§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

要求：

1、完成本文档中所有的题目并写出分析、运行结果

2、无特殊说明，均使用VS2022编译即可

3、直接在本文件上作答，写出答案/截图（不允许手写、手写拍照截图）即可；填写答案时，为适应所填内容或贴图，允许调整页面的字体大小、颜色、文本框的位置等

★ 贴图要有效部分即可，不需要全部内容

★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可

★ 不允许手写在纸上，再拍照贴图

★ 允许在各种软件工具上完成（不含手写），再截图贴图

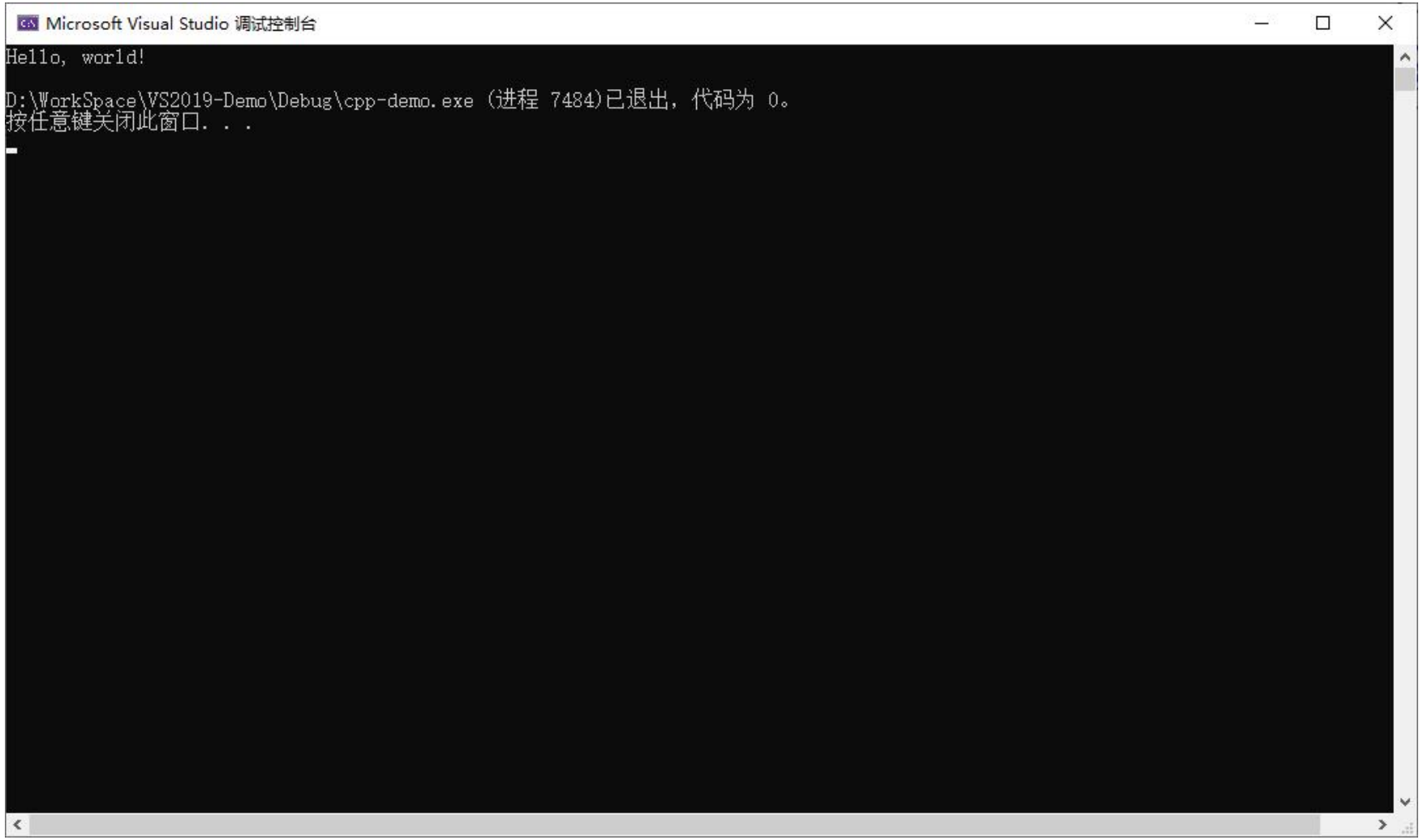
4、转换为pdf后提交

5、3月7日前网上提交本次作业（在“文档作业”中提交）

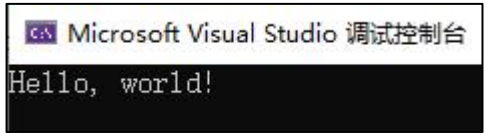
§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

