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）

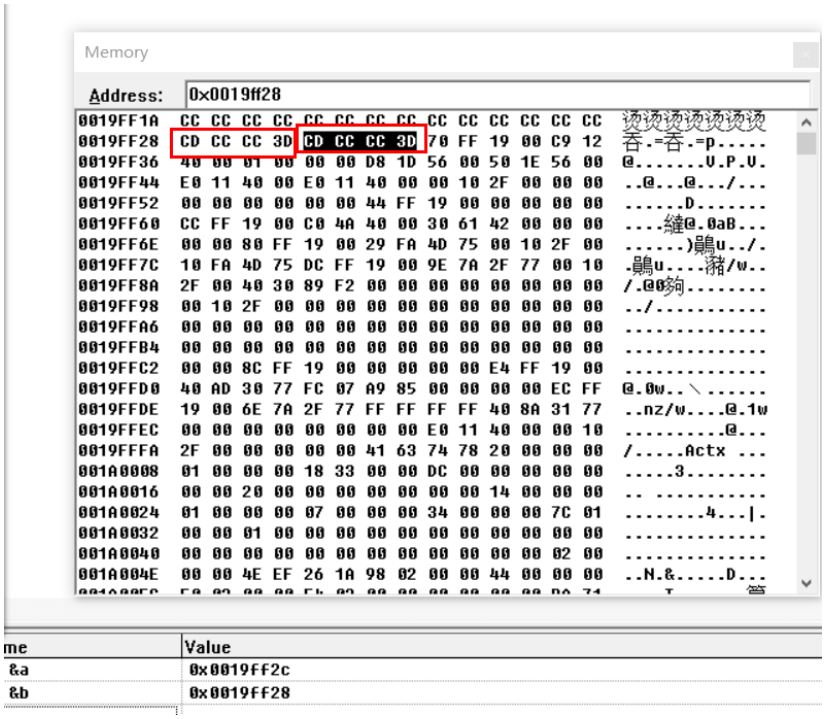
（2）简单变量的内存采用小端字节序存储，即低位保存在低地址，

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

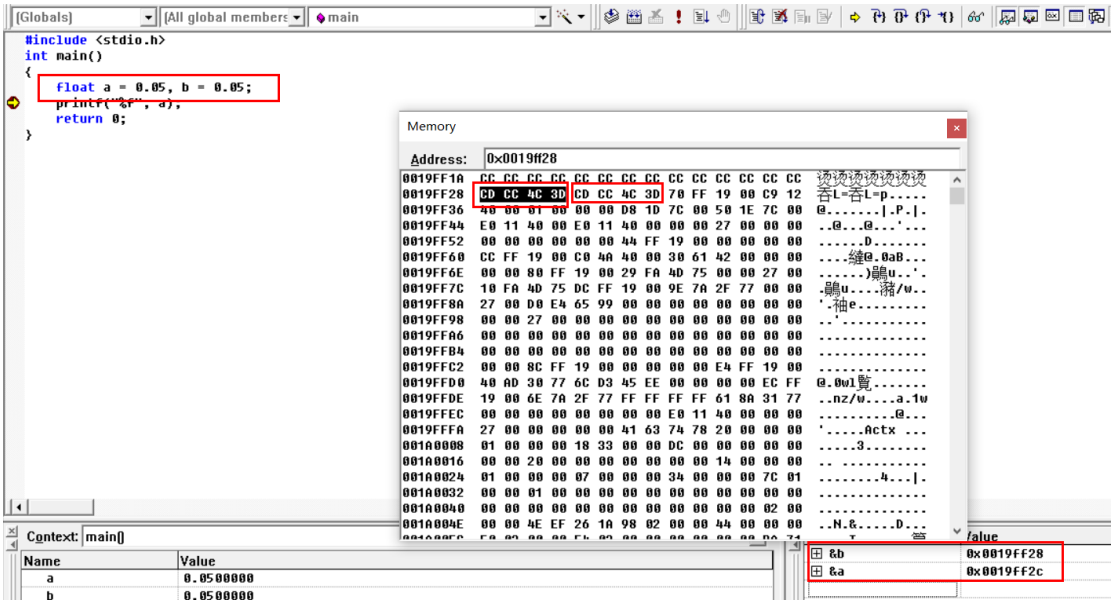
（2）0.1
二进制：0.000 1100 1100 1100 （乘出来 0.6 后循环了）
规格化为 1.100110011001100…
阶码为：-4+127 = 123，二进制位 0111 1011
完整表达：0 0111 1011 1001 1001 1001 1001 1001 100
3 D C C C C C C
内存中，（我的猜测）由于后面舍掉了 1100…，有了进位，变成 3D CC CC CD
地址 0x0019ff28 存放的是 CD，29 单元存放 CC ，2A 单元存放 CC 2B 单元放 3D

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 1001 1001 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。