

§ . 基础知识题 – 浮点数机内存存储格式(IEEE 754)理解



要求:

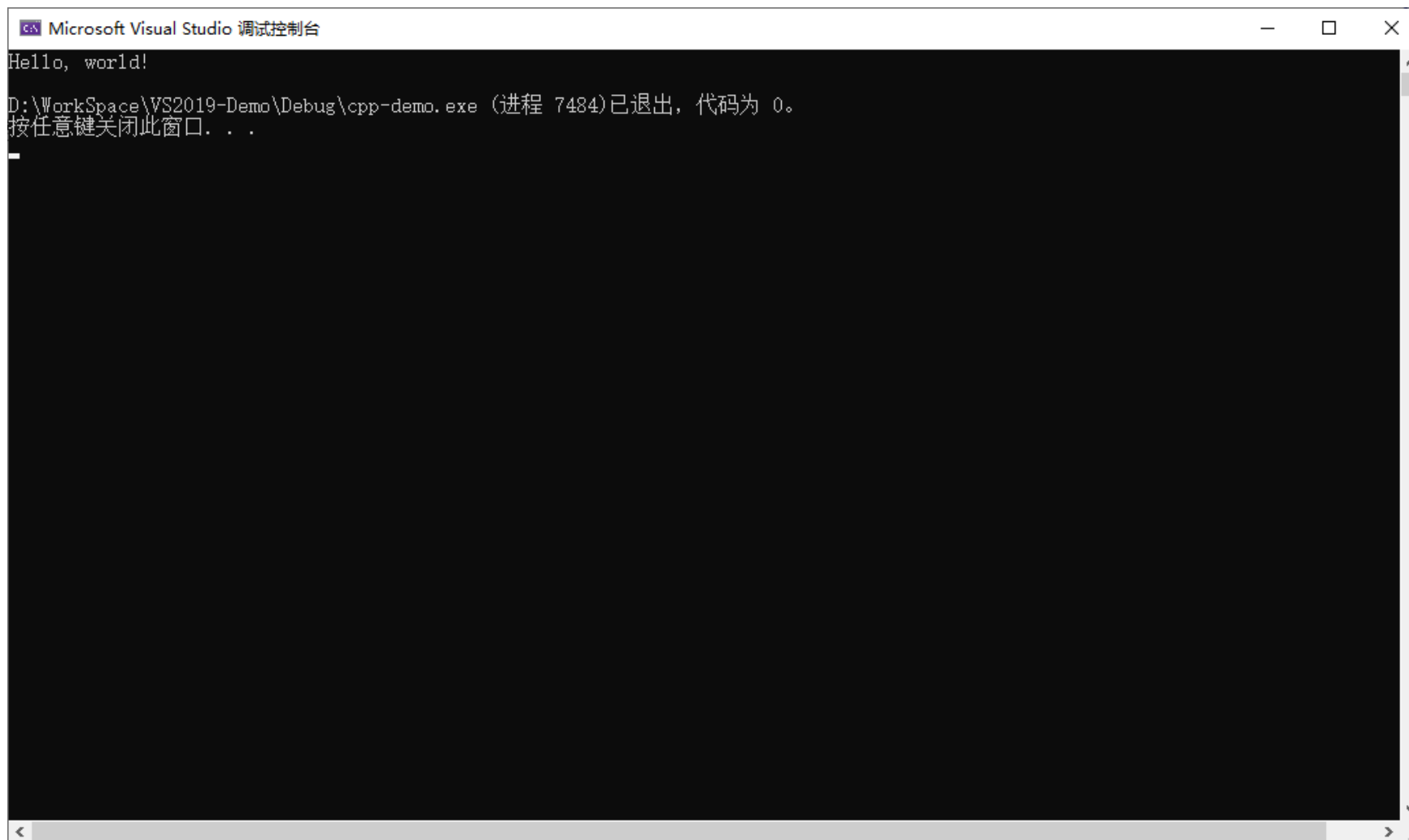
- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
- 4、转换为pdf后提交
- 5、**3月14日前**网上提交本次作业（在“文档作业”中提交）

§. 基础知识题 – 浮点数机内存储格式(IEEE 754)理解

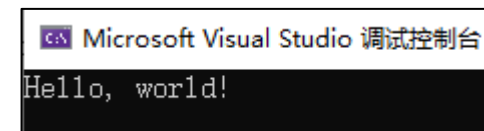


贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

A screenshot of the Microsoft Visual Studio debug console window. The window title is "Microsoft Visual Studio 调试控制台". The output text is: "Hello, world!", "D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0.", and "按任意键关闭此窗口. . .". The entire window, including its title bar and scrollbars, is captured, which is considered an invalid screenshot according to the requirements.