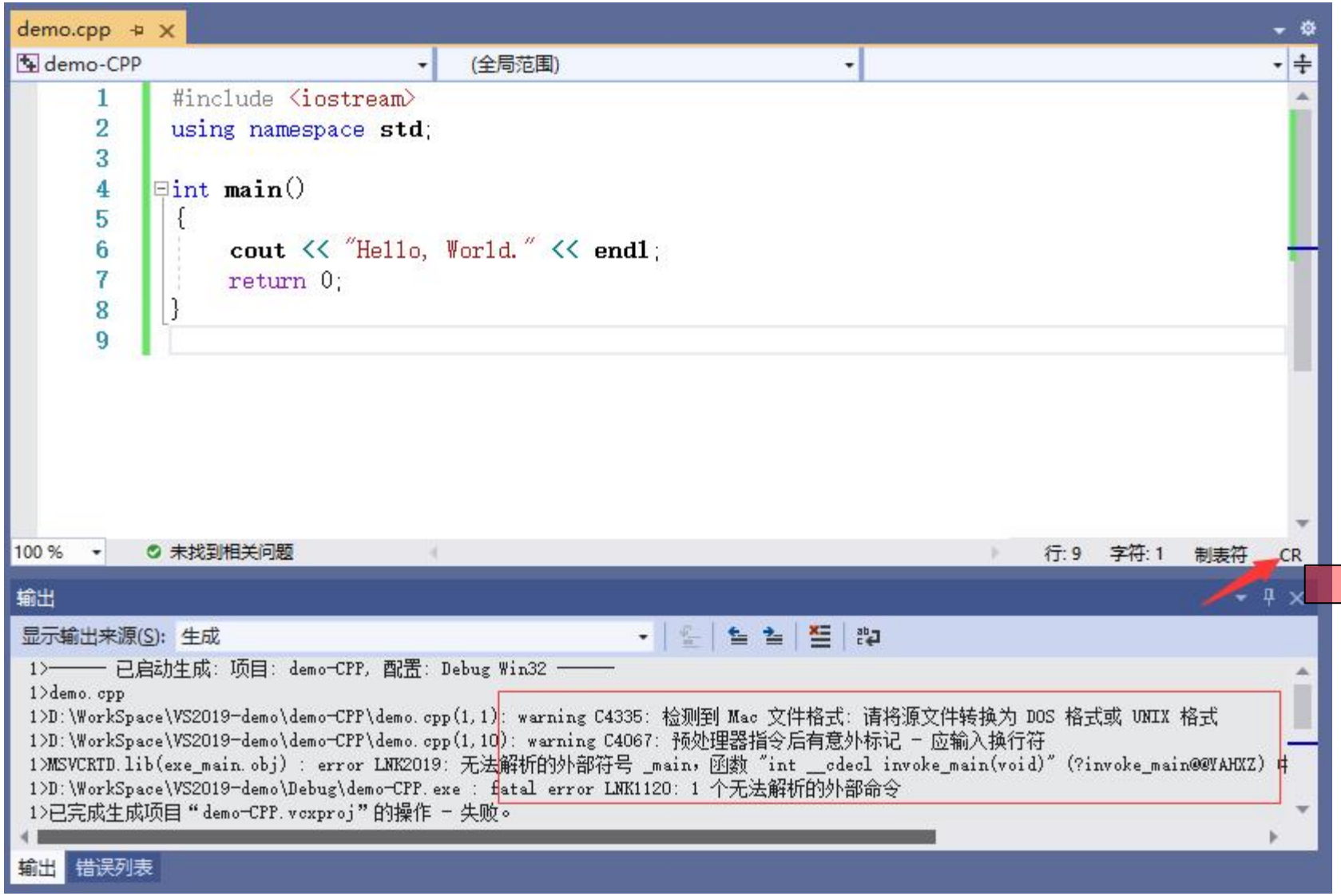


例：有效贴图



§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

附：用WPS等其他第三方软件打开PPT，将代码复制到VS2022中后，如果出现类似下面的编译报错，则观察源程序编辑窗的右下角是否为CR，如果是，单击CR，在弹出中选择CRLF，再次CTRL+F5运行即可



§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

基础知识：用于看懂float型数据的内部存储格式的程序如下：

注意：除了对黄底红字的具体值进行改动外，其余部分不要做改动，也暂时不需要弄懂为什么（需要第6章的知识才能弄懂）

```
#include <iostream>
using namespace std;
int main()
{
    float f = 123.456;
    unsigned char* p = (unsigned 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：忽略本题出现的warning
```

