

## 作业参考

递归的解法，一般来说有2种：

1. 一种如同数学公式
2. 一种类似循环，相当于循环的改版，将循环迭代，变成了函数调用压栈

### 1、n的阶乘

```
1 def factorial(n):
2     s = 1
3     for i in range(2, n+1):
4         s *= i
5     return s
6
7 def factorial(n):
8     s = 1
9     for i in range(n, 1, -1):
10        s *= i
11    return s
```

```
1 # 循环 变 递归。三元表达式自己实现
2 def factorial2(n, p=1): # 需要提供一个形参传入上一次
3     # 循环中的循环体
4     if n == 1: # 边界条件。1就不用乘了，直接返回
5         return p # 直到现在的乘积
6
7     # 进行下一次计算，使用函数调用实现
8     return factorial2(n-1, p * n)
```

```
1 # 阶乘公式。三元表达式自己实现
2 def factorial1(n):
3     if n < 2:
4         return 1
5     return factorial1(n-1) * n
```

### 2、猴子吃桃问题

猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个。第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第10天早上想吃时，只剩下一个桃子了。求第一天共摘多少个桃子

```
1 思路：
2 假设猴子摘了x个桃
3
4 d1 x //2 - 1
5 d2 d1//2 - 1
6 d3 d2//2 - 1
7 ...
8 d9 d8//2 - 1
9 d10 1
```

```
1 # 循环实现
2 peach = 1
3 days = 9
4
5 for i in range(days):
6     peach = 2 * (peach + 1)
7
8 print(peach)
9
10 # 改版的递归实现
11 def fn(days=9, peach=1):
12     peach = 2 * (peach + 1)
13     if days == 1: # 保证计算过days次, 9天就是9次
14         return peach
15     return fn(days-1, peach)
16
17 print(fn())
```

```
1 def peach(days=10): # 虽然是10, 但是计算了9次, 最后一次是为了等于1触底反弹的
2     if days == 1:
3         return 1
4     return 2 * (peach(days-1) + 1)
5
6 print(peach())
```