



链滴

Counting Bits

作者: [moloe](#)

原文链接: <https://ld246.com/article/1502292152153>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

题目

Given a non negative integer number **num**. For every numbers **i** in the range **0 ≤ i ≤ num** calculate the number of 1's in their binary representation and return them as an array.

Example:

For **num = 5** you should return **[0,1,1,2,1,2]**.

讨论：

一般情况下会采用**O(n*sizeof(integer))**的时间复杂度来实现，主要是循环下，然后按位判断。而**O(n)**杂度的实现就需要一点思路了。

先看一个稍微更可能想到的解决方案：

4-7可以由0-3的高位加一个1，类比0-1|2-3，0-7|8-15，所以这里就存在一个规律

实现借鉴如下：

```
public int[] countBits(int num) {  
    int[] ret = new int[num+1];  
    ret[0] = 0;  
    int pow = 1; // 2的幂 1,2,4,8..  
    for(int i = 1, t = 0; i <= num; i++, t++) {  
        if(i == pow) { // 每当达到 pow 时，将 t=>0, pow 到 2*pow-1 就可以从 0-(pow-1) 来推算出  
            pow *= 2;  
            t = 0;  
        }  
        ret[i] = ret[t] + 1; // 由此推算  
    }  
    return ret;  
}
```

而更进一步有

$$f[i] = f[i / 2] + i \% 2$$

所以精简版如下：

```
public int[] countBits(int num) {  
    int[] f = new int[num + 1];  
    for (int i=1; i<=num; i++)  
        f[i] = f[i >> 1] + (i & 1);  
    return f;  
}
```

精妙。

实现取自该题 discuss