上例解读：单精度浮点数123.456，在内存中占四个字节，四个字节的值依次为0x42 0xf6 0xe9 0x79（按打印顺序逆向取）

转换为32bit则为：0100 0010 | 1111 0110 | 1110 1001 | 0111 1001

符号位　8位指数　23位尾数

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

基础知识：用于看懂double型数据的内部存储格式的程序如下：

注意：除了对黄底红字的具体值进行改动外，其余部分不要做改动，也暂时不需要弄懂为什么（需要第6章的知识才能弄懂）

```
#include <iostream>
using namespace std;
int main()
{
    double d = 1.23e4;
    unsigned char* p = (unsigned char*)&d;
    cout << hex << (int)(*p)     << endl;
    cout << hex << (int)(*(p+1)) << endl;
    cout << hex << (int)(*(p+2)) << endl;
    cout << hex << (int)(*(p+3)) << endl;
    cout << hex << (int)(*(p+4)) << endl;
    cout << hex << (int)(*(p+5)) << endl;
    cout << hex << (int)(*(p+6)) << endl;
    cout << hex << (int)(*(p+7)) << endl;
    return 0;
}
```

上例解读：双精度浮点数1.23e4，在内存中占八个字节，八个字节的值依次为0x40 0xc8 0x06 0x00 0x00 0x00 0x00 0x00(逆向)

转换为64bit则为：0100 0000 | 1100 1000 | 0000 0100 | 0000 0000 | 0000 0000 | 0000 0000 | 0000 0000 | 0000 0000

符号位　11位指数　52位尾数

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

自学内容：自行以"IEEE754"/"浮点数存储格式"/"浮点数存储原理"/"浮点数存储方式"等关键字，
在网上搜索相关文档，读懂并了解浮点数的内部存储机制

学长们推荐的网址：

https://baike.baidu.com/item/IEEE%20754/3869922?fr=aladdin

https://zhuanlan.zhihu.com/p/343033661

https://www.bilibili.com/video/BV1iW411d7hd?is_story_h5=false&p=4&share_from=ugc&share_medium=android&share_plat=android&share_session_id=e12b54be-6ffa-4381-9582-9d5b53c50fb3&share_source=QQ&share_tag=s_i&timestamp=1662273598&unique_k=AuouME0

https://blog.csdn.net/gao_zhennan/article/details/120717424

https://www.h-schmidt.net/FloatConverter/IEEE754.html

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

例：float型数的机内表示

格式要求：多字节时，每8bit中间加一个空格或-(例："11010100 00110001" 或 "11010100-00110001")

例1：100.25

下面是float机内存储手工转十进制的的方法：

(1) 得到的32bit的机内表示是：__0100 0010 1100 1000 1000 0000 0000 0000_ (42 c8 80 00)

(2) 其中：符号位是____0_____

指数是__1000 0101__(填32bit中的原始形式)

指数转换为十进制形式是___133_____(32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是___6_____(32bit中的原始形式按IEEE754的规则转换)

1000 0101

- 0111 1111

= 0000 0110 (0x06 = 6)

尾数是___100 1000 1000 0000 0000 0000____(填32bit中的原始形式)

尾数转换为十进制小数形式是__0.56640625__(32bit中的原始形式按二进制原码形式转换)

尾数表示的十进制小数形式是__1.56640625__(加整数部分的1后)

100 1000 1000 0000 0000 0000 = $2^{0} + 2^{-1} + 2^{-4} + 2^{-8}$

= 0.5 + 0.0625 + 0.00390625 = 0.56640625 => 加1 => 1.56640625

1.56640625 x 2^{6} = 100.25 (此处未体现出误差)

下面是十进制手工转float机内存储的方法：

100 = 0110 0100 (整数部分转二进制为7位，最前面的0只是为了8位对齐，可不要)

0.25 = 01 (小数部分转二进制为2位)

100.25 = 0110 0100.01 = 1.1001 0001 x 2^{6} (确保整数部分为1，移6位)

符 号 位：0

阶 码：6 + 127 = 133 = 1000 0101

尾数(舍1)：1001 0001 => 1001 0001 0000 0000 0000 000 (补齐23位，后面补14个蓝色的0)

100 1000 1000 0000 0000 0000 (从低位开始四位一组，共23位)

注意：

1、作业中绿底/黄底文字/截图可不填

2、计算结果可借助第三方工具完成，没必要完全手算

本页不用作答

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

例：float型数的机内表示

格式要求：多字节时，每8bit中间加一个空格或-(例："11010100 00110001" 或 "11010100-00110001")

例2：1.2

下面是float机内存储手工转十进制的的方法：

(1) 得到的32bit的机内表示是：__0011 1111 1001 1001 1001 1001 1001 1010_ (3f 99 99 9a)

(2) 其中：符号位是____0_____

指数是__0111 1111__(填32bit中的原始形式)

指数转换为十进制形式是___127_____(32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是___0_____(32bit中的原始形式按IEEE754的规则转换)

0111 1111

- 0111 1111

= 0000 0000 (0x0 = 0)

