**C语言中的浮点数存储原则：**

**（1）符合IEEE 754标准，**

**符号位1位，阶码8位，尾数23位 （单精度浮点型）**

**（2）简单变量的内存采用小端字节序存储，即低位保存在低地址，高位保存在高地址**

例如：如变量a取如下值，且&a为0x0019ff28

（1） 0.5

二进制 0.1

（2）0.1

二进制：0.000 1100 1100 1100 （乘出来0.6后循环了）

规格化为1.100110011001100…

阶码为：-4+127 = 123，二进制位0111 1011

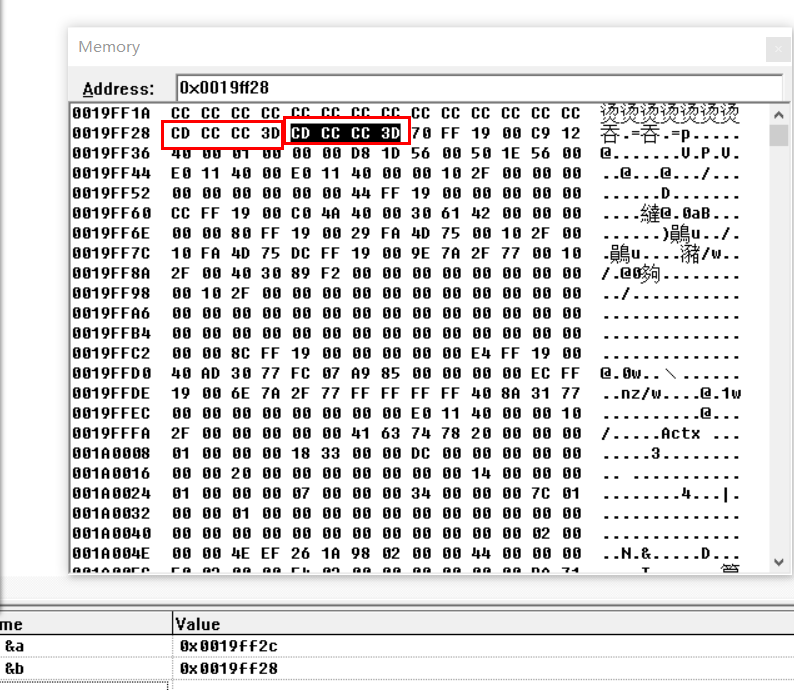
完整表达：**0 0111 1011 1001 1001 1001 100**1 100**1 100**

**3 D C C C C C C**

**内存中,（我的猜测）由于后面舍掉了1100…，有了进位，变成3D CC CC CD**

**地址**0x0019ff28存放的是CD，29单元存放 CC ，2A单元存放CC 2B单元放3D

若先后定义a/b两个变量，由于自动变量放在栈中，a地址大，在后面



（3）0.05（乘以2即为0.1，所以二进制跟0.1跟相似）

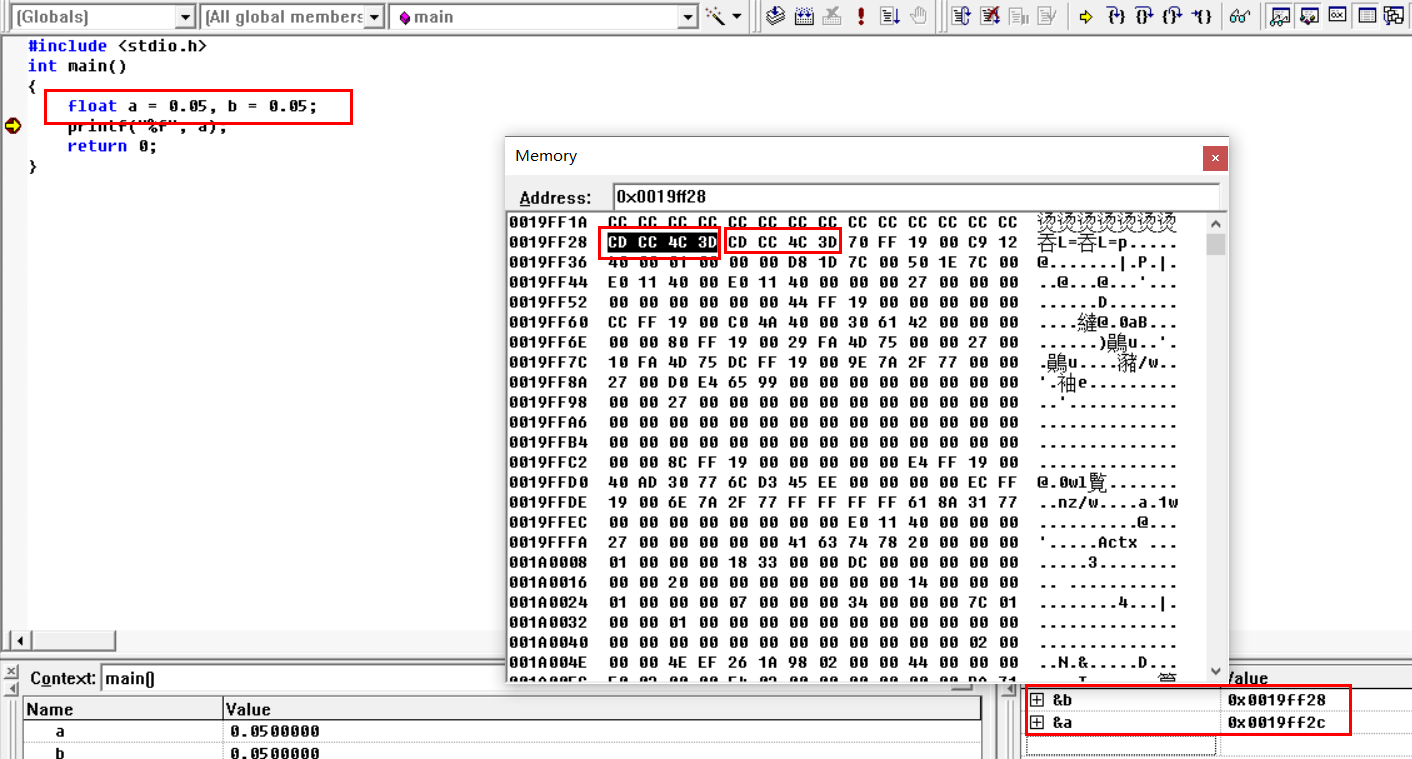
二进制 0.0000 1100 1100 1100…

规格化为：1.100 1100 1100… \* 2的-5次幂，故阶码为122，即0111 1010

完整表达：**0 0111 1010 1001 1001 1001 100**1 100**1 100**

**3 D 4 C C C C C**

**内存中,（我的猜测）由于后面舍掉了1100…，有了进位，变成3D 4C CC CD**



**浮点型在二进制转换和计算（如浮点型乘法）中都会出现舍入问题，且无法预测是舍了，还是入了，导致计算不精确。所以，能用整型的地方，一定首选整型，比如货币、人口问题，数值太大，可以考虑用大的整型数。**