例：有效贴图

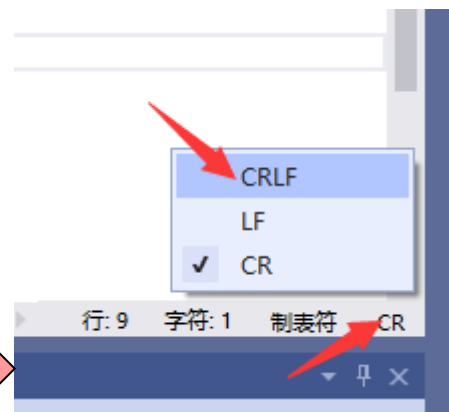
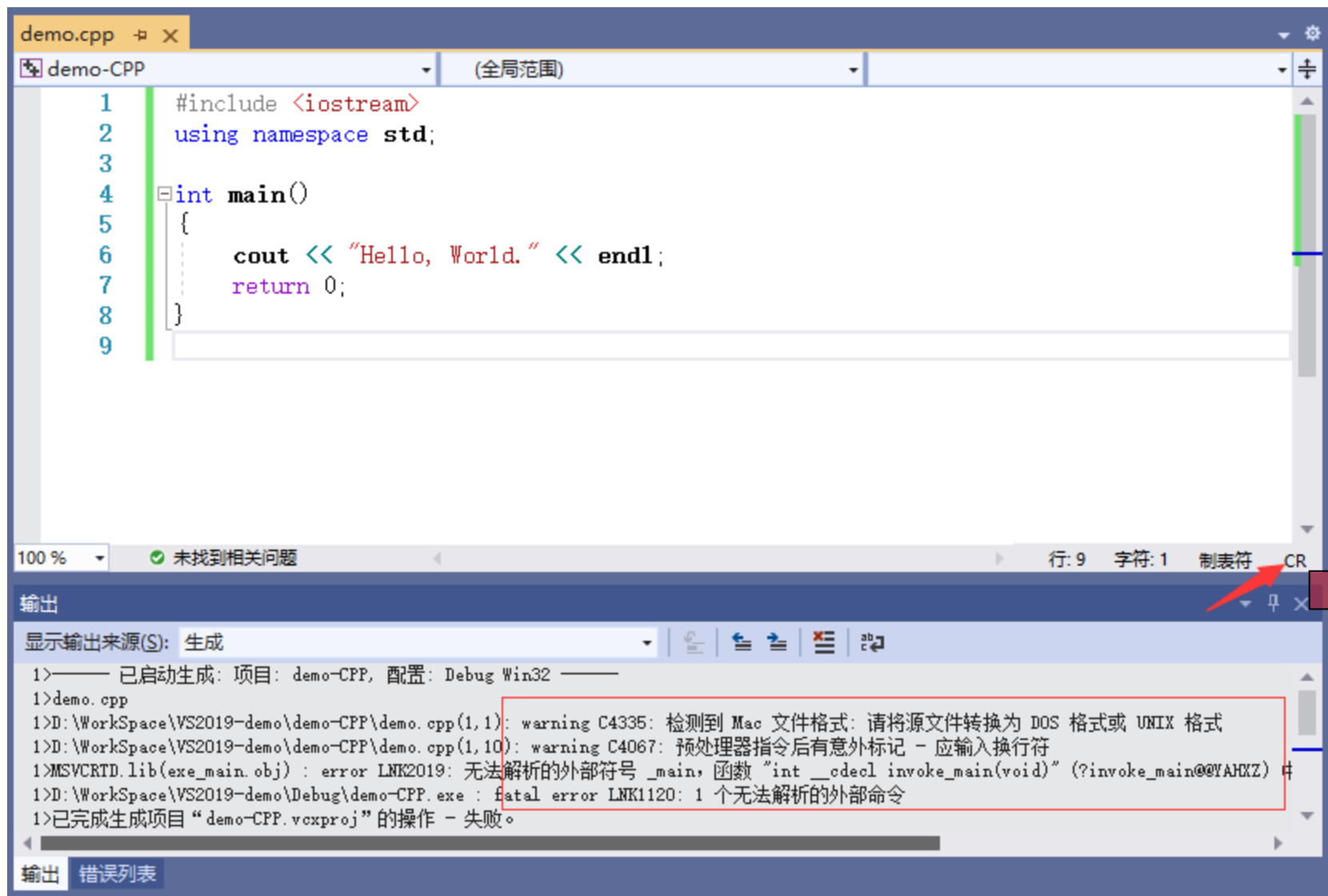
A screenshot of the Microsoft Visual Studio debug console window. The window title is "Microsoft Visual Studio 调试控制台". The output text is: "Hello, world!". Only the text output is captured, which is considered a valid screenshot according to the requirements.



