# 1.3 递归简论

当一个函数用它自己定义时，就称为时递归(recursive)的

基准情况(base case)，即此时函数的值可以直接算出而不用求助递归。

Java的递归方法若无基准情况也是毫无意义的。

虽然我们定义一个方法用的是这个方法本身，但是我们并没有用方法本身定义该方法的一个特定实例。

所以，递归并不是循环推理(circular logic)

递归调用将反复进行直到基准情况出现

递归有两个基本法则：

1、基准情况 base case必须总要有些基准的情形，他们不用递归就能求解

2、不断推进 making progress 对于那些要递归求解的情形，递归调用必须总能够朝着一个基准情形推进

3、设计法则，假设所有的递归调用都能运行

4、合成效益法则，在求解一个问题的同一个实例时，切勿在不同的递归调用中作重复性的工作

# 1.4 实现泛型特性构建

1.4.1 使用Object表示泛型

1.4.2 基本类型的包装

1.4.3 使用接口类型表示泛型

1.4.4 数组类型的兼容性

# 1.5 利用Java5泛性实现反省特性成分

1.5.1 简单的泛性类和接口

1.5.2 自动装箱/拆箱

1.5.3 带有限制的通配符