补充：

```
//用于看懂 float/double 内部存储格式的例子
#include <iostream>
using namespace std;
int main()
{
    float f = 123.456f;
    char* p = (char*)&f;
    cout << hex << (int)(*p)     << endl;
    cout << hex << (int)(*(p+1)) << endl;
    cout << hex << (int)(*(p+2)) << endl;
    cout << hex << (int)(*(p+3)) << endl;
    return 0;
}
//注意：x86 系列 CPU 的多字节数据存储，是低位在前
```

2、结合上课内容及上面的示例程序，自行查阅相关资料，并回答一下的问题

（1） float 型数据的 32bit 是如何分段来表示一个单精度的浮点数的？给出 bit 位的分段解释，尾数的正负如何表示？尾数如何表示？指数的正负如何表示？指数如何表示？

三段式
- 尾数 Fraction([0,22] 共 23 位) 范围：0 ~ 2^23-1
- 阶码 Exponent([23,30] 共 8 位) 范围：0 ~ 2^8-1
- 数符 Sign([31,31] 共 1 位) 范围：0 ~ 1

Sign： 0(正) 1(负)
- 即整个浮点数的正负性

尾数真值 = 1 + 尾数(小于 1)
- 尾数可以理解为二进制下小数点后的数

阶码真值 = 阶码 - 127

浮点值 = -1^s * (1+f) * 2^(e-127)

（2） 为什么 float 型数据只有 7 位有效数字？为什么最大只能是 $3.4x10^{38}$？

7 位有效数字（尾数决定）：2^24=16777216，八位，确保精度的话就是七位了。（指数不能算）

最大： 二进制下： 1.11111111111111111111111 << 127

十进制下： 约等于 2^128 约等于 3.4e38

（3） double 型数据的 64bit 是如何分段来表示一个双精度的浮点数的？给出 bit 位的分段解释，尾数的正负如何表示？尾数如何表示？指数的正负如何表示？指数如何表示？

三段式
- 尾数 ([0,51] 共 52 位)
- 阶码 ([52,62] 共 11 位)
- 数符 ([63,63] 共 1 位)

其它类似 32bit，阶码真值就是=阶码 - 1023，即 2^10 -1

浮点值 = -1^s * (1+f) * 2^(e-127)

（4） 为什么 double 型数据有 15 位有效数字？为什么最大是 $1.7x10^{308}$？

15 位有效数字（尾数决定）：2^52=4503599627370496，16 位，去头就是 15 位

最大值：约等于 2^(1024) 约等于 1.7e308

（5） 给出下列 8 个小题（float/double 各自有尾数正负/指数正负）对应变量的 32/64bit 的具体值及解释（写二进制表示时，每 8bit 加 1 个“-”方便查看，例：00100000-01010001）

