3.1 知识点定位

青少年编程能力“Python 二级”核心知识点 3：递归及算法。

3.2 能 力 要 求

掌握并熟练使用简单的函数递归，具备利用递归处理简单问题的能力。

3.3 建议教学时长

本单元建议 2 课时。

3.4 教 学 目 标

1. 知识目标

本单元以函数递归为主，通过大自然场景联系实际案例，让学生掌握递归与函数递归及简单的递归实现方法，为 Python 的后续学习打好坚实的基础。

2. 能力目标

通过对 Python 函数递归的学习，掌握计算机解决困难问题的方法，锻炼学习者发现问题、解决问题的能力，培养学习者化繁为简的能力。

3. 素养目标

引入斐波那契数列、斐波那契数列螺旋线及数学问题等相关内容，增加趣味性，增进学习者对自然界的了解；通过学习自然界的递归问题，用自然科学解析自然界现象，促进学生养成学以致用的良好习惯。

3.5 知 识 结 构

本单元知识结构如图 3-1 所示。

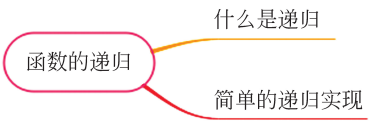

图 3-1　函数的递归知识结构

3.6 补充知识点

1. 神秘的斐波那契数列

斐波那契数列在大自然中普遍存在，如植物的花瓣数量、花草分蘖的枝丫、密集的种子排列……例如，如图 3-2 所示的松塔种子、鹦鹉螺结构等也满足斐

波那契数列排列的规律。据估计，植物中大约有 90% 的叶片排列方式或花瓣数列涉及斐波那契数列。大家感受到这串神秘数字的厉害之处了吗?

斐波那契数列的规律很简单，但就是这么简单的一个数列，却蕴藏着无穷的秘密。

如果计算这个数列的前一项与后一项的比值，就会发现，随着取得数越来越大，其比值也会越来越趋近于 0.618 033 988…没错，这就是黄金分割率，让无数科学家、数学家、艺术家为之着迷的数字，所以斐波那契数列又被称为黄金分割数列。

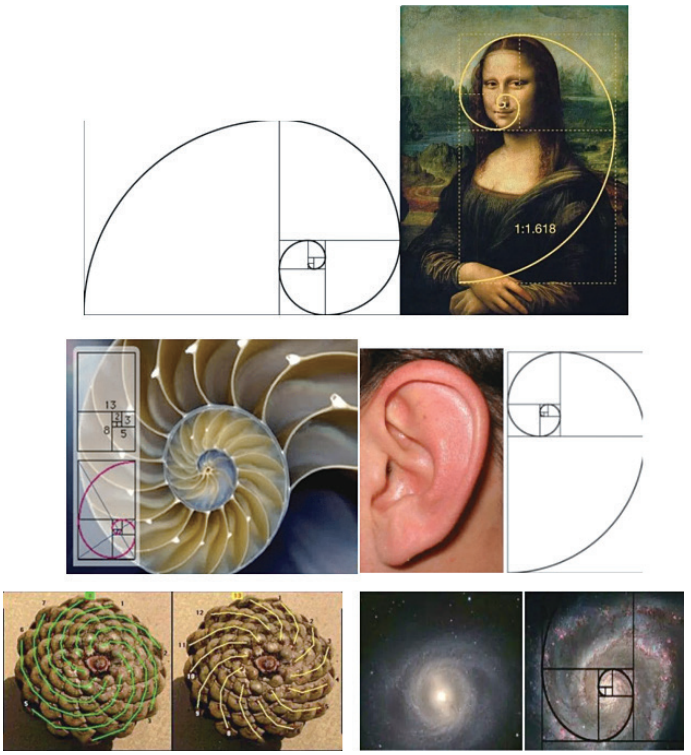

图 3-2　满足斐波那契数列排列规律的例子

视频资料：“斐波那契数列 .mp4”

