示。

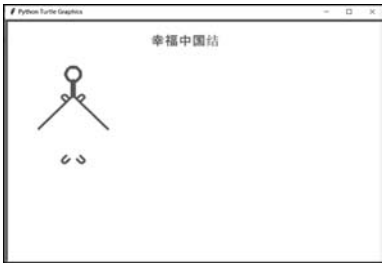

图 3-7　增加边缘拱形的中国结运行结果

【拓展思考 3-3】 研究代码 3-7，修改其中的绘制角度，观察运行效果，感受绘图时角度改变对直线绘制的影响。

【代码 3-8】 完善代码 3-2 中的函数 ear()，绘制中国结两侧的圆耳朵，其中涉及四个圆弧的绘制。

代码 3-8

```
1    def ear(x,y,size):
2        t.pensize(4)
3        step=size/10/2
4        upgoto(x-step*9*pow(2,0.5),y-step*9*pow(2,0.5))     # 确定左侧内圆弧位置
5        t.seth(135)                                  # 确定绘制方向
6        t.circle(step*  2,270)                       # 逆时针绘制 270°圆弧,半径 2*step
7        upgoto(x-step*8*pow(2,0.5),y-step*8*pow(2,0.5))     # 确定左侧外圆弧位置
8        t.seth(135)                                  # 确定绘制方向
9        t.circle(step*4,270)                         # 逆时针绘制 270°圆弧,半径 4*step
10       upgoto(x+step*9*pow(2,0.5),y-step*9*pow(2,0.5))     # 确定右侧内圆弧位置
11       t.seth(45)                                   # 确定绘制方向
12       t.circle(-step*2,270)                        # 顺时针绘制 270°圆弧,半径 2*step
13       upgoto(x+step*8*pow(2,0.5),y-step*8*pow(2,0.5))     # 确定右侧内圆弧位置
14       t.seth(45)                                   # 确定绘制方向
15       t.circle(-step*4,270)                        # 顺时针绘制 270°圆弧,半径 4*step
```

代码说明：

(1) 代码第 3 行,step 设为结心线长的 1/10。

(2) 代码第 5～7 行绘制左侧的内圆弧,绘制方向北偏西 45°。起点坐标在(x,y) 左下方,所以横纵坐标分别减去以 9 倍 step 为斜边的等腰直角三角形的直角边长度,公式分别为 x-step * 9 * pow(2,0.5)和 y-step * 9 * pow(2,0.5),之后逆时针绘制一个半径为 2 倍 step 的 270°的圆弧。

(3) 代码第 9～11 行绘制左侧的外圆弧,绘制方向东偏南 45°。起点坐标在(x,y) 左下方,所以横纵坐标分别减去以 8 倍 step 为斜边的等腰直角三角形的直角边长度,公式分别为 x-step * 8 * pow(2,0.5)和 y-step * 8 * pow(2,0.5),之后逆时针绘制一个半径为 4 倍 step 的 270°的圆弧。

(4) 代码第 13～15 行绘制右侧的内圆弧,起点坐标与绘制方向均可参考左侧内圆弧。

(5) 代码第 17～19 行绘制右侧的外圆弧,起点坐标与绘制方向均可参考右侧内圆弧。

运行结果如图 3-8 所示。

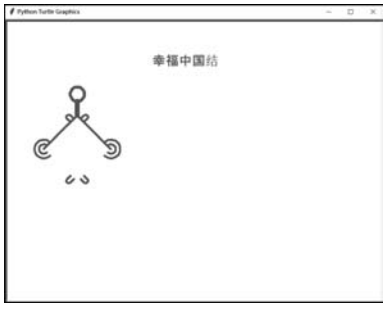

图 3-8 增加中国结两侧的圆耳朵后的运行结果

【代码 3-9】 完善代码 3-2 中的函数 tassel(),绘制中国结下方的穗子。


代码 3-9

```
1    def tassel(x,y,size):
2        step=size/10
3        upgoto(x,y-size*pow(2,0.5))                  # 确定穗子上面粗扣的位置
4        t.pensize(step)                              # 设定粗笔
5        t.seth(-90)                                  # 方向垂直向下
6        t.fd(step)                                   # 绘制粗结
7        t.pensize(1)                                 # 设定细笔
8        s=x-step                                     # 设定最左侧细穗的横坐标
```