



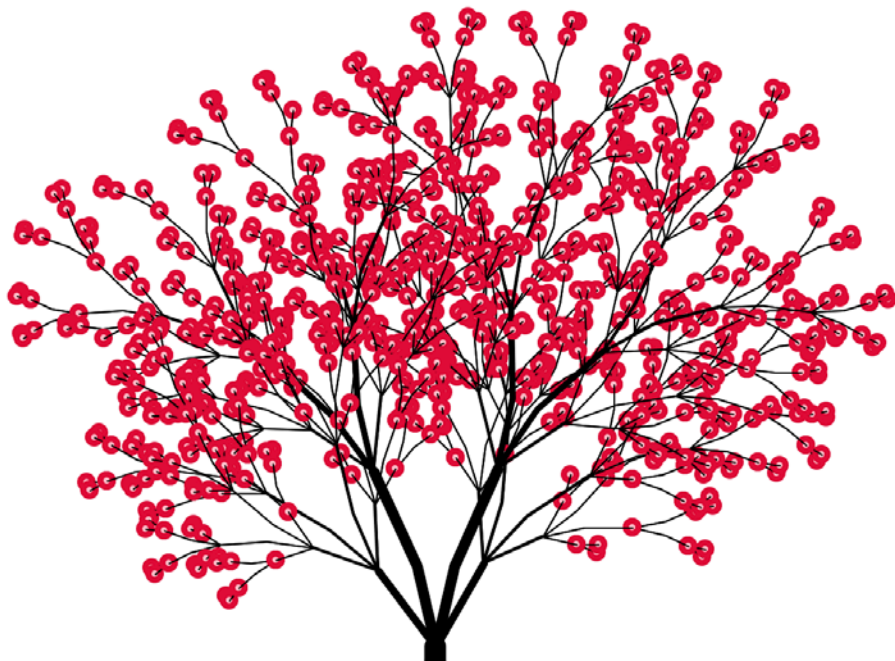
数据结构与算法 (Python版)

递归可视化：分形树

陈斌 北京大学 gischen@pku.edu.cn

递归可视化：图示

- ❖ 前面的种种递归算法展现了其简单而强大的一面，但还是难有个直观的概念
- ❖ 下面我们通过**递归作图**来展现递归调用的视觉影像



递归可视化：图示

❖ Python的**海龟作图系统turtle** module

Python内置，随时可用，以LOGO语言的创意为基础

其意象为模拟海龟在沙滩上爬行而留下的足迹

爬行：forward(n); backward(n)

转向：left(a); right(a)

抬笔放笔：penup(); pendown()

笔属性：**pensize**(s); pencolor(c)

递归可视化：图示

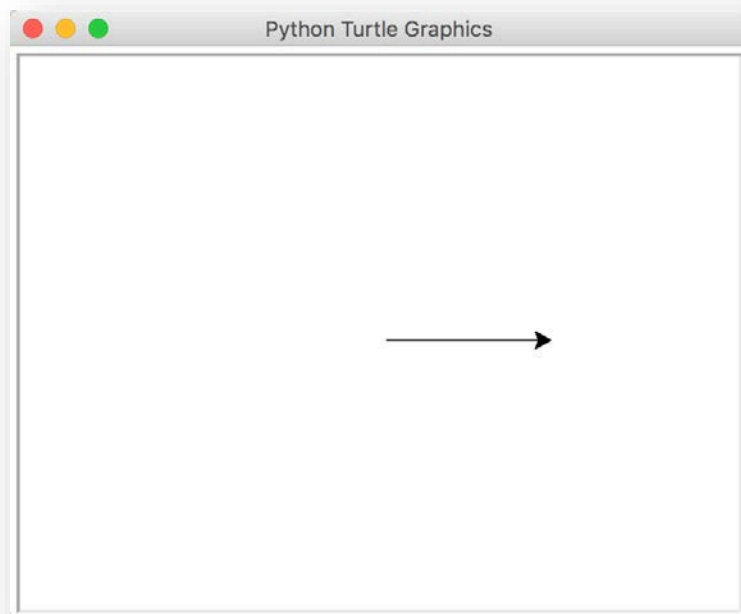
```
import turtle  
t= turtle.Turtle()
```

