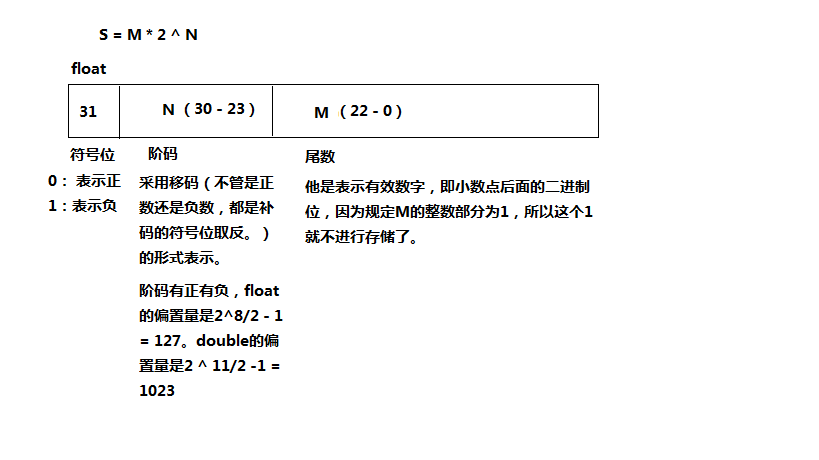
**浮点数在内存中的存储形式**

    a.他是以实型的数据来存储的，即指数的形式来存储的。由于指数形式的小数点可以随便的移动，就像可以浮动似的，所以把这种形式称为浮点数。

    b.这里要说的规范化的指数形式：小数点前一位的数字为0，小数点后第一位的数字不为0。

    c.存储的形式：如下图



23 + 8 + 1 = 32

问题1、为什么要有偏置量，是为了把原来的7位都填充为1，留出最高位出来。在此基础上进行相加或相减，所以为什么float默认小数点的位数是6位。

问题2、尾数的整数部分恒为1，所以整数部分不需要存储，只存小数的部分。

            小数部分求二进制如果不满23位，则补相应的0；

            小数部分求二进制如果是死循环，则算到23位为止。

d.实际的例子：float a = 125.5;

     125 ： 0111 1101               0.5：1

     125.5二进制：111 1101 . 1

     阶数：1.11 1101 1 \* 2 ^ 6，阶数为6。float的偏置量是127。127 + 6 = 133，阶数的二进制为：1000 0101。

     尾数：11 1101 1， 由于规定尾数的整数部分恒为1，要求尾数要达到23位

所以尾数的二进制为：111 1011 0000 0000 0000 0000

    125.5内存中的存储形式为：

    0 10000101 11110110000000000000000，则在内存中存放方式为：

    00000000   低地址

    00000000

    11111011

    01000010   高地址

扩展：小数转换为二进制。

整数和小数分别转换。  
整数除以2，商继续除以2，得到0为止，将余数逆序排列。  
22 / 2 11 余0  
11/2 5 余 1  
5 /2 2 余 1  
2 /2 1 余 0  
1 /2 0 余 1  
所以22的二进制是10110

小数乘以2，取整，小数部分继续乘以2，取整，得到小数部分0为止，将整数顺序排列。  
0.8125x2=1.625 取整1,小数部分是0.625  
0.625x2=1.25 取整1,小数部分是0.25  
0.25x2=0.5 取整0,小数部分是0.5  
0.5x2=1.0 取整1,小数部分是0，结束  
所以0.8125的二进制是0.1101  
十进制22.8125等于二进制10110.1101

0.6 \* 2 = 1.2 ----------- 1

0.2 \* 2 = 0.4 ------------ 0

0.4 \* 2 = 0.8 ------------ 0

0.8 \* 2 = 1.6 ------------ 1

0.6 \* 2 = 1.2 ------------- 1

0.2 \* 2 = 0.4 ------------- 0

0.4 \* 2 = 0.8 -------------- 0

0.8 \* 2 = 1.6 -------------- 1

1 0 0 1 1 0 0 1

1111101.1001 1001 1001 10011

1.111101 1001 1001 1001 100

127 + 6 = 133 1000 0101

1111 0110 0110 0110 0110 0

01000010

11111011

00110011

001100

     反过来计算的话：

    由于符号为为0，则为正数。阶码为133-127=6，尾数为11110110000000000000000，则其真实尾数为1.1111011。所以其大小为1.1111011\*2^6，将小数点右移6位，得到1111101.1，而1111101的十进制为125，0.1的十进制为1\*2^(-1)=0.5，所以其大小为125.5。

    同理若将float型数据0.5转换为二进制形式：

0.5的二进制形式为0.1，由于规定正数部分必须为1，将小数点右移1位，则为1.0\*2^(-1)，其阶码为-1+127=126，表示为01111110，而尾数1.0去掉整数部分为0，补齐0到23位00000000000000000000000，则其二进制表示形式为

0 01111110 00000000000000000000000

由上分析可知float型数据最大表示范围为1.11111111111111111111111\*2^127=3.4\*10^38

对于double型数据情况类似，只不过其阶码为11位，偏置量为1023，尾数为52位。