尾数是___001 1001 1001 1001 1001 1010____(填32bit中的原始形式)

尾数转换为十进制小数形式是__0.20000004768371582031 25___(32bit中的原始形式按二进制原码形式转换)

尾数表示的十进制小数形式是__1.20000004768371582031 25___(加整数部分的1后)

001 1001 1001 1001 1001 1010 = $2^{-3} + 2^{-4} + 2^{-7} + 2^{-8} + 2^{-11} + 2^{-12} + 2^{-15} + 2^{-16} + 2^{-19} + 2^{-20} + 2^{-22}$

= 0.125 + ... + 0.00000023841857910156 25(详见右侧蓝色) = 0.20000004768371582031 25

=> 加1 = 1.20000004768371582031 25 (此处已体现出误差)

下面是十进制手工转float机内存储的方法：

1 = 1 (整数部分转二进制为1位)

0.2 = 0011 0011 0011 0011 0011 0011 (小数部分无限循环，转为二进制的24位)

=> 0011 0011 0011 0011 0011 010 (四舍五入为23位，此处体现出误差)

1.2 = 1.0011 0011 0011 0011 0011 010 = 1.0011 0011 0011 0011 0011 010 x 2^0 (确保整数部分为1，移0位)

符 号 位：0

阶 码：0 + 127 = 127 = 0111 1111

尾数(舍1)：0011 0011 0011 0011 0011 010 (共23位)

001 1001 1001 1001 1001 1010 (从低位开始四位一组，共23位)

注意：

1、作业中绿底/黄底文字/截图可不填

2、计算结果可借助第三方工具完成，没必要完全手算

0.125 +
0.0625 +
0.0078125 +
0.00390625 +
0.00048828125 +
0.000244140625 +
0.000030517578125 +
0.0000152587890625 +
0.0000019073486328125 +
0.0000009536743164062 5 +
0.0000002384185791015625

0.20000004768371582031 25

本页不用作答

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

1、float型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001"或"1101-0100-0011-0001")

A.2153393.3933512 （此处设学号是1234567，需换成本人学号，小数为学号逆序，非本人学号0分，下同!!!）
注：尾数为正、指数为正

(1) 得到的32bit的机内表示是：__0100 1010 0000 0011 0110 1110 1100 0110___(不是手算，用P.4方式打印)

(2) 其中：符号位是___0_____

指数是__1001 0100__(填32bit中的原始形式)
指数转换为十进制形式是_____148____(32bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是_____21_____(32bit中的原始形式按IEEE754的规则转换)

尾数是__000 0011 0110 1110 1100 0110__(填32bit中的原始形式)
尾数转换为十进制小数形式是_0.026818037033081054687 5_(32bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是_1.026818037033081054687 5_(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

1、float型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

B.-3933512.2153393 （设学号为1234567，按规则更换为学号和学号逆序）

注：尾数为负、指数为正

(1) 得到的32bit的机内表示是：_1100 1010 0111 0000 0001 0101 0010 0001_(不是手算，用P.4方式打印)

(2) 其中：符号位是_____1_____

指数是___1001 0100___(填32bit中的原始形式)
指数转换为十进制形式是___148_____(32bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是___21_____(32bit中的原始形式按IEEE754的规则转换)

尾数是__111 0000 0001 0101 0010 0001__(填32bit中的原始形式)
尾数转换为十进制小数形式是___0.8756448030471802__(32bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是___1.8756448030471802__(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

1、float型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

C.0.002153393 （设学号为1234567，按规则更换为学号和学号逆序）
注：尾数为正、指数为负

(1) 得到的32bit的机内表示是：__0011 1011 0000 1101 0001 1111 1111 0001__(不是手算，用P.4方式打印)

(2) 其中：符号位是____0_____

指数是___0111 0110___(填32bit中的原始形式)
指数转换为十进制形式是__118__(32bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是__-9___(32bit中的原始形式按IEEE754的规则转换)

尾数是___000 1101 0001 1111 1111 0001___(填32bit中的原始形式)
尾数转换为十进制小数形式是__0.10253727436065674__(32bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是__1.10253727436065674__(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

1、float型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

D.-0.003933512 （设学号为1234567，按规则更换为学号和学号逆序）

注：尾数为负、指数为负

(1) 得到的32bit的机内表示是：__1011 1011 1000 0000 1110 0100 1011 0001__(不是手算，用P.4方式打印)

(2) 其中：符号位是_____1_____

指数是__0111 0111__(填32bit中的原始形式)
指数转换为十进制形式是___119___(32bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是___-8____(32bit中的原始形式按IEEE754的规则转换)

尾数是__000 0000 1110 0100 1011 0001__(填32bit中的原始形式)
尾数转换为十进制小数形式是__0.00697910785675048 8__(32bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是__1.00697910785675048 8__(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

