# 问题

现在假设有 n 个货物，其中有一个货物是次品，次品比正常的要轻一些，你只有一个天平，请问，至少需要称多少次能保证一定找到次品？

这其实是一道非常简单的问题，但是要是之前没遇到过还真有可能不知道怎么下手，通常就是采用对半分的方法了，但这个思路是错的。

我第一次碰到这个题目是在家辅导小学的妹妹做数学题的时候，那个时候我想的就是对半分，然后错了，然后就被鄙视了。后来在一些公司笔试以及面经当中居然又碰到，所以感觉这种题目考察地挺多的，还有有必要梳理一下解题的思路和套路。

# 解题思路

其实解题方法很简单，就是不断地３等分即可，那为什么是３等分，而不是４等分或者２等分呢？你想想，如果是三等分的话，假设分成 A、B、C三组，阿么可以先比较 A 和 B，如果次品在这两组中，那么后续就只要在轻的那一组中继续寻找即可，如果 A 和 B相等，那么次品就肯定在 C 中，后续只要在 C 中找即可，也就是说称一次可以排除掉三分之二的可能性。

而要是２等分的话，称一次只能排除掉二分之一的可能性。４等分的话，就不好说了，可能性很多，最差的情况是比３等分差的。

所以只要不断地３等分即可。但是有一个问题是每一步中要从中寻找次品的货物集合的数量 n 可能不是３的倍数，所以并不能完全地等分，一共有三种情况。

1. n % 3 = 0
2. n % 3 = 1
3. n % 3 = 2

## 第一种情况：n % 3 = 0

这种情况就很简单，直接三等分即可。例如 n = 21，那么三等分就是 7 、7、 7

## 第二种情况：n % 3 = １

例如 n = 22时，这时候应该这么分： 7 、7、8。也就是前面两组数量相同，这样便可以先把这两组放到天平上去对比。第三组的数量比前面两组多一个。那么这个数字是怎么计算出来的呢？

前面两组的数量 m = (n - 1) / 3

那么第三组的数量便是 m + 1

## 第三种情况：n % 3 = 2

例如 n = 23时，这时候应该这么分： 8、8、7。也就是前面两组数量相同，这样便可以先把这两组放到天平上去对比。第三组的数量比前面两组少一个。那么这个数字又是怎么计算出来的呢？

前面两组的数量 m = (n + 1) / 3

那么第三组的数量便是 m - 1

# 总结

所以你需要不断三等分，然后下次迭代时选择数量组多的那组进行迭代，这样得到的次数便是符合要求的答案。例如 n = 22时，这时候应该这么分： 7 、7、8，然后下一次把 8 分成 3 、3、 2，然后下一次把3 分成 1 、1、 1。如此类推。

直到货物堆只剩1个，就能找到答案。

答案为 $log_3(n)$ 往下取整。

# 代码实现

网上看到一段代码，也摘抄下来，说不定面试官就叫你实现这个算法呢。

```
int end(int n)
{
    int ans = 0;
    if(n % 3 == 0)
        n--;
    while(n > 0)
    {
        n /= 3;
        ans++;
    }
    return ans;
}
```

可能你不理解为什么上面需要n--

那是因为，当n为3的指数倍时，如9，总有9/3 = 3, 9/3=1，剩下1一个本应该停止，但是还是继续进入循环。。变成3次

所以使用n--来减少一次次数。因为4~8进入循环返回的ans都为2。虽然6满足6%3==0但不会出问题，因为6跟5是返回ans都为2。

特殊的只有n的指数倍，但刻意去找n的指数太浪费时间，所以才使用n%3==0来排除，使用这个也不会影响6, 12等虽然满足3的整数倍但不是n的指数倍的数的问题

# 参考资料

[用天平找次品的算法题，即三等分算法]