§. 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

Microsoft
79
e9
f6
42

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;
}
```

Microsoft
0
0
0
0
0
6
c8
40

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/BV1iW41ld7hd?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×tamp=1662273598&unique_k=AuouMEO

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方法:

(1) 得到的32bit的机内表示是: 0011 1111 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 1010 (填32bit中的原始形式)

尾数转换为十进制小数形式是 0.2000000476837158203125 (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 1.2000000476837158203125 (加整数部分的1后)

001 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 + ... + 0.0000002384185791015625 (详见右侧蓝色) = 0.2000000476837158203125

=> 加1 = 1.2000000476837158203125 (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 +
0.00000095367431640625 +
0.0000002384185791015625

0.2000000476837158203125

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. 2352018. 8102532 (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 1111 1000 1110 0100 1011 (4A 0F 8E 4B)

(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)

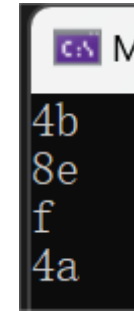
指数表示的十进制形式是____21____ (32bit中的原始形式按IEEE754的规则转换)

尾数是 000 1111 1000 1110 0100 1011 (填32bit中的原始形式)

尾数转换为十进制小数形式是 0.12152993679046630859375 (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 1.12152993679046630859375 (加整数部分的1)

注：转换为十进制小数用附加的工具去做，自己去网上找工具也行，但要满足精度要求 (下同!!!)





§. 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. -8102532. 2352018 (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 1111 0111 0100 0101 0000 1000 (CA F7 45 08)

(2) 其中：符号位是1

指数是1001 0101 (填32bit中的原始形式)

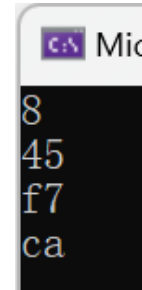
指数转换为十进制形式是149 (32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是22 (32bit中的原始形式按IEEE754的规则转换)

尾数是 111 0111 0100 0101 0000 1000 (填32bit中的原始形式)

尾数转换为十进制小数形式是 -0.93179416656494140625 (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 -1.93179416656494140625 (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. 0.002352018 (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 0001 1010 0010 0100 0101 0000 (3B 1A 24 50)

(2) 其中：符号位是____0____

指数是__0111 0110__ (填32bit中的原始形式)

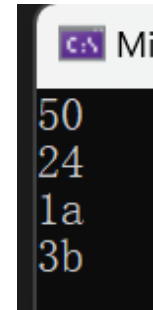
指数转换为十进制形式是__118__ (32bit中的原始形式按二进制原码形式转换)

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 001 1010 0010 0100 0101 0000 (填32bit中的原始形式)

尾数转换为十进制小数形式是0.2042331695556640625 (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1.2042331695556640625 (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. -0.008102532 (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 1100 0000 0100 1100 0000 0111 1011 (BC 04 C0 7B)

(2) 其中：符号位是____1____

指数是__0111 1000__ (填32bit中的原始形式)

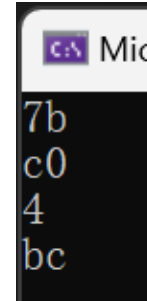
指数转换为十进制形式是____120____ (32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是____-7____ (32bit中的原始形式按IEEE754的规则转换)

尾数是000 0100 1100 0000 0111 1011 (填32bit中的原始形式)

尾数转换为十进制小数形式是-0.03712403774261474609375 (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-1.03712403774261474609375 (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. 2352018.8102532 (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 0001 1111 0001 1100 1001 0110 0111 1011 0110 0110 0000 0111 1010
(41 41 F1 C9 67 B6 60 7A)

(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 0001 1111 0001 1100 1001 0110 0111 1011 0110 0110 0000 0111 1010 (填64bit中的原始形式)

尾数转换为十进制小数形式是 0.121529965521431027042353889555670320987701416015625 (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 1.121529965521431027042353889555670320987701416015625 (加整数部分的1)

Micro

7a
60
b6
67
c9
f1
41
41



§. 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. -8102532. 2352018 (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 0101 1110 1110 1000 1010 0001 0000 1111 0000 1101 1000 1011 1101 1010
(C1 5E E8 A1 0F 0D 8B DA)

(2) 其中：符号位是____1____

指数是__100 0001 0101__ (填64bit中的原始形式)

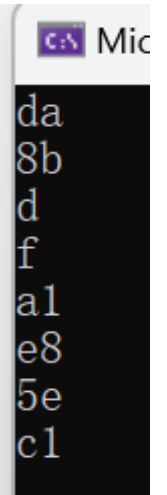
指数转换为十进制形式是__1045__ (64bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是__22__ (64bit中的原始形式按IEEE754的规则转换)

尾数是 1110 1110 1000 1010 0001 0000 1111 0000 1101 1000 1011 1101 1010 (填64bit中的原始形式)

尾数转换为十进制小数形式是 -0.931794222641420422093005981878377497196197509765625 (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 -1.931794222641420422093005981878377497196197509765625 (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. 0.002352018 (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 0011 0100 0100 1000 1010 0000 1100 0111 0111 1010 0001 0110 0110
(3F 63 44 8A 0C 77 A1 66)

(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 0011 0100 0100 1000 1010 0000 1100 0111 0111 1010 0001 0110 0110 (填64bit中的原始形式)

尾数转换为十进制小数形式是 0.20423321599999995204916558577679097652435302734375 (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 1.20423321599999995204916558577679097652435302734375 (加整数部分的1)

Micro

66
a1
77
c
8a
44
63
3f



§. 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. -0.008102532 (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 1000 0000 1001 1000 0000 1111 0110 1111 1010 0011 0110 1010 1100 0011
(BF 80 98 0F 6F A3 6A C3)

(2) 其中：符号位是____1____

指数是011 1111 1000 (填64bit中的原始形式)

指数转换为十进制形式是1016 (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-7 (64bit中的原始形式按IEEE754的规则转换)

尾数是0000 1001 1000 0000 1111 0110 1111 1010 0011 0110 1010 1100 0011 (填64bit中的原始形式)

尾数转换为十进制小数形式是-0.0371240960000001063434638126636855304241180419921875 (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-1.0371240960000001063434638126636855304241180419921875 (加整数部分的1)

CA Micro

c3
6a
a3
6f
f
98
80
bf



§ . 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？

①float型数据存储时分三个部分

1位为符号位，8位为指数位，23位为尾数位

②尾数的正负通过符号位来表示，0表示正，1表示负

③尾数部分占用23位，是一个二进制小数，

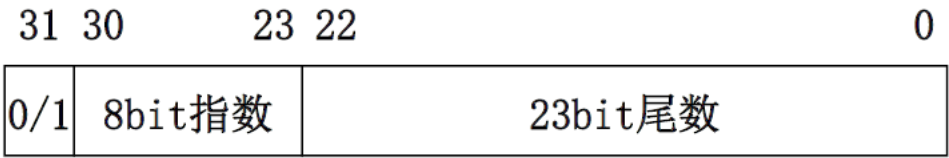
在表示时，首位默认是1，这个1不用显式表达，不占内存

23bit表示二进制小数中小数点后的部分，不足的补0

④指数值是通过对该8位的二进制数进行解码得到

在float型中指数的偏移值为127，指数部分的实际值等于编码值减去偏移值127

指数的正负由阶码对应的十进制数和127的大小关系决定，大于127为正，小于127为负





(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 3.4×10^{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？

①float型数据有23位的尾数，换算成十进制时候， $2^{23} = 8388608 \approx 10^7$ ，因此一般而言有效数位为7位

②float型数据有8位的指数，阶码取值范围从00000000到11111111最大补码，对应阶码取值-127-128，最多乘以 2^{128} ，
 $2^{128} \approx 3.4 \times 10^{38}$

所以最大只能是 3.4×10^{38}

③第6/7位不同的例子如下

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4  int main()
5  {
6      float n = 12.56780f;
7      cout << setprecision(10) << n << endl;
8      return 0;
9  }
```