2、double型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

A.2153393.3933512 （设学号为1234567，按规则更换为学号和学号逆序）
注：尾数为正、指数为正

(1) 得到的64bit的机内表示是：___0100 0001 0100 0000 0110 1101 1101 1000 1011 0010 0101 1001 0101 0101 0000 0110___(不是手算，用P.5方式打印)

(2) 其中：符号位是_____0_____

指数是__100 0001 0100__(填64bit中的原始形式)
指数转换为十进制形式是___1044_____(64bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是___21_____(64bit中的原始形式按IEEE754的规则转换)

尾数是__0000 0110 1101 1101 1000 1011 0010 0101 1001 0101 0101 0000 0110__(填64bit中的原始形式)
尾数转换为十进制小数形式是__0.026817986178970354__(64bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是__1.026817986178970354__(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

2、double型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

B.-3933512.2153393 （设学号为1234567，按规则更换为学号和学号逆序）
注：尾数为负、指数为正

(1) 得到的64bit的机内表示是：_1100 0001 0100 1110 0000 0010 1010 0100 0001 1011 1001 0000 0011 1100 1111 1010_(不是手算，用P.5方式打印)

(2) 其中：符号位是____1_____

指数是___100 0001 0100_____(填64bit中的原始形式)
指数转换为十进制形式是____1044_____(64bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是____21_____(64bit中的原始形式按IEEE754的规则转换)

尾数是_1110 0000 0010 1010 0100 0001 1011 1001 0000 0011 1100 1111 1010_(填64bit中的原始形式)
尾数转换为十进制小数形式是_0.8756447865196706_(64bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是_1.8756447865196706_(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

2、double型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

C.0.002153393 （设学号为1234567，按规则更换为学号和学号逆序）
注：尾数为正、指数为负

(1) 得到的64bit的机内表示是：_0011 1111 0110 0001 1010 0011 1111 1110 0001 0000 0101 0101 0111 1100 0111 1000_(不是手算，用P.5方式打印)

(2) 其中：符号位是____0_____

指数是___011 1111 0110_____(填64bit中的原始形式)
指数转换为十进制形式是___1014_____(64bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是___-9_____(64bit中的原始形式按IEEE754的规则转换)

尾数是_0001 1010 0011 1111 1110 0001 0000 0101 0101 0111 1100 0111 1000_(填64bit中的原始形式)
尾数转换为十进制小数形式是_0.10253721599999999_(64bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是_1.10253721599999999_(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

2、double型数的机内表示

格式要求：多字节时，每4bit中间加一个空格或-(例："1101 0100 0011 0001" 或 "1101-0100-0011-0001")

D.-0.003933512 （设学号为1234567，按规则更换为学号和学号逆序）
注：尾数为负、指数为负

(1) 得到的64bit的机内表示是：_1011 1111 0111 0000 0001 1100 1001 0110 0001 0110 0101 1111 1111 0001 0110 0000_(不是手算，用P.5方式打印)

(2) 其中：符号位是____1_____

指数是___011 1111 0111_____(填64bit中的原始形式)
指数转换为十进制形式是___1015_____(64bit中的原始形式按二进制原码形式转换)
指数表示的十进制形式是___-8_____(64bit中的原始形式按IEEE754的规则转换)

尾数是_0000 0001 1100 1001 0110 0001 0110 0101 1111 1111 0001 0110 0000_(填64bit中的原始形式)
尾数转换为十进制小数形式是_0.0069790720000000031_(64bit中的原始形式按二进制原码形式转换)
尾数表示的十进制小数形式是_1.0069790720000000031_(加整数部分的1)

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

3、总结

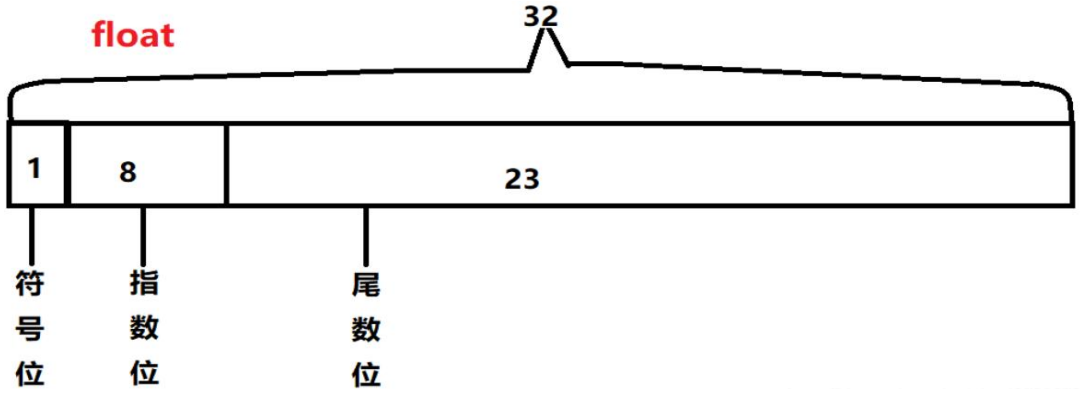
(1) float型数据的32bit是如何分段来表示一个单精度的浮点数的？给出bit位的分段解释

