# 题目

编写一个函数，输入是一个无符号整数，返回其二进制表达式中数字位数为 ‘1’的个数（也被称为汉明重量）。

**示例 1：**

输入：00000000000000000000000000001011

输出：3

解释：输入的二进制串00000000000000000000000000001011中，共有三位为'1'。

**示例 2：**

输入：00000000000000000000000010000000

输出：1

解释：输入的二进制串00000000000000000000000010000000 中，共有一位为 '1'。

**示例 3：**

输入：11111111111111111111111111111101

输出：31

解释：输入的二进制串 11111111111111111111111111111101 中，共有 31 位为 '1'。

**提示：**

请注意，在某些语言（如Java）中，没有无符号整数类型。在这种情况下，输入和输出都将被指定为有符号整数类型，并且不应影响您的实现，因为无论整数是有符号的还是无符号的，其内部的二进制表示形式都是相同的。

在Java中，编译器使用二进制补码记法来表示有符号整数。因此，在上面的 示例3中，输入表示有符号整数-3。

**进阶:**

如果多次调用这个函数，你将如何优化你的算法？

# 分析

## 方法一：按位与

根据与运算定义，设二进制数字n，则有：

若n & 1 =0，则n二进制最右一位为0；

若n & 1 =1，则n二进制最右一位为1。

根据以上特点，考虑以下循环判断：

判断n最右一位是否为1，根据结果计数。

将n右移一位（本题要求把数字n看作无符号数，因此使用无符号右移操作）

注：凡是计算1的个数的都是与操作，凡是计算0的个数的都是或操作。

**代码：**

class Solution {

public:

int hammingWeight(uint32\_t n) {

int num = 0;

for(int i=0;i<32;i++){

if(n & (1<<i)) //而不是i<<1

num++;

}

return num;

}

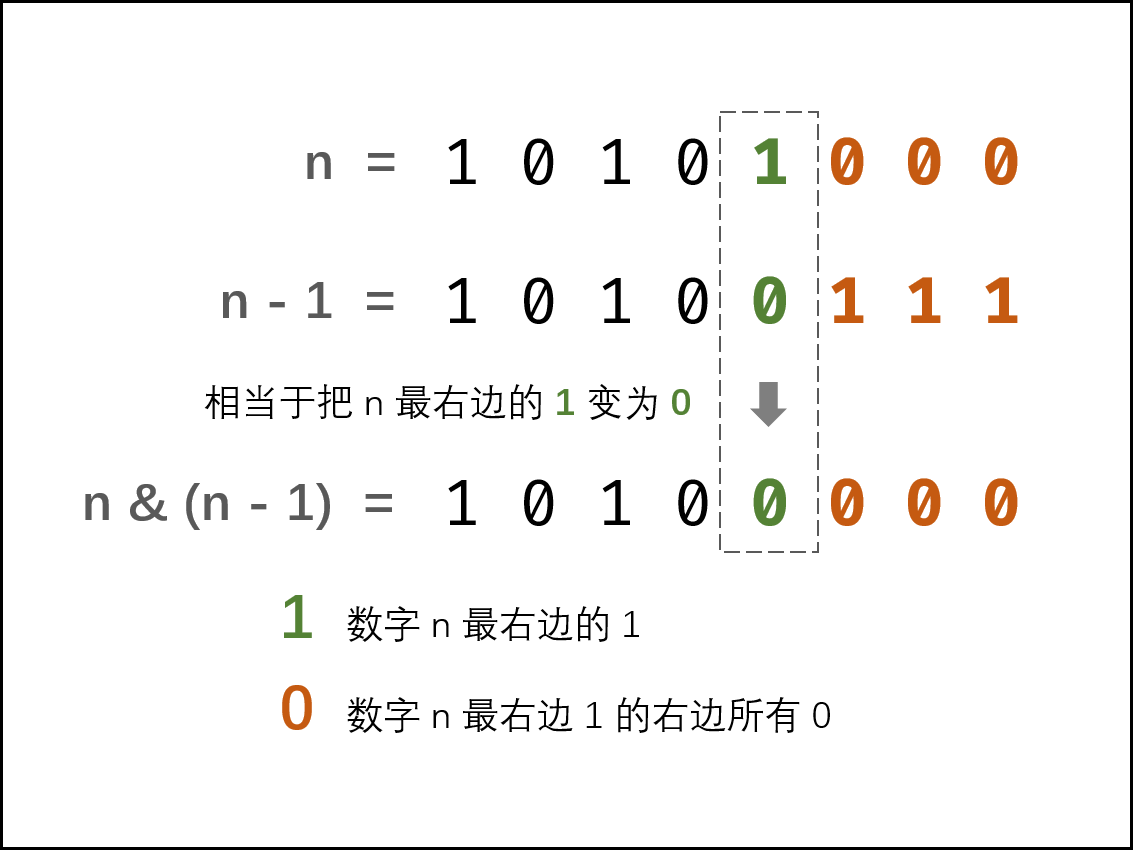
};

## 方法二：巧用n&(n−1)

思路：

(n−1) 解析：二进制数字n最右边的1变成0，此1右边的0都变成1。

n&(n−1) 解析：二进制数字n最右边的1变成0，其余不变。



1、初始化数量统计变量res。

2、循环消去最右边的1：当n=0时跳出。

2.1、res += 1：统计变量加1；

2.2、n &= n - 1：消去数字n最右边的1。

3、返回统计数量res 。

代码：

public class Solution {

public int hammingWeight(int n) {

int res = 0;

while(n != 0) {

res++;

n &= n - 1;

}

return res;

}

}