2. 科赫雪花

所谓科赫雪花，也就是分形几何图形，如图 3-3 所示。

分形几何是一种迭代的几何图形，广泛存在于自然界中。

（1）科赫曲线的原理如图 3-4 所示。

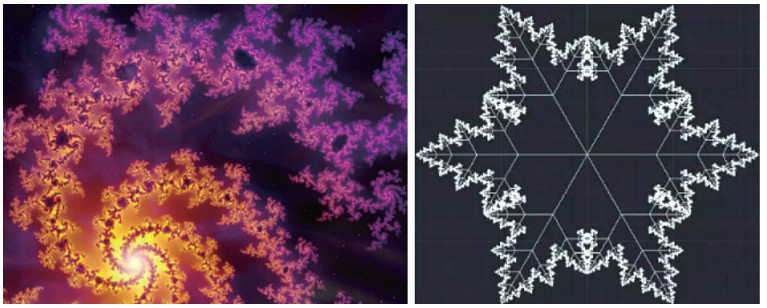

图 3-3　科赫雪花

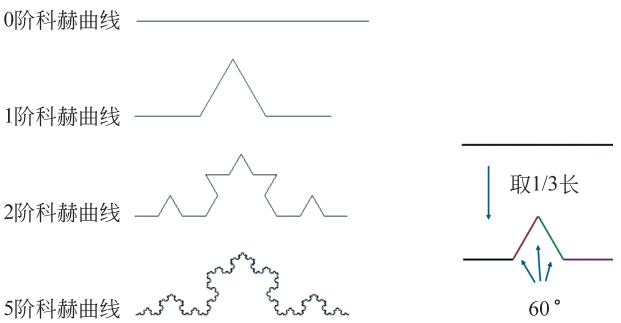

图 3-4　科赫曲线的原理

原理总结：

① n 阶科赫曲线怎么画出来？先旋转 0°，画 1/3 长的 n–1 阶科赫曲线，左转 60°，画 1/3 长的 n–1 阶科赫曲线，左转 –120°，画 1/3 长的 n–1 阶科赫曲线，左转 60°，画 1/3 长的 n–1 阶科赫曲线。

② 科赫曲线的边界是当 n 等于 0，画一条直线。

根据分析完成程序如下：

```
from turtle import *
def keh(n,size):
    if n==0:
        fd(size)
    else:
        angles=[0,60,-120,60]
        for angle in [0,60,-120,60]:
            left(angle)
            keh(n-1,size/3)
keh(3,300)
```

```
hideturtle()
done()
```

（2）将科赫曲线围成一个倒等边三角形，得到如图 3-5 所示的科赫雪花。

```
from turtle import *
def keh(n,size):
    if n==0:
        fd(size)
    else:
        angles=[0,60,-120,60]
        for angle in [0,60,-120,60]:
            left(angle)
            keh(n-1,size/3)
speed(0)
for i in range(3):
    keh(3,300)
    right(120)
hideturtle()
done()
```

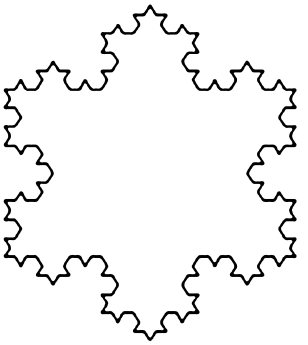

图 3-5　科赫雪花

3.7　教学组织安排

教 学 环 节	教 学 过 程	建议课时
知识导入	通过中国传统故事，了解故事中讲故事，从而了解程序递归的实际应用场景	1 课时
知识拓展	播放视频，了解递归的相关概念，引起学生的学习兴趣	
递归的概念	通过提问、讨论、测试、动手操作等互动及实践掌握递归的概念	
递归的简单实现	采用代码演示操作熟练掌握简单的递归实现方法	1 课时
单元总结	以提问方式总结本次课所学内容，布置课后作业	

3.8　教学实施参考

1. 讨论式知识导入

通过图 3-6 提问，引出从前有座山的故事……故事里不断提到同样的故事。引导学生查找自然界中的一个结构嵌套在另一个结构中，使同学们意识到递归的强大。

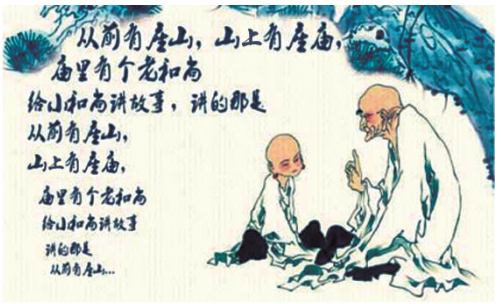

图 3-6　递归方式讲故事

2. 播放视频资料：“斐波那契数列 .mp4”

