

没有数学就没有算法

新知一下

# 数论初步

NOI 基础算法系列课程

---

版本: 1.0.0

讲师: 孙伟航

The background of the slide features a high-angle view of the Earth from space, showing the curvature of the planet and the cloud-covered surface. Overlaid on this is a network of thin, white lines connecting various points, resembling a global communication or data network. Several bright, star-like light sources are scattered across the scene, adding a sense of depth and technological sophistication.

# 01

机器数

一个数在计算机中的二进制表示形式, 叫做这个数的**机器数**。机器数是带符号的, 在计算机用一个数的最高位存放符号, 正数为0, 负数为1。

比如, 十进制中的数 +3 , 计算机字长为8位, 转换成二进制就是00000011。如果是 -3 , 就是10000011 。

那么, 这里的 00000011 和 10000011 就是机器数。



因为第一位是符号位，所以机器数的形式值就不等于真正的数值。例如上面的有符号数10000011，其最高位1代表负，其真正数值是 -3 而不是形式值131（10000011转换成十进制等于131）。所以，为区别起见，将带符号位的机器数对应的真正数值称为机器数的真值。  
例：0000 0001的真值 = +000 0001 = +1，1000 0001的真值 = -000 0001 = -1



# 原码

原码就是符号位加上真值的绝对值, 即用第一位表示符号, 其余位表示值。

优点: 简单直观; 例如, 我们用8位二进制表示一个数, +11的原码为00001011, -11的原码就是10001011

缺点: 原码不能直接参加运算, 可能会出错。例如数学上,  $1+(-1)=0$ , 而在二进制中原码  $00000001+10000001=10000010$ , 换算成十进制为-2。显然出错了。



# 反码

反码通常是用来由原码求补码或者由补码求原码的过渡码

正数的反码是其本身

负数的反码是在其原码的基础上, 符号位不变, 其余各个按位取反.

$[+1] = [00000001]_{\text{原}} = [00000001]_{\text{反}}$

$[-1] = [10000001]_{\text{原}} = [11111110]_{\text{反}}$



# 补码

正数的补码就是其本身

负数的补码是在其原码的基础上, 符号位不变, 其余各位取反, 最后+1. (即在反码的基础上+1)

$[+1] = [00000001]_{\text{原}} = [00000001]_{\text{反}} = [00000001]_{\text{补}}$

$[-1] = [10000001]_{\text{原}} = [11111110]_{\text{反}} = [11111111]_{\text{补}}$



# 机器数的意义

思路：用加法代替减法，例如  $1 + (-1) = 1 - 1 = 0$ ;

$$1 - 1 = 1 + (-1) = [00000001]_{\text{原}} + [10000001]_{\text{原}} = [10000010]_{\text{原}} = -2$$

$$1 - 1 = 1 + (-1) = [0000\ 0001]_{\text{原}} + [1000\ 0001]_{\text{原}} = [0000\ 0001]_{\text{反}} + [1111\ 1110]_{\text{反}} = [1111\ 1111]_{\text{反}} = [1000\ 0000]_{\text{原}} = -0。$$

$$1 - 1 = 1 + (-1) = [0000\ 0001]_{\text{原}} + [1000\ 0001]_{\text{原}} = [0000\ 0001]_{\text{补}} + [1111\ 1111]_{\text{补}} = [000\ 0000]_{\text{补}} = [0000\ 0000]_{\text{原}}$$

数学原理：同余，不再扩展。

## 小提示：

算术运算对象与结果在机器内部都是以补码的形式存储的，只是在输出时被还原成原码。

所有的数学运算在机器内部都是以补码的形式进行计算的。

由于计算机运算器的特性，所以最高位进位时，不会溢出，会被直接舍弃。原理：场运算，不扩展。

原码和反码能够表示的范围是-127到127，共255个。

补码能够表示的范围是-128到127共256个。因为补码中没有-0的概念，所以10000000就被表达为-128。

$$-128 + 1 = -127$$

$$-128 = 1000\ 0000$$





# 感谢观看

联系地址：河北省廊坊市大厂县孔雀英国宫二期

