题目 二进制中1的个数

考点 位运算 热点指数 85618 通过率 34.00%

具体题目

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。

绝对最佳答案及分析：

```
  public class Solution {
      public int NumberOf1(int n) {
          int count = 0;
        while(n!= 0){
            count++;
            n = n & (n - 1);
         }
           return count;
      }
  }
```

答案正确:恭喜！您提交的程序通过了所有的测试用例 分析一下代码： 这段小小的代码，很是巧妙。 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。 举个例子：一个二进制数1100，从右边数起第三位是处于最右边的一个1。减去1后，第三位变成0，它后面的两位0变成了1，而前面的1保持不变，因此得到的结果是1011.我们发现减1的结果是把最右边的一个1开始的所有位都取反了。这个时候如果我们再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。如1100&1011=1000.也就是说，把一个整数减去1，再和原整数做与运算，会把该整数最右边一个1变成0.那么一个整数的二进制有多少个1，就可以进行多少次这样的操作。

```
public class Solution {
    public int NumberOf1(int n) {
        int result=0;
        int test=n;
        while (test!=0){
            test&=(test-1);
            result++;
        }
        return result;
    }
}
```

/**

- 思路：将n与n-1想与会把n的最右边的1去掉，比如
- 1100&1011 = 1000
- 再让count++即可计算出有多少个1
- @author skyace * */

```
public class CountOne {
     public static int NumberOf1(int n) {
            int count = 0;
            while(n!=0){
                count++;
                n = n&(n-1);
            }

            return count;
        }
}
```