作图开始

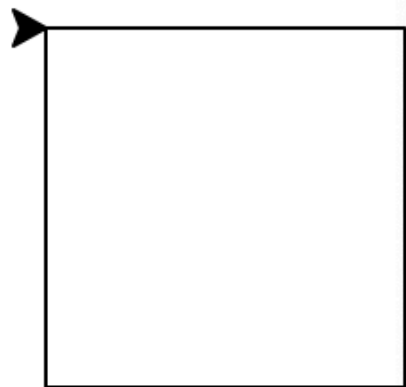
```
t.forward(100) #指挥海龟作图
```

作图结束

```
turtle.done()
```



海龟作图



t1.py ×

```
1  import turtle
2
3  t = turtle.Turtle()
4
5  for i in range(4):
6      t.forward(100)
7      t.right(90)
8
9  turtle.done()
10
```

海龟作图



```
t1.py x t2.py x
1  import turtle
2
3  t = turtle.Turtle()
4
5  t.pencolor('red')
6  t.pensize(3)
7  for i in range(5):
8      t.forward(100)
9      t.right(144)
10 t.hideturtle()
11
12 turtle.done()
13
```

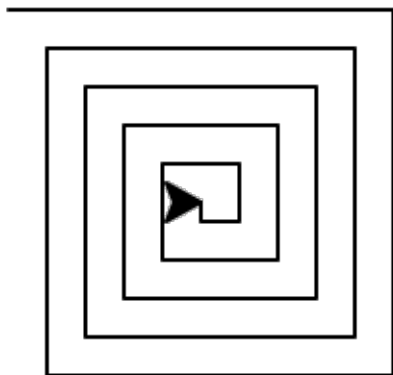
一个递归作图的例子：螺旋

最小规模, 0直接退出

减小规模, 边长减5

```
1 import turtle
2
3 t = turtle.Turtle()
4
5 def drawSpiral(t, lineLen):
6     if lineLen > 0:
7         t.forward(lineLen)
8         t.right(90)
9         drawSpiral(t, lineLen - 5)
10
11 drawSpiral(t, 100)
12
13 turtle.done()
```

