剑指offer python版-9-12

9变态跳台阶

题目：一只青蛙一次可以跳上1级台阶，也可以跳上2级……它也可以跳上n级。求该青蛙跳上一个n级的台阶总共有多少种跳法

思路：当前台阶的跳法种数 = 当前台阶之前的所有台阶的跳法之和   
即：f (n) = f (n-1) + f (n-2) + f (n-3) + … + f (2) + f (1)

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** jumpFloorII(self, number):
4. # write code here
5. **if** number<3:
6. **return** number
7. a=[0,1,2]
8. **for** i **in** range(3,number+1):
9. a.append(sum(a)+1)
10. **return** a[number]

10矩阵覆盖

题目：我们可以用2\*1的小矩形横着或者竖着去覆盖更大的矩形。请问用n个2\*1的小矩形无重叠地覆盖一个2\*n的大矩形，总共有多少种方法？

思路：这是Fibonacci数列的变形

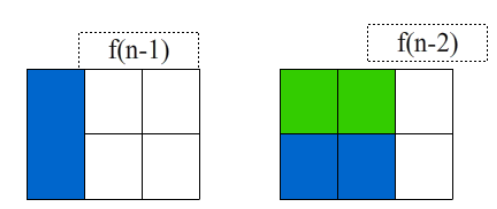
当n=1时，记作F(1), 共有1种方法；

当n=2时，记作F(2), 共有两种方法；

当n=3时，记作F(3), 分为两种情况：

第一次用一个矩形竖着覆盖（左图蓝色），则剩下共有F(n-1)种方法，即F(2) 种方法；

第一次用一个矩形横着覆盖（右图蓝色），则剩下绿色区域只能有图示一种方法，那么剩下F(n-2)种方法，即F(1) 种方法；



代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** rectCover(self, number):
4. # write code here
5. num=[];
6. num.append(0);
7. num.append(1);
8. num.append(2);
9. **for** i **in** range(3 , number+1):
10. num.append(num[i-1]+num[i-2]);
11. **return** num[number];

11二进制中1的个数

题目：输入一个整数，输出该数二进制表示中1的个数。其中负数用补码表示。

思路1：可以将n上的每一位与1取与，统计1的个数；在求解的过程中每次循环时1左移一位，即n<<1

思路2：如果一个整数不为0，那么这个整数至少有一位是1。如果我们把这个整数减1，那么原来处在整数最右边的1就会变为0，原来在1后面的所有的0都会变成1(如果最右边的1后面还有0的话)。其余所有位将不会受到影响

把原来的整数和减去1之后的结果做与运算，从原来整数最右边一个1那一位开始所有位都会变成0

举例来说，6的二进制是 110 ，6-1=5的二进制是 101，6&5=100， 如此操作之后6中原来的110变为100，循环计数统计1的个数，直至n变为0为止。

\*\*到了Python3，当长度超过32位或64位之后，Python3会自动将其转为长整型，长整型理论上没有长度限制。

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** NumberOf1(self, n):
4. # write code here
5. count=0
6. **while** n&0xffffffff !=0:
7. count+=1
8. n=n&(n-1)
9. **return** count

12数值的整数次方

题目：给定一个double类型的浮点数base和int类型的整数exponent。求base的exponent次方

思路：分类讨论就可以

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** Power(self, base, exponent):
4. # write code here
5. **if** exponent==0:
6. **return** 1
7. exp=abs(exponent)
8. ans=1.0
9. **for** i **in** range(0,exp):
10. ans=ans\*base
11. **if** exponent<0:
12. **return** 1/ans
13. **else**:
14. **return** ans