

## Day8-1原码，反码，补码的介绍

### 原码：

就是一个二进制数字，从“数学观念”上表达出来的形式，其中，我们规定：

一个数字的最左边的一位是“符号位” 0表示正数，1表示负数：

比如：

5的原码：			
00000000	00000000	00000000	00000101
-3的原码：			
10000000	00000000	00000000	00000011

### 反码：

正数的反码就是其本身（即不变）；

负数的反码：就是符号位不变，其他位取反；

5的3码：			
原码：00000000	00000000	00000000	00000101
反码：00000000	00000000	00000000	00000101
-3的3码：			
原码：10000000	00000000	00000000	00000011
反码：11111111	11111111	11111111	11111001

### 补码：

正数的补码就是其本身（即不变）；

负数的补码：就是符号位不变，其他位取反后加1，即反码加1；

5的3码：			
原码：00000000	00000000	00000000	00000101
反码：00000000	00000000	00000000	00000101
补码：00000000	00000000	00000000	00000101
-3的3码：			
原码：10000000	00000000	00000000	00000011
反码：11111111	11111111	11111111	11111001
补码：11111111	11111111	11111111	11111010

一个小提示：

计算机内部的运算，实际上全都是使用补码进行的，而且运算的时候，符号位也不再区分，直接也当做“数据”参与运算：

实例1：

5+3：

演示5+3的cpu运算“过程”：			
5的补码：	00000000	00000000	00000000
3的补码：	11111111	11111111	11111111
+			
结果：	00000000	00000000	00001000
可见结果是：2的3次方，即8；			

实例2：

5-3：

实际上，cpu内部，会将“减法”运算，转换为“加法”运算，即，5+(-3)

演示5-3的cpu运算“过程”：实际进行的是5+（-3）

5的补码：      00000000      00000000      00000000      00000101

-3的补码：    11111111      11111111      11111111      11111101

+

-----  
结果：            00000000      00000000      00000000      00000010

可见结果是：2