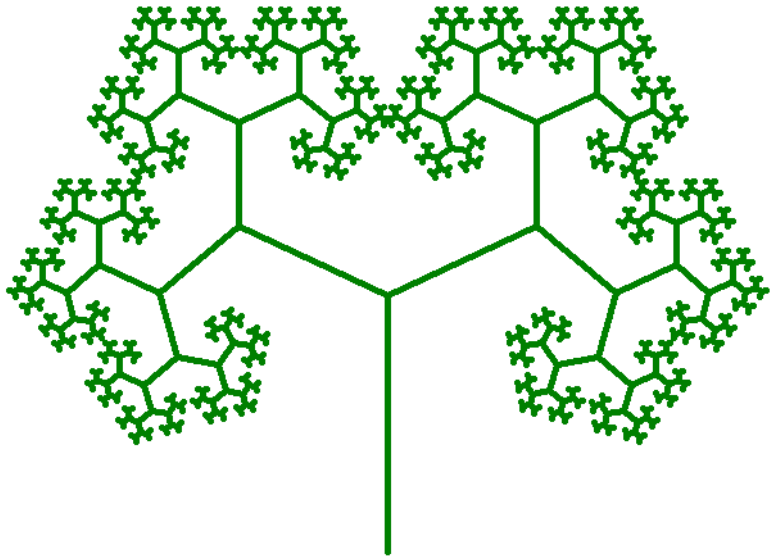
函数实例

礼 欣

北京理工大学

实例

- 任务：通过编写程序完成在电脑上绘制如图所示的这颗树。
- 将任务拆解为两部分
  1. 学习简单图形绘制的指令
  2. 为树的绘制设计算法



python 语言程序设计

Turtle库回顾

- turtle库是非常适合初学者甚至小朋友使用的简单图形绘制模块。
- 自Python2.6版本以后，turtle库就已经成为Python的内嵌模块，无需特别安装。
- turtle中的指令，形象而简单，它绘制的坐标轴以屏幕中心点为原点。

python 语言程序设计

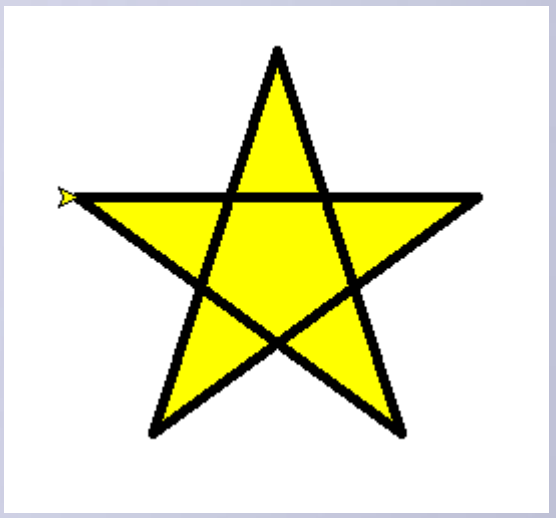
Turtle库

- 下列turtle库的简单常用指令，请大家练习查询官方提供的turtle使用手册掌握以下命令的用法
  - forward(distance) #将箭头移到某一指定坐标
  - left(angel) right(angel)
  - penup() #提起笔，用于另起一个地方绘制时用，与pendown()配对使用
  - goto(x,y)
  - home()
  - circle(radius)
  - speed()

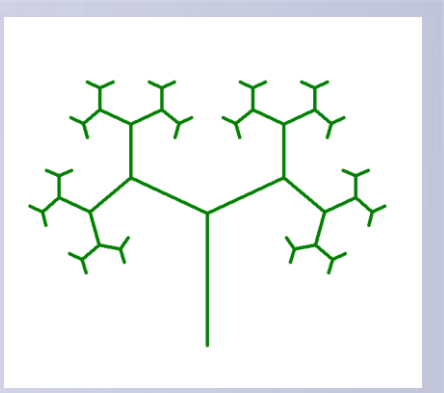
Turtle库

- 下面是一个利用turtle库绘制并填充一个五角星的简单程序
- 代码如下

```
from turtle import Turtle

p = Turtle()
p.speed(3)
p.pensize(5)
p.color("black", 'yellow')
#p.fillcolor("red")
p.begin_fill()
for i in range(5):
    p.forward(200)
    p.right(144)
p.end_fill()
```



