四种常用的数制及它们之间的相互转换：

| 进制 | 基数 | 基数个数 | 权 | 进数规律 |
|---|---|---|---|---|
| 十进制 | 0、1、2、3、4、5、6、7、8、9 | 10 | $10^i$ | 逢十进一 |
| 二进制 | 0、1 | 2 | $2^i$ | 逢二进一 |
| 八进制 | 0、1、2、3、4、5、6、7 | 8 | $8^i$ | 逢八进一 |
| 十六进制 | 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F | 16 | $16^i$ | 逢十六进一 |

十进制数转换为二进制数、八进制数、十六进制数的方法：

二进制数、八进制数、十六进制数转换为十进制数的方法：**按权展开求和法**

1．二进制与十进制间的相互转换：

（1）二进制转十进制

方法：**“按权展开求和”**

例：　$(1011.01)_2 = (1\times2^3+0\times2^2+1\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2})_{10}$

$= (8+0+2+1+0+0.25)_{10}$

$= (11.25)_{10}$

规律：个位上的数字的次数是0，十位上的数字的次数是1，.....，依奖递增，而十分位的数字的次数是-1，百分位上数字的次数是-2，.....，依次递减。

注意：不是任何一个十进制小数都能转换成有限位的二进制数。

（2）十进制转二进制

- 十进制整数转二进制数：“除以2取余，**逆序**排列”（短除反取余法）

例：　$(89)_{10} = (1011001)_2$

| | | |
|---|---|---|
| 2 | 89 | |
| 2 | 44 | ……1 |
| 2 | 22 | ……0 |
| 2 | 11 | ……0 |
| 2 | 5 | ……1 |
| 2 | 2 | ……1 |
| 2 | 1 | ……0 |
| | 0 | ……1 |

- 十进制小数转二进制数：“乘以2取整，**顺序**排列”（乘2取整法）

例：　$(0.625)_{10} = (0.101)_2$

| | | |
|---|---|---|
| | 0．625 | |
| X | 2 | |
| | 1．25 | 1 |
| X | 2 | |
| | 0．5 | 0 |
| X | 2 | |
| | 1．0 | 1 |

2．八进制与二进制的转换：

**二进制数转换成八进制数**：从小数点开始，整数部分向左、小数部分向右，每3位为一组用一位八进制数的数字表示，不足3位的要用“0”补足3位，就得到一个八进制数。

**八进制数转换成二进制数**：把每一个八进制数转换成3位的二进制数，就得到一个二进制数。

例：将八进制的37.416转换成二进制数：

| | | | | | |
|---|---|---|---|---|---|
| 3 | 7 | . | 4 | 1 | 6 |
| 011 | 111 | . | 100 | 001 | 110 |

即：（37.416）$_8$ ＝（11111.10000111）$_2$

例：将二进制的 10110.0011 转换成八进制：

010 110 0.0 011 000

2 6 . 1 4

即：（10110.011）$_2$ ＝ （26.14）$_8$

3．十六进制与二进制的转换：

**二进制数转换成十六进制数**：从小数点开始，整数部分向左、小数部分向右，每 4 位为一组用一位十六进制数的数字表示，不足 4 位的要用“0”补足 4 位，就得到一个十六进制数。

**十六进制数转换成二进制数**：把每一个八进制数转换成 4 位的二进制数，就得到一个二进制数。

例：将十六进制数 5DF.9 转换成二进制：

5 D F . 9

0101 1101 1111 . 1001

即：（5DF.9）$_{16}$ ＝（10111011111.1001）$_2$

例：将二进制数 1100001.111 转换成十六进制：

0110 0001 . 1110

6 1 . E

即：（1100001.111）$_2$ ＝（61.E）$_{16}$

注意：以上所说的二进制数均是无符号的数。这些数的范围如下表：

| 无符号位二进制数位数 | 数值范围 | 十六进制范围表示法 |
|---|---|---|
| 8 位二进制数 | 0~255 （255=$2^8$-1） | 00~0FFH |
| 16 位二进制数 | 0~65535 （65535=$2^{16}$-1） | 0000H~0FFFFH |
| 32 位二进制数 | 0~$2^{32}$-1 | 00000000H~0FFFFFFFFH |