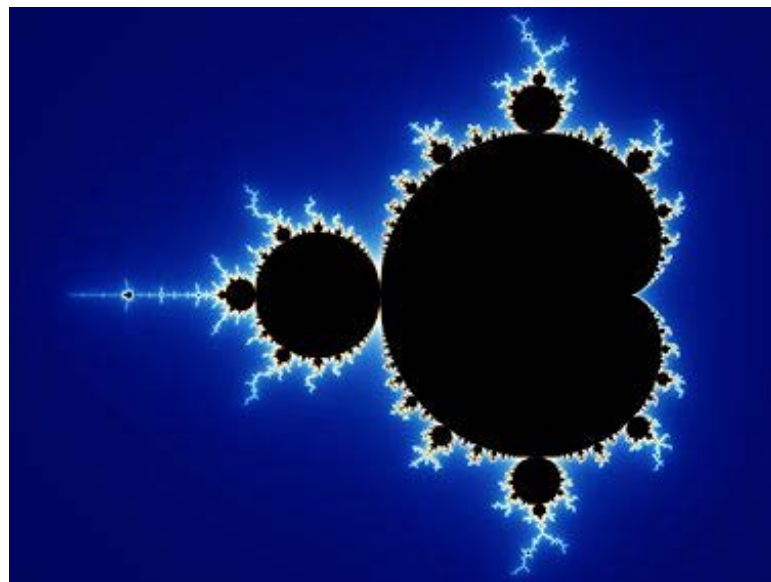
调用自身



分形树：自相似递归图形

❖ **分形Fractal**，是1975年由Mandelbrot
开创的新学科

“一个粗糙或零碎的几何形状，可以分成数个部分，且每一部分都（至少近似地）是整体缩小后的形状”，即具有**自相似的性质**。



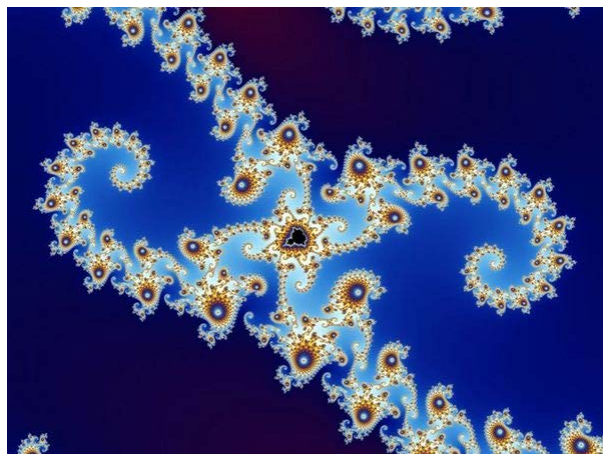
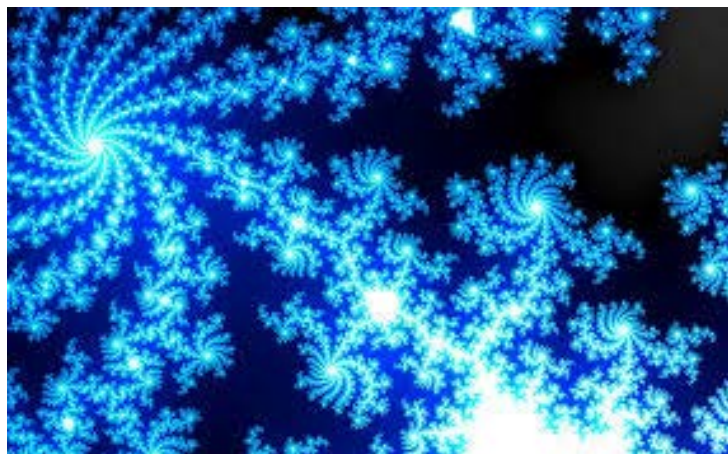
分形树：自相似递归图形