科普计算机编程中的斐波那契数列与大自然的关系，提高学生的学习兴趣。

3. 知识点一：什么是递归

（1）通过帮助小萌的提问使学生理解递归、代码封装的概念。

（2）讲解汉诺塔问题，需要移动五千多年，用递归实现起来却只有十几行代码，说明递归的优势。

（3）通过讲解例 3-1 中求 1+2+…+n 的和问题，编程实现过程。

（4）运行出错，提出递归边界问题。

（5）提出递归的两个关键特征。

（6）以问答的方式完成学生用书上的“想一想”中的问题 3-1，理解递归边界即递归链。

（7）以问答的方式完成学生用书上的“想一想”中的问题 3-2，理解递归的简洁。

（8）以测试的方式完成“练一练”中的问题 3-3，了解学生关于递归概念的掌握情况。

4. 知识点二：简单的递归实现

（1）通过例 3-2 用递归解决的问题：兔子问题，引入斐波那契数列及其代码实现。

（2）斐波那契数列用在兔子问题上，引出自然界的斐波那契数列，引出例 3-3 斐波那契数列的图形和黄金螺旋线。

（3）分析斐波那契曲线的算法，完成斐波那契数列的图形和螺旋线程序。

（4）以问答的方式，完成学生用书上的“想一想”中的问题 3-4，加深学生对斐波那契数列的图形和螺旋线的程序理解。

（5）以测试的方式完成“练一练”中的问题 3-5，测试学生关于递归程序的掌握情况。

5. 单元总结

小结本次课的内容，布置课后作业。

3.9 拓展练习

（1）使用 turtle 库和递归绘制如图 3-7 所示的二叉树。

参考代码如下：

```
from turtle import *
def tree(branch_len):
    if branch_len>5:
        fd(branch_len)
        right(20)
```

```
            tree(branch_len-15)
            left(40)
            tree(branch_len-15)
            right(20)
            fd(-branch_len)
    left(90)
    pencolor('green')
    speed(0)
    hideturtle()
    penup()
    goto(0,-100)
    pendown()
    tree(95)
    done()
```

（2）使用递归完成输入数据的逆序，并判断该数据是不是回文数据，例如，如图 3-8 所示的文字“人人为我 我为人人”正着看反着看都一样，还有数字 12321 等。

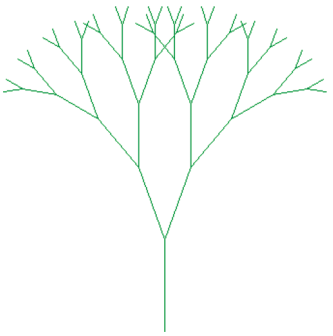

图 3-7　二叉树

图 3-8　人人为我 我为人人

程序代码如下。

```
def reverse(s):
    if s=="":
        return s
    else:
        return reverse(s[1:])+s[0]
s = input("输入要判断回文数的数据：")
if s == reverse(s):
```

```
    print(s,"是回文数据")
else:
    print(s,"不是回文数据")
```

运行结果 1：

```
输入要判断回文数的数据：人人为我 我为人人
人人为我 我为人人 是回文数据
```

运行结果 2：

```
输入要判断回文数的数据：我为人人
我为人人 不是回文数据
```

3.10 问 题 解 答

【问题 3-1】 递归函数如果没有边界条件，会内存溢出报错。

【问题 3-2】 递归函数的代码简单，不复杂。

【问题 3-3】 选 A。递归函数必须有边界条件，递归代码简单，不包含循环结构。

【问题 3-4】 将 for i in range(1,n+1) 改为 for i in range(n) 不行。如果改了，i 的值将从 0 开始，而斐波那契数列从第一项开始，不存在第 0 项。会出错。

【问题 3-5】 选 C。阶乘用递归实现 else 部分 n 的阶乘 return n 乘以 (n–1) 的阶乘，(n–1)! 用函数 jc(n–1)。

3.11 第 3 单元习题答案

1. A　2. C　3. A　4. D　5. C　6. C　7. B　8. C　9. A

10. C

11. 参考代码如下。

```
def mu2(n):
    if n in [1,2]:
        return n
    else:
        return mu2(n-1)*mu2(n-2)
print(mu2(6))
```

12. 参考代码如下。

```
def apple(m):
    n=2
    if m == 0:
        return 2
    else:
        return (apple(m-1)+1)*2
print(apple(3))
```

13. 参考代码如下。

```
def add2(n):
    if n in [1,2,3,4]:
        return n
    else:
        return add2(n-2)+add2(n-4)
print(add2(8))
```

本单元资源下载可扫描下方二维码。

课件 3　　扩展资源 3