**带符号数的机器码表示方法**

1．带符号二进制数的表示方法：

带符号二进制数用最高位的一位数来表示符号：0 表示正，1 表示负。

| 含符号位二进制数位数 | 数值范围 | 十六进制范围表示法 |
|---|---|---|
| 8 位二进制数 | -128 ~ +127 | 80H~7FH |
| 16 位二进制数 | -32768 ~ +32767 | 8000H~7FFFH |
| 32 位二进制数 | -2147483648 ~ +2147483647 | 80000000H~7FFFFFFFH |

2、符号位的表示：最常用的表示方法有原码、反码和补码。

（1）原码表示法：一个机器数 x 由符号位和有效数值两部分组成，设符号位为 $x_0$，x 真值的绝对值|x|=$x_1x_2x_3...x_n$，则 x 的机器数原码可表示为：

$[x]_{原}=x_0x_1x_2...x_n$，当 x>=0 时，$x_0$=0，当 x<0 时，$x_0$=1。

例如：已知：$x_1$=-1011B，$x_2$= +1001B，则 $x_1$，$x_2$ 有原码分别是

$[x_1]_{原}$=11011B，$[x_2]_{原}$=01001B

规律：正数的原码是它本身，负数的原码是取绝对值后，在最高位（左端）补“1”。

（2）反码表示法：一个负数的原码符号位不变，其余各位按位取反就是机器数的反码表示法。正数的反码与原码相同。

按位取反的意思是该位上是 1 的，就变成 0，该位上是 0 的就变成 1。即 $\overline{1}$=0，$\overline{0}$=1

例：$x_1=-1011B$，$x_2=+1001B$，求$[x_1]_{反}$和$[x_2]_{反}$。

解：$[x_1]_{反}=10100B$，$[x_2]_{反}=01001B$

（3）补码表示法：

首先分析两个十进制数的运算：78-38=41，79+62=141

如果使用两位数的运算器，做 79+62 时，多余的 100 因为超出了运算器两位数的范围而自动丢弃，这样在做 78-38 的减法时，用 79+62 的加法同样可以得到正确结果。

模是批一个计量系统的测量范围，其大小以计量进位制的基数为底数，位数为指数的幂。如两位十进制数的测量范围是 1——9，溢出量是 100，模就是 $10^2$=100，上述运算称为模运算，可以写作：

79+（-38）=79+62　（mod　100）

进一步写为　-38=62，此时就说　- 38 的补法（对模 100 而言）是 62。计算机是一种有限字长的数字系统，因此它的运算都是有模运算，超出模的运算结果都将溢出。n 位二进制的模是 $2^n$,

一个数的补码记作[x]$_{补}$，设模是 M，x 是真值，则补码的定义如下：

$$[x]_{补}=\begin{cases}[x]_{原} & (x\geq 0)\\ M+x & (x<0)\end{cases}$$

例：设字长 n=8 位，x=-1011011B，求[x]$_{补}$。

解：因为　n=8，所以模 M=$2^8$=100000000B，x<0，所以

[x]$_{补}$=M+x=100000000B-1011011B=10100101B

注意：这个 x 的补码的最高位是“1”，表明它是一个负数。对于二进制数还有一种更加简单的方法由原码求出补码：

（1）正数的补码表示与原码相同；

（2）负数的补码是将原码符号位保持“1”之后，其余各位按位取反，末位再加 1 便得到补码，即取其原码的反码再加“1”：[x]$_{补}$=[x]$_{反}$+1。