❖ 自然界中找到众多具有分形性质的物体

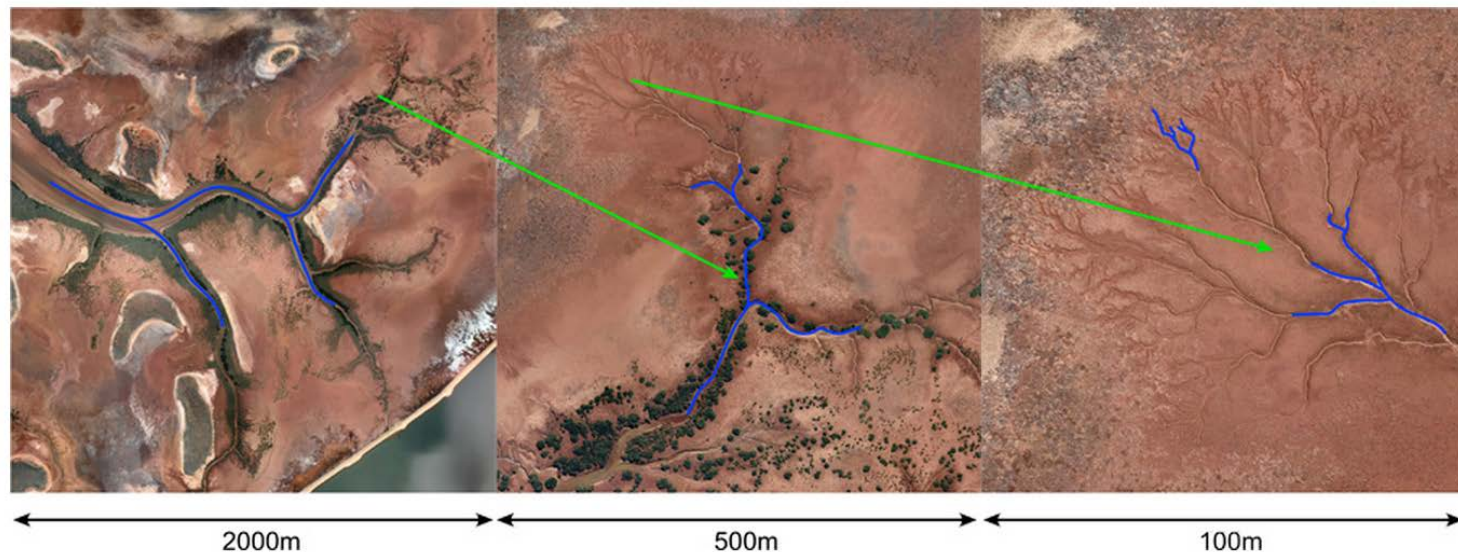
海岸线、山脉、闪电、云朵、雪花、树

<http://paulbourke.net/fractals/googleearth/>

<http://recursivedrawing.com/>



自然界不是平滑的



分形树：自相似递归图形

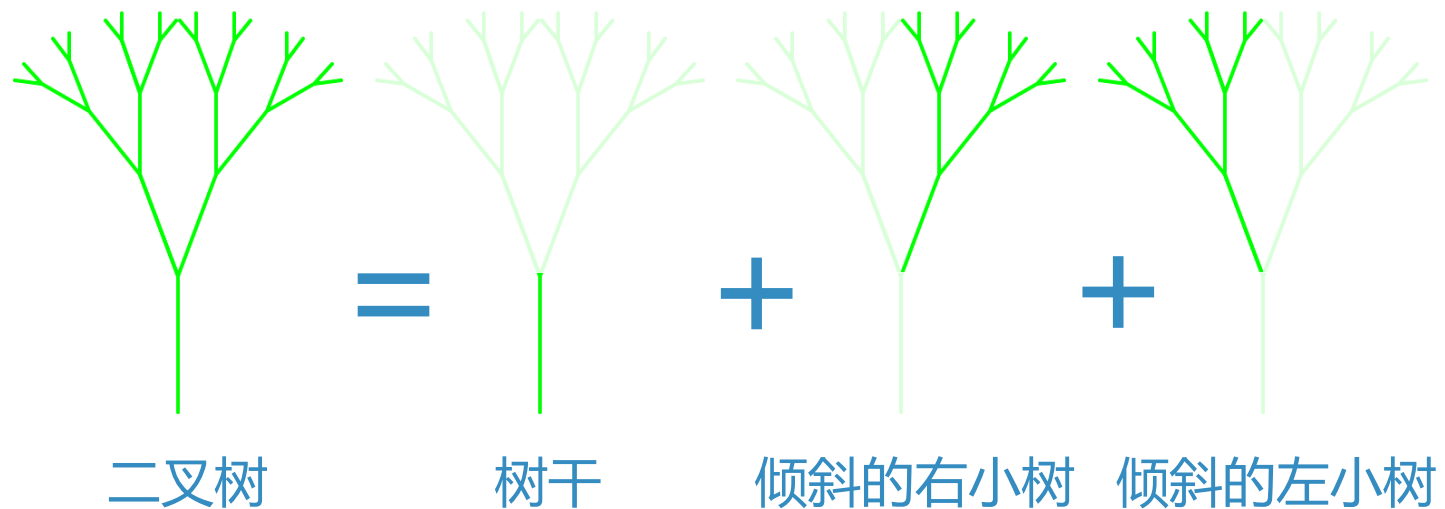
- ❖ 自然现象中所具备的分形特性，使得计算机可以通过分形算法生成非常逼真的自然场景
- ❖ 分形是在不同尺度上都具有相似性的事物
我们能看出一棵树的每个分叉和每条树枝，实际上都具有整棵树的外形特征（也是逐步分叉的）



分形树：自相似递归图形

❖ 这样，我们可以把树分解为三个部分：树干、左边的小树、右边的小树

分解后，正好符合递归的定义：对自身的调用



分形树：代码

```
1  import turtle
2
3
4  def tree(branch_len):
5      if branch_len > 5: # 树干太短不画，即递归结束条件
6          t.forward(branch_len) # 画树干
7          t.right(20) # 右倾斜20度
8          tree(branch_len - 15) # 递归调用，画右边的小树，树干减15
9          t.left(40) # 向左回40度，即左倾斜20度
10         tree(branch_len - 15) # 递归调用，画左边的小树，树干减15
11         t.right(20) # 向右回20度，即回正
12         t.backward(branch_len) # 海龟退回原位置
13
15  t = turtle.Turtle()
16  t.left(90)
17  t.penup()
18  t.backward(100)
19  t.pendown()
20  t.pencolor('green')
21  t.pensize(2)
22  tree(75) # 画树干长度75的二叉树
23  t.hideturtle()
24  turtle.done()
```