a) float d=123.456

b) float d=-123.456

c) float d=0.123e-3

d) float d=-1.23e-4

e) double d=123.456

f) double d=-123.456

g) double d=0.123e-3

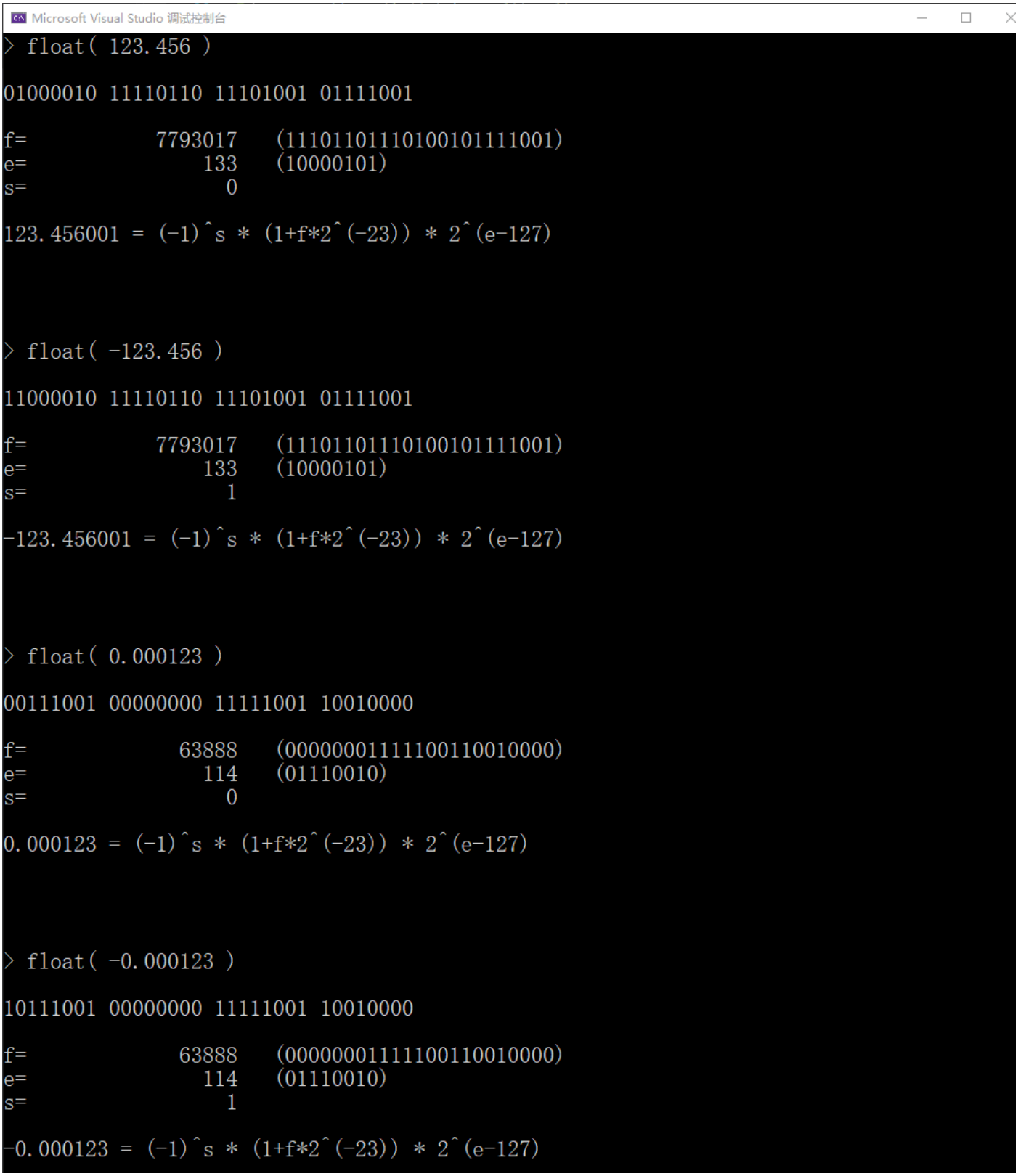
h) double d=-1.23e-4

i)

```
Microsoft Visual Studio 调试控制台
01000010 11110110 11101001 01111001
  133     7793017
2^(133-127)+7793017*2^(133-127-23)=123.4560012317383

11000010 11110110 11101001 01111001
  133     7793017
-(2^(133-127)+7793017*2^(133-127-23))=-123.4560012317383

00111001 00000000 11111001 10010000
  114     63888
2^(114-127)+63888*2^(114-127-23)=0.000123000053

10111001 00000000 11111001 10010000
  114     63888
-(2^(114-127)+63888*2^(114-127-23))=-0.000123000053

01000000 01011110 11011101 00101111 00011010 10011111 10111110 01110111

11000000 01011110 11011101 00101111 00011010 10011111 10111110 01110111

00111111 00100000 00011111 00110001 11110100 01101110 11010010 01000110

10111111 00100000 00011111 00110001 11110100 01101110 11010010 01000110

D:\workspace\2021-spring-sj\Debug\IEEE754_explore.exe (进程 8120)已退出，代码为 0。
按任意键关闭此窗口. . .
```

(更清楚的图在下一面)