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？

答：符号位（1bit）：0代表正，1代表为负

指数位（8bit）:用于存储科学计数法中的指数数据，并且要加上偏移量（float偏移量127）

尾数部分（23bit）：尾数部分占23个bit，其实它被认为是24个bit，但是第一个bit取值一直都为1，被隐藏掉



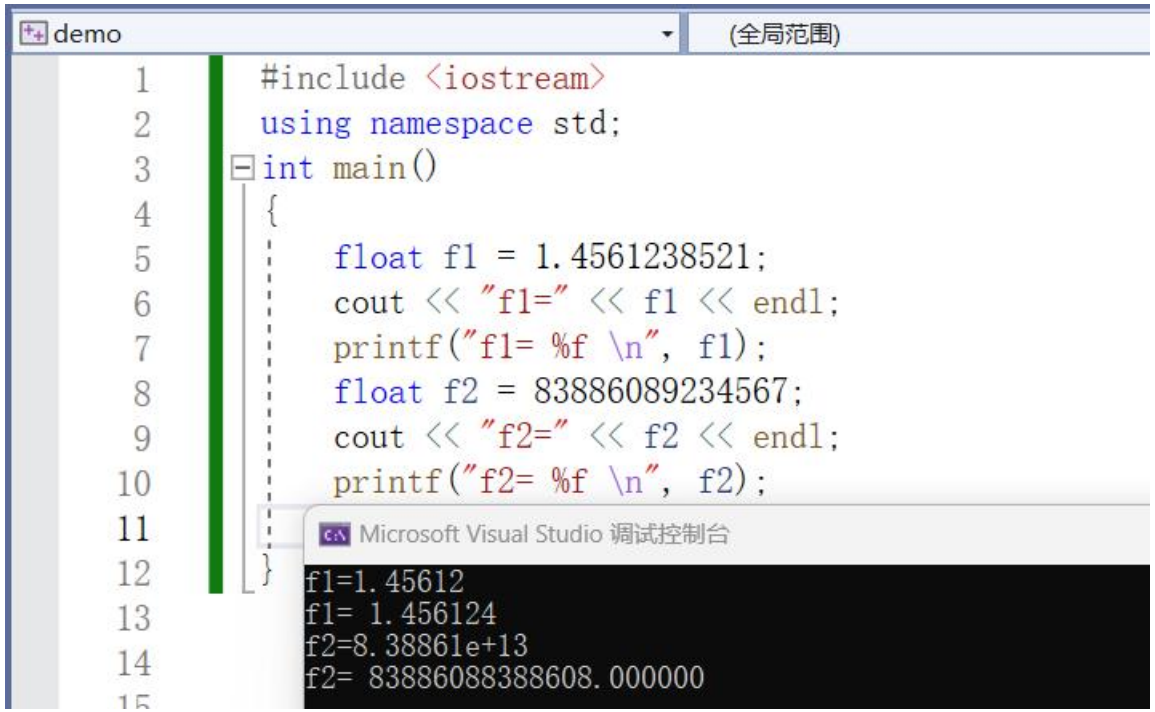
尾数的正负用符号位表示，0代表为正，1代表为负。尾数表示：首先把十进制浮点数表示为二进制数，然后把这个二进制数转换为以2为底的指数形式，转换时,对于乘号左边需要把小数点放在左起第一位和第二位之间，且第一位是非0数。这样表示好以后，得到的小数点后的数字就是尾数，如果不足位数则需要低位补0，补齐23位。

指数的正负我们选用一个偏移量表示，因为指数可正可负，8位的指数位能表示的指数范围就应该为:-127～128，所以指数部分的存储采用移位存储，存储的数据为元数据+127。指数表示：元数据+127后得到的十进制数用除2取余法得到二进制表示后，位数不足在高位补0，补齐8位。

(2) 为什么float型数据只有7位十进制有效数字？为什么最大只能是$3.4x10^{38}$ ？
有些资料上说有效位数是6~7位，能找出6位/7位不同的例子吗？

