浮点数的乘除法运算

一.总体思路

1. 从标准输入读入两个浮点数保存在两个 float 当中
2. 通过 union 对其对应的 unsigned long 进行位运算，得到乘除法结果
3. 输出结果，对比计算机原本的运算和我实现的运算的区别

二.算法证明

1. 乘法

$$x \times y = 2^{(E_x + E_y)} \cdot (M_x \times M_y)$$

细节：Mx * My 时需要先将 1 补上，而且乘得的积可能会有进位，为保证规范化可能需要让阶码+1。

2. 除法

$$x \div y = 2^{(E_x - E_y)} \cdot (M_x \div M_y)$$

细节：同样 Mx / My 时需要先将 1 补上，如果不够除，还需要进行对应的移位，同时阶码要做对应的处理。

三.使用方法

源文件 float.c 在文件夹 src 当中，可直接编译运行。

输入两个合法的浮点数，能得到多行结果，分别是程序实现的乘除法得数，c 语言自身实现的乘除法得数，以及他们的 2 进制表示。

若在 linux 系统中，可直接使用 make 命令编译得到可执行程序，再通过命令 cat test.txt | ./true_form.out 来得到实例分析中的测试结果。

四.特殊处理

对于尾数的处理，我采用了直接截断的方式，即没有进位。所以可以发现很多例子当中 c 语言实现，与程序实现的得数 2 进制码尾数相差 1，采用这种方法可适当提高运算速度，省去了对尾数运算后几位的判断，但精度稍差。

五.实例

1.0001 15.789

mul:

15.790577888488769531250000000000000000000

15.790578842163085937500000000000000000000

div:

0.063341557979583740234375000000000000000

0.06334157288074493408203125000000000000

01000001011111001010011000110101
01000001011111001010011000110110

00111101100000011011100100111000
00111101100000011011100100111010

985347.456 125.410154

mul:

123572576.0000000000000000000000000000000000000
123572576.0000000000000000000000000000000000000

div:

7856.998046875000000000000000000000000000
7856.998535156250000000000000000000000000

01001100111010111011001000101100
01001100111010111011001000101100

01000101111101011000011111111100
01000101111101011000011111111101

-123.456 987.654

mul:

-121931.8046875000000000000000000000000000
-121931.8125000000000000000000000000000000

div:

-0.12499924004077911376953125000000000000
-0.12499924004077911376953125000000000000

01000111111011100010010111100111
01000111111011100010010111101000

00111101111111111111111110011010

00111101111111111111111110011010

-15.469 -1.00001

mul:

15.4691543579101562500000000000000000000000

15.4691543579101562500000000000000000000000

div:

15.4688444137573242187500000000000000000000

15.4688453674316406250000000000000000000000

01000001011101111000000110101000

01000001011101111000000110101000

01000001011101111000000001100011

01000001011101111000000001100100

32154789.2 0.1254796

mul:

4034769.7500000000000000000000000000000000000000

4034770.0000000000000000000000000000000000000000

div:

256255120.0000000000000000000000000000000000000000

256255136.0000000000000000000000000000000000000000

01001010011101100100001101000111

01001010011101100100001101001000

01001101011101000110001001001001

01001101011101000110001001001010

0.1254796 32154789.2

mul:

4034769.7500000000000000000000000000000000000000

4034770.0000000000000000000000000000000000000000

div:

0.000000003902360212748590129194781184196５

0.0000000039023606568377999792573973536491

01001010011101100100001101000111

01001010011101100100001101001000

00110001100001100001010110000110

00110001100001100001010110000111