下表列出$\pm 0,\pm 39,\pm 127$及$-128$的 8 位二进制原码，反码和补码并将补码用十六进制表示。

| 真值 | 原码（B） | 反码（B） | 补码（B） | 补码（H） |
|---|---|---|---|---|
| +127 | 0 111 1111 | 0 111 1111 | 0 111 1111 | 7F |
| +39 | 0 010 0111 | 0 010 0111 | 0 010 0111 | 27 |
| +0 | 0 000 0000 | 0 000 0000 | 0 000 0000 | 00 |
| -0 | 1 000 0000 | 1 111 1111 | 0 000 0000 | 00 |
| -39 | 1 010 0111 | 1 101 1000 | 1 101 1001 | D9 |
| -127 | 1 111 1111 | 1 000 0000 | 1 000 0001 | 81 |
| -128 | 无法表示 | 无法表示 | 1 000 0000 | 80 |

从上可看出，真值+0 和-0 的补码表示是一致的，但在原码和反码表示中具有不同形式 。8 位补码机器数可以表示-128，但不存在+128 的补码与之对应，由此可知，8 位二进制补码能表示数的范围是-128——+127。还要注意，不存在-128 的 8 位原码和反码形式。

**定点数和浮点数**

（一）定点数（*Fixed-Point Number*）

计算机处理的数据不仅有符号，而且大量的数据带有小数，小数点不占有二进制一位而是隐含在机器数里某个固定位置上。通常采取两种简单的约定：一种是约定所有机器数的小数的小数点位置隐含在机器数的最低位之后，叫定点纯整机器数，简称定点整数。另一种约定所有机器数的小数点隐含在符号位之后、有效部分最高位之前，叫定点纯小数机器数，简称定点小数。无论是定点整数，还是定点小数，都可以有原码、反码和补码三种形式。

（二）浮点数（*Floating-Point Number*）

计算机多数情况下采作浮点数表示数值，它与科学计数法相似，把一个二进制数通过移动小数点位置表示成阶码和尾数两部分：

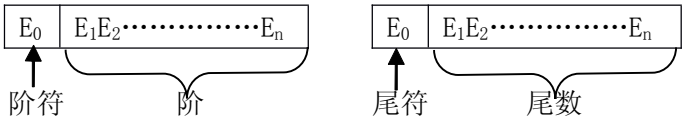
$$N = 2^E \times S$$

其中：E——N 的阶码（Expoent），是有符号的整数

S——N 的尾数（Mantissa），是数值的有效数字部分，一般规定取二进制定点纯小数形式。

例：1011101B=$2^{+7}$*0.1011101，101.1101B=$2^{+3}$*0.1011101，0.01011101B=$2^{-1}$*0.1011101

浮点数的格式如下：




浮点数由阶码和尾数两部分组成，底数 2 不出现，是隐含的。阶码的正负符号 $E_0$，在最前位，阶反映了数 N 小数点的位置，常用补码表示。二进制数 N 小数点每左移一位，阶增加 1。尾数是这点小数，常取补码或原码，码制不一定与阶码相同，数 N 的小数点右移一位，在浮点数中表现为尾数左移一位。尾数的长度决定了数 N 的精度。尾数符号叫尾符，是数 N 的符号，也占一位。

例：写出二进制数-101.1101B 的浮点数形式，设阶码取 4 位补码，尾数是 8 位原码。

$-101.1101=-0.1011101*2^{+3}$

浮点形式为：

阶码 0011　　尾数 11011101

补充解释：阶码 0011 中的最高位“0”表示指数的符号是正号，后面的“011”表示指数是“3”；尾数 11011101 的最高位“1”表明整个小数是负数，余下的 1011101 是真正的尾数。