树的绘制算法设计



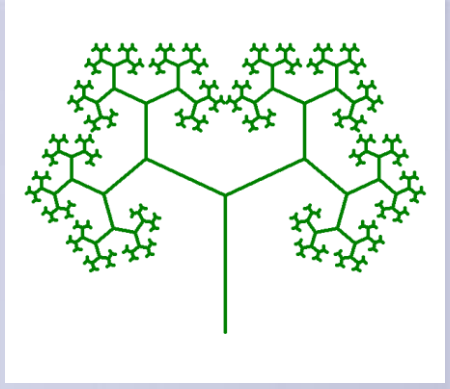
- 观察树的图案， 这是一个对称树， 从主杆出发以一定角度向左向右生成对称的枝丫， 而每一棵枝杈上以相同的角度生成更小的左右枝杈，如此往复。联系我们所学过的内容，很容易想到可以利用递归程序实现， 程序代码如下：
- 注意：以下代码为turtle库提供标准例程的简化版，以注释的形式保留了原例程中的部分代码和解释，供大家练习、加深理解
  - 可以练习以p.speed()替代p.getscreen.tracer()的调用，调整绘画速度

```
def main():
    p = Turtle()
    p.color("green")
    p.pensize(5)
    #p.setundobuffer(None)
    p.hideturtle()
    #Make the turtle invisible. It's a good idea to do this while
    # you're in the middle of doing some complex drawing,
    #because hiding the turtle speeds up the drawing observably.
    #p.speed(10)
    p.getscreen().tracer(30,0)
    #Return the TurtleScreen object the turtle is drawing on.
    #TurtleScreen methods can then be called for that object.
    p.left(90)# Turn turtle left by angle units. direction 调整画笔

    p.penup() #Pull the pen up - no drawing when moving.
    p.goto(x,y)#Move turtle to an absolute position.
    # If the pen is down, draw line. Do not change the turtle's orientation.
    p.pendown()# Pull the pen down - drawing when moving.
    #这三条语句是一个组合相当于先把笔收起来再移动到指定位置，再把笔放下开始画
    #否则turtle一移动就会自动的把线画出来

    #t = tree([p], 200, 65, 0.6375)
    t = tree([p], 110, 65, 0.6375)

main()
```

Tree函数的代码

```
def tree(plist, l, a, f):
    """ plist is list of pens
    l is length of branch
    a is half of the angle between 2 branches
    f is factor by which branch is shortened
    from level to level."""
    if l > 5:
        lst = []
        for p in plist:
            p.forward(l)#沿着当前的方向画画
            #Move the turtle forward by the specified distance,
            #in the direction the turtle is headed.
            q = p.clone()#Create and return a clone of the turtle
            #with same position, heading and turtle properties.
            p.left(a) #Turn turtle left by angle units
            q.right(a)# turn turtle right by angle units, nits are
            #by default degrees, but can be set via the degrees() and
            #radians() functions.
            lst.append(p)#将元素增加到列表的最后
            lst.append(q)
        tree(lst, l*f, a, f)
```

树的绘制算法设计



- 请通过调整绘画速度，观察绘画过程，更好的理解递归程序的调用过程。
- 思考：之前讲过递归程序一定要有基例，否则递归程序一直进行无法终止，将导致程序崩溃，那在我们这个tree()函数的编写过程中，我们使用的基例是什么呢？

python 语言程序设计

森林的绘制

- 如何画出多棵树，甚至整片森林呢？
  - 答案很简单，只要在画每棵树之前调整画笔的位置，调用画树程序，就可以从新位置生成一颗新树了。
- 利用模块化的函数思想，调整代码：
  - 将每棵树的绘制以maketree函数封装，参数x,y为画树的起点位置即树根位置。在main函数中只要以不同的参数设置来调用maketree函数就可以完成多棵树的绘制了

```
def maketree(x,y):
    p = Turtle()
    p.color("green")
    p.pensize(5)
    #p.setundobuffer(None)
    p.hideturtle()
    #Make the turtle invisible. It's a good idea to do this while
    # you're in the middle of doing some complex drawing,
    #because hiding the turtle speeds up the drawing observably.
    #p.speed(10)
    p.getscreen().tracer(30,0)
    #Return the TurtleScreen object the turtle is drawing on.
    #TurtleScreen methods can then be called for that object.
    p.left(90)# Turn turtle left by angle units. direction 调整画笔

    p.penup() #Pull the pen up - no drawing when moving.
    p.goto(x,y)#Move turtle to an absolute position.
    # If the pen is down, draw line. Do not change the turtle's orientation.
    p.pendown()# Pull the pen down - drawing when moving.
    #这三条语句是一个组合相当于先把笔收起来再移动到指定位置，再把笔放下开始画
    #否则turtle一移动就会自动的把线画出来

    #t = tree([p], 200, 65, 0.6375)
    t = tree([p], 110, 65, 0.6375)
    print(len(p.getscreen().turtles()))#用了多少个turtle绘制
def main():
    maketree(-200,-200)
    maketree(0,0)
    maketree( 200,-200)
main()
```