答：因为单精度数的尾数用23位存储，加上默认的小数点前的1位1，$2^{(23+1)} = 16777216$。因为 $10^7<16777216<10^8$，所以说单精度浮点数的有效位数是7位。最大数字二进制表示为0111 1111 1111 1111 1111 1111 1111 1111尾数0.1111111 11111111 11111111，指数为1111 1111，但是指数全是1时有特殊用途，所以指数位最大时为1111 1110，指数减去127得到127，所以最大的数字就是$1.1111111\ 11111111\ 11111111 * 2^{127}$，这个值为340282346638528859811704183484516925440，通常表示为3.4028235E38，所以最大只能是$3.4x10^{38}$。

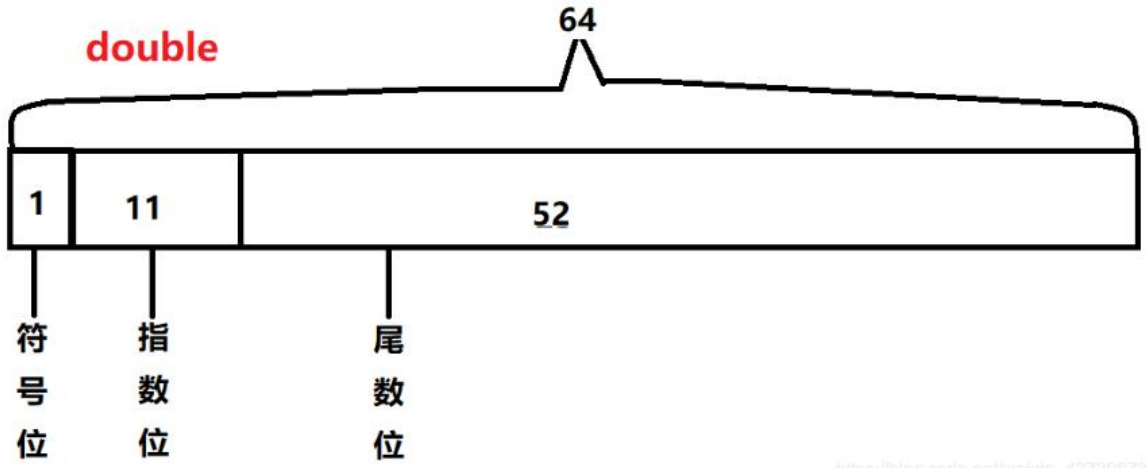
例子如下图所示：



左图我们可以观察到1.4561238521输出为1.456124，精确位数是6位；而83886089234567输出为83886088388608.000000，精确位数为7位

(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？

答：符号位（1bit）：0代表正，1代表为负
指数位（11bit）:用于存储科学计数法中的指数数据，并且要加上偏移量（double偏移量1023）
尾数部分（52bit）：尾数部分占52个bit，其实它被认为是53个bit，但是第一个bit取值一直都为1，被隐藏掉