例：计算机浮点数格式如下，写出 x=0.0001101B 的规格化形式，阶码是补码，尾数是原码。

$x=0.0001101=0.1101*10^{-3}$

又$[-3]_{补}=[-001B]_{补}=[1011]_{补}=1101B$

所以　浮点数形式是

| | | | |
|---|---|---|---|
| 1 | 101 | 0 | 1101000 |

**ASCII 码**（ American Standard Code for Information Interchange ）

美国标准信息交换代码

将每个字符用 7 位的二进制数来表示，共有 128 种状态

大小字母、0…9、其它符号、控制符

‘ 0 ’　——　48

‘ A ’　——　65

‘ a ’　——　97

**汉字信息编码**

1. 汉字输入码

汉字输入方法大体可分为：区位码（数字码）、音码、形码、音形码。

- 区位码：优点是无重码或重码率低，缺点是难于记忆；
- 音码：优点是大多数人都易于掌握，但同音字多，重码率高，影响输入的速度；
- 形码：根据汉字的字型进行编码，编码的规则较多，难于记忆，必须经过训练才能较好地掌握；重码率低；
- 音形码：将音码和形码结合起来，输入汉字，减少重码率，提高汉字输入速度。

2. 汉字交换码

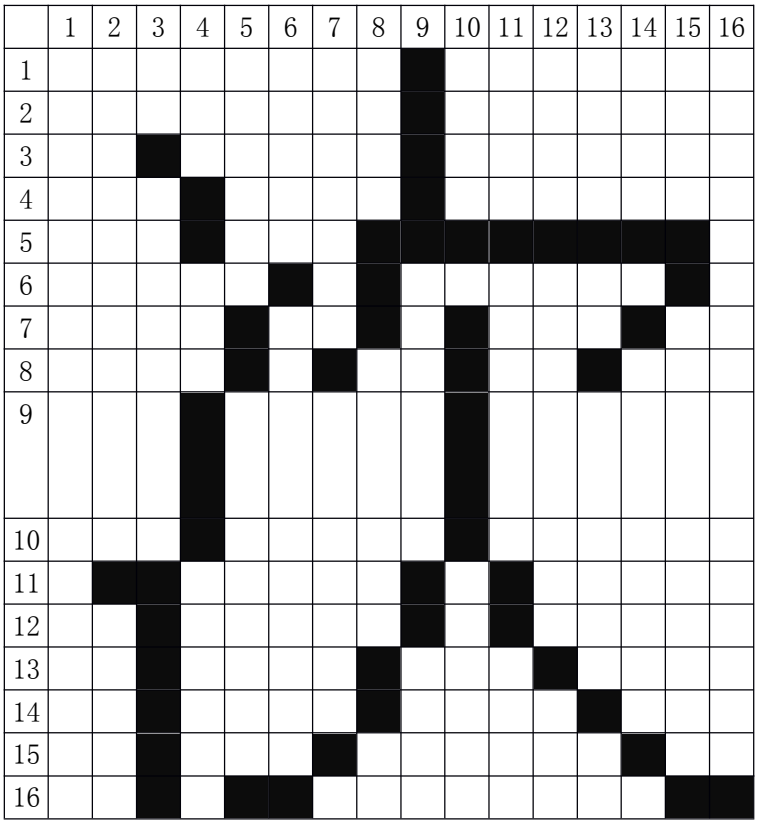
汉字交换码是指不同的具有汉字处理功能的计算机系统之间在交换汉字信息时所使用的代码标准。自国家标准 GB2312－80 公布以来，我国一直延用该标准所规定的国标码作为统一的汉字信息交换码。

GB2312－80 标准包括了 **6763 个汉字**，按其使用频度分为一级汉字 **3755** 个和二级汉字 **3008** 个。**一级汉字按拼音排序**，**二级汉字按部首排序**。此外，该标准还包括标点符号、数种西文字母、图形、数码等符号 682 个。

由于 GB2312－80 是 80 年代制定的标准，在实际应用时常常感到不够，所以，建议处理文字信息的产品采用新颁布的 GB18030 信息交换用汉字编码字符集，这个标准繁、简字均处同一平台，可解决两岸三地间 GB 码与 BIG5 码间的字码转换不便的问题。

