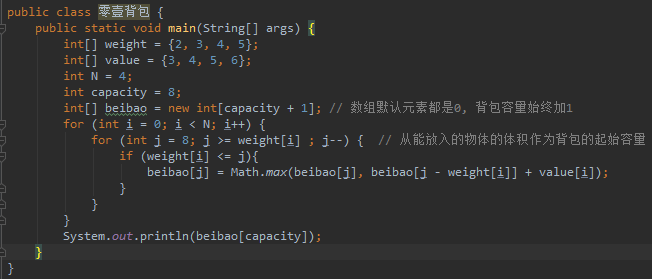
# 1、动态规划入门到熟悉

<https://blog.csdn.net/hebtu666/article/details/100585136>

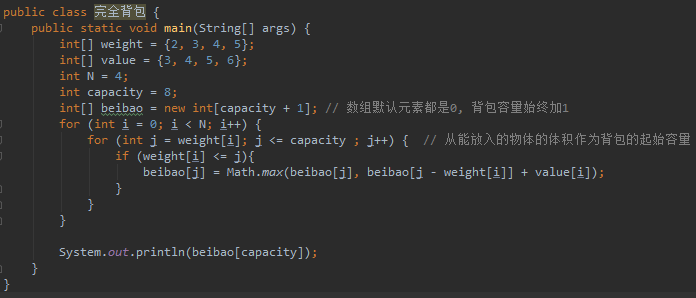
## 0/1背包问题：

思路：因为每个物体只能取一次，且用的是一维数组，所以背包容量从大到小，因为这样能保证数组左边的元素是前面的的物体组成的和，与该物体无关，这样当放入该物体后，求剩余背包容量所能放置的最大价值，且背包容量到能够放入该物体即可，无需到0.



## 完全背包问题：（每个物体可以去无限个）

思路：相比于0/1背包，其实只需将第二个循环的顺序调转过来即可。好好体会。



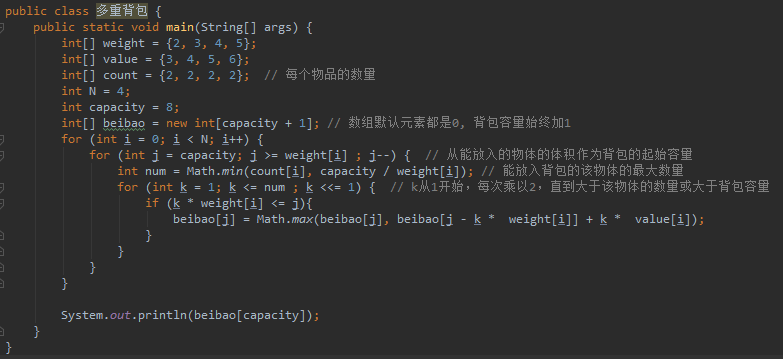
## 多重背包问题：（每个物体只能取有限个）

思路：1、将多个物体拆成一个一个物体，用01背包求解

1. 将多个物体拆成二进制数的物体堆，用01背包求解

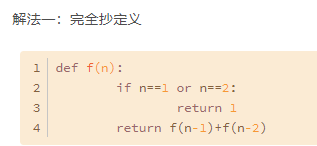
举例：15个物体拆成15个，可以将1-15个物体放入背包，15个物体才成1、2、4、8四堆，同样可以表示成1-15个物体放入背包。

注意：虽然需要上面的数进行组合，但是还是要从大到小，因为1、2、4、8不能重复取数。

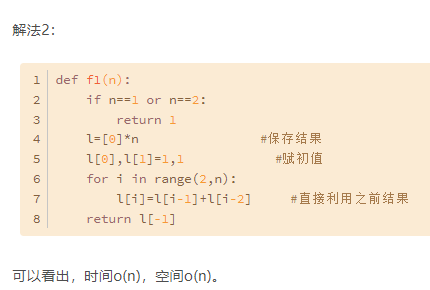


## 斐波那契数列系列问题：

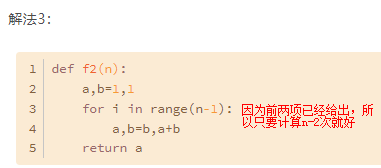
**思路：如果只是求第n项，以下三种方法，从坏到好：**



该解法效率极低，因为每次求一个f（n）都要求出f（n...1）, 出现了大量的重复计算



该解法保存了中间值，避免了重复计算，但如果只需要求出第n项，那么是不是空间有点浪费呢？所以有了下面的解法：



**青蛙跳台阶和爬楼梯问题：（只能跳、爬一阶或两阶）**

思路：代码只是将刚开始的两个值从1,1改为1,2。

动态规划思想：到第n阶的情况只有两种情况：从第n-1阶迈一步；从第n-2阶迈两步

所以f（n） = f（n - 1） + f（n - 2）

**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？**

动态规划思想：覆盖到长度为n的地方有两种情况：

在第n个位置竖着放一个小矩形；

从第n-1个位置横着放两个小矩形；

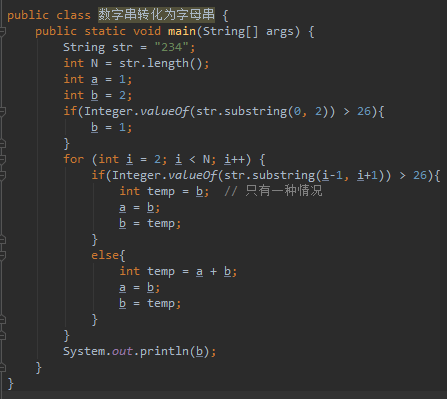
所以f（n） = f（n - 1） + f（n - 2）

**给定一个由0-9组成的字符串，1可以转化成A，2可以转化成B。依此类推。。25可以转化成Y，26可以转化成z，给一个字符串，返回能转化的字母串的有几种？**

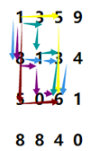
动态规划思想：第n个数字字符转换完成字母分为两种情况：

第n个位置，也就是自身转化为字母

结合第n-1个位置，转换为字母（两位数不能大于26）



## 矩阵走路问题

从矩阵L左上角到右下角经过的路径的最小和

状态转移方程：dp[j] = min(dp[j-1], dp[j]) + L[i][j]。