测试代码：https://paste.ubuntu.com/p/h9MGbz9P3T/



```
> double( 123.456 )

01000000 01011110 11011101 00101111 00011010 10011111 10111110 01110111

f=  4183844053827191   (1110110111010010111100011010100111111011111001110111)
e=              1029   (10000000101)
s=                 0

123.456000 = (-1)^s * (1+f*2^(-52)) * 2^(e-1023)



> double( -123.456 )

11000000 01011110 11011101 00101111 00011010 10011111 10111110 01110111

f=  4183844053827191   (1110110111010010111100011010100111111011111001110111)
e=              1029   (10000000101)
s=                 1

-123.456000 = (-1)^s * (1+f*2^(-52)) * 2^(e-1023)



> double( 0.000123 )

00111111 00100000 00011111 00110001 11110100 01101110 11010010 01000110

f=    34299414762054   (0000000111110011000111110100011011101101001001000110)
e=              1010   (01111110010)
s=                 0

0.000123 = (-1)^s * (1+f*2^(-52)) * 2^(e-1023)



> double( -0.000123 )

10111111 00100000 00011111 00110001 11110100 01101110 11010010 01000110

f=    34299414762054   (0000000111110011000111110100011011101101001001000110)
e=              1010   (01111110010)
s=                 1

-0.000123 = (-1)^s * (1+f*2^(-52)) * 2^(e-1023)



D:\workspace\2021-spring-sj\Debug\IEEE754_explore.exe (进程 14784)已退出，代码为 0。
按任意键关闭此窗口. . .
```

【文档格式要求：】

1、文档用自己的语言组织

2、如果用到某些小测试程序进行说明，可以贴上小测试程序的源码及运行结果

3、为了使文档更清晰，允许将网上的部分图示资料截图后贴入

4、不允许在答案处直接贴某网址，再附上“见**”（或类似行为），否则实验报告部分直接总分-50

（注：上学期 VS2019 的 Debug 报告，有同学直接将官方文档复制后贴入，后果是优变及格）

【作业要求：】

1、3 月 18 日前网上提交本次作业，直接在本文档上作答，转换为 pdf 后提交即可

2、每题所占平时成绩的具体分值见网页（本题在“实验报告”中提交）

3、超过截止时间提交作业会自动扣除相应的分数，具体见网页上的说明

A 64-bit double format number X is divided as shown in Fig 2. The value v of X is inferred from its constituent fields thus

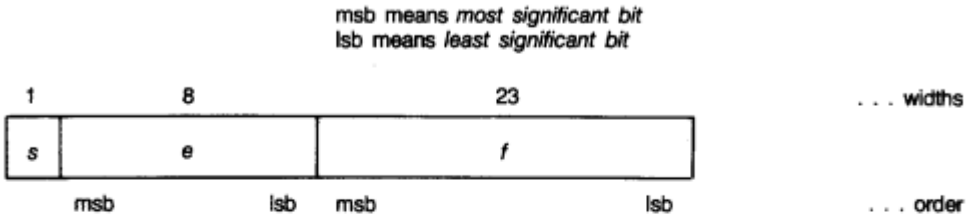


Figure 1—Single Format

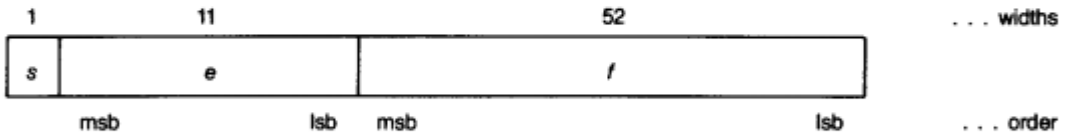


Figure 2—Double Format

1) If $e = 2047$ and $f \neq 0$, then v is NaN regardless of s
2) If $e = 2047$ and $f = 0$, then $v = (-1)^s\infty$
3) If $0 < e < 2047$, then $v = (-1)^s 2^{e-1023}(1 \cdot f)$
4) If $e = 0$ and $f \neq 0$, then $v = (-1)^s 2^{-1022}(0 \cdot f)$ (denormalized numbers)
5) If $e = 0$ and $f = 0$, then $v = (-1)^s 0$ (zero)