3．字形存储码

字形存储码是指供计算机输出汉字（显示或打印）用的二进制信息，也称字模。通常，采用的是数字化点阵字模。如下图：



16×16 点表示

一般的点阵规模有 16×16，24×24，32×32，64×64 等，每一个点在存储器中用一个二进制位（bit）存储。例如，在 16×16 的点阵中，需 16×16bit=32 byte 的存储空间。在相同点阵中，不管其笔划繁简，每个汉字所占的字节数相等。

为了节省存储空间，普遍采用了字形数据压缩技术。所谓的矢量汉字是指用矢量方法将汉字点阵字模进行压缩后得到的汉字字形的数字化信息。

例题

十进制数 11/128 可用二进制数码序列表示为（ D ）。

A)1011/1000000　　B)1011/100000000　　C) 0.001011　　D) 0.0001011

算式$(2047)_{10}-(3FF)_{16}+(2000)_{8}$的结果是（ A ）。

A)$(2048)_{10}$　　B)$(2049)_{10}$　　C) $(3746)_{8}$　　D) $(1AF7)_{16}$

已知 x=$(0.1011010)_{2}$,则[x/2] =（ C ）$_{2}$。

A) 0.1011101.　　B) 11110110　　C) 0.0101101　　D) 0.100110

已知 A=35H,则 A∧05H∨A∧30H 的结果是:( C ) 。

A)30H B)05H C) 35H D) 53H

[x]补码=10011000，其原码为(B )

A)011001111 B)11101000 C)11100110 D)01100101

下列无符号数中，最小的数是（ C ）

A.（11011001）2 B.(75)10 C.(37)8 D.(2A)16

计算机的运算速度取决于给定的时间内，它的处理器所能处理的数据量。处理器一次能处理的数据量叫字长。 已知 64 位的奔腾处理器一次能处理 64 个信息位，相当于（ A ）字节。

A.8 个 B.1 个 C.16 个 D. 2 个

在 24*24 点阵的“字库”中，汉字“一”与“编”的字模占用字节数分别是（C）

A.32，32 B.32，72 C.72，72 D.72，32

计算机中的数有浮点数与定点数两种，其中用浮点数表示的数，通常由（C ）这两部分组成。

A.指数与基数 B. 尾数与小数 C. 阶码与尾数 D.整数与小数

十进制算术表达式：3*512+7*64＋4*8＋5 的运算结果，用二进制表示为（B).

A. 10111100101 B.11111100101

C111110100101 D.11111101101

组成’教授'（jiao shou ）’副教授'（fu jiao shou ）与’讲师'（ jiang shi）这三个词的汉字，在 GB2312－80 字符集中都是一级汉字. 对这三个词排序的结果是（D).

A 教授，副教授，讲师 B. 副教授，教授，讲师

C 讲师，副教授，教授 D. 副教授，讲师，教授

GB2312-80 规定了一级汉字 3755 个，二级汉字 3008 个，其中二级汉字字库中的汉字是以（ B ）为序排列的。

A.以笔划多少 B.以部首 C.以 ASCⅡ码 D.以机内码

十进制数 2004 等值于八进制数（ B ）。

A.3077 B.3724 C.2766 D.4002 E.3755

$(2004)_{10}+(32)_{16}$的结果是（ D ）。

A.$(2036)_{10}$ B.$(2054)_{16}$ C.$(4006)_{10}$ D.$(100000000110)_2$ E.$(2036)_{16}$

十进制数 100.625 等值于二进制数（ B ）。

A.1001100.101 B.1100100.101 C.1100100.011 D.1001100.11 E.1001100.01

以下二进制数的值与十进制数23.456 的值最接近的是（D ）。

A.10111.0101 B.11011.1111 C.11011.0111 D.10111.0111 E.10111.1111