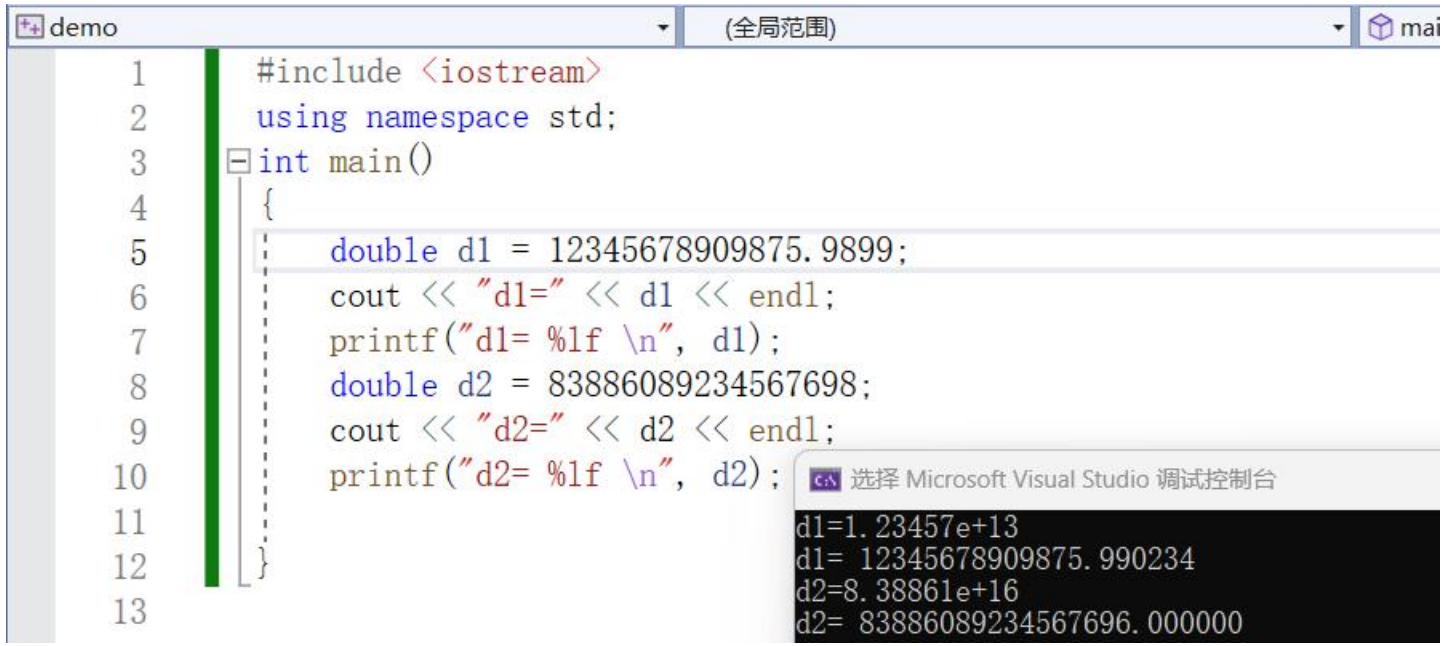


尾数的正负用符号位表示，0代表为正，1代表为负。尾数表示：首先把十进制浮点数表示为二进制数，然后把这个二进制数转换为以2为底的指数形式，转换时,对于乘号左边需要把小数点放在左起第一位和第二位之间，且第一位是非0数。这样表示好以后，得到的小数点后的数字就是尾数，如果不足位数则需要低位补0，补齐52位。
指数的正负我们选用一个偏移量表示，因为指数可正可负，11位的指数位能表示的指数范围就应该为:-1023～1024，所以指数部分的存储采用移位存储，存储的数据为元数据+1023。指数表示：元数据+1023后得到的十进制数用除2取余法得到二进制表示后，位数不足在高位补0，补齐11位。

(4) 为什么double型数据只有15位十进制有效数字？为什么最大只能是1.7×10^{308} ？
有些资料上说有效位数是15~16位，能找出15位/16位不同的例子吗？

答：双精度浮点数总共用64位来表示浮点数，其中尾数用52位存储，2^{53}有15个十进制位，所以double型数据只有15位十进制有效数字。double的指数范围是-1023～+1024，因而最大值是2^{1024}，约为1.79E+308。

例子如下图所示：



从左图我们看到12345678909875.9899输出为12345678909875.990234，有效位数为15位；
83886089234567698输出为
83886089234567696.000000，有效位数为16位

- 文档用自己的语言组织
- 篇幅不够允许加页
- 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
- 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
- 不允许在答案处直接贴某网址，再附上“见**”（或类似行为），否则文档作业部分直接总分-50

§.基础知识题 - 浮点数机内存储格式(IEEE 754)理解

4、思考

(1) 8/11bit的指数的表示形式是2进制补码吗？如果不是，一般称为什么方式表示?

答：8/11bit的指数的表示形式不是2进制补码，一般称为移码表示

(2) double赋值给float时，下面两个程序，double型常量不加F的情况下，左侧有warning，右侧无warning，为什么？总结一下规律

```
#include <iostream>
using namespace std;
int main()
{
    float f = 1.2;
    unsigned char* p = (unsigned 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;
}
```

warning C4305: “初始化”: 从“double”到“float”截断

```
#include <iostream>
using namespace std;
int main()
{
    float f = 100.25;
    unsigned char* p = (unsigned 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;
}
```

答：原因：1.2 单精度表示：0 01111111 00110011001100110011010

双精度表示：0 01111111111 0011001100110011001100110011001100110011001100110011

100.25 单精度表示：0 10000101 10010001000000000000000

双精度表示：0 10000000101 1001000100000000000000000000000000000000000000000000

由上我们可以观察：将十进制的小数转换为二进制的小数的方法为将小数*2，取整数部分，所以0.2*2=0.4，所以二进制小数第一位为0，0.4×2=0.8，第二位为0,0.8*2=1.6,第三位为1，0.6×2=1.2，第四位为1，0.2*2=0.4，第五位为0，这样一直乘下去也不会得到1.0，得到的二进制是一个无限循环的排列00110011001100110011...，对于单精度数据来说（见后页）

尾数只能表示24bit的精度，所以1.2的float存储为:0 01111111 00110011001100110011010 但是在换算成十进制的值的时候，却不恰好为1.2，因为十进制在转换为二进制的时候可能会不准确，而double型的数据也存在同样的问题，所以在表示浮点数时候会存在一定的误差，在double转float，过程中，也可能会存在相应误差；对于能够用二进制精确表示的十进制数据，如 100.25，这个误差就会不存在。

规律：能写成2的n次幂（n为负数）相加的小数（不超过float的范围的小数）如：用0.5，0.25，0.125...这些数字和可以表示的小数，在double赋值给float时候不会存在误差，其余情况均会出现误差。