Microsoft Visual Studio 调试控制台

12.56779957



(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？

①float型数据存储时分三个部分

1位为符号位，11位为指数位，52位为尾数位

②尾数的正负通过符号位来表示，0表示正，1表示负

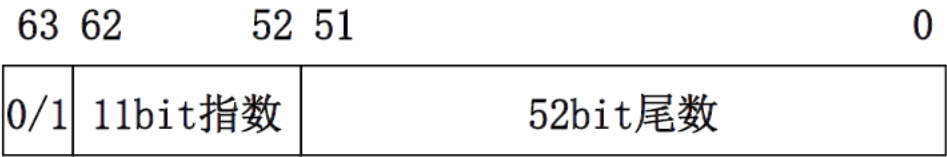
③尾数部分占用52位，是一个二进制小数，

在表示时，首位默认是1，这个1不用显式表达，不占内存
52bit表示二进制小数中小数点后的部分，不足的补0

④指数值是通过对该11位的二进制数进行解码得到

在double型中指数的偏移值为1023，指数部分的实际值等于编码值减去偏移值1023

指数的正负由阶码对应的十进制数和1023的大小关系决定，大于1023为正，小于1023为负





(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？

- ①float型数据有52位的尾数，换算成十进制时候， $2^{52}=4503599627370469$ 是 10^{15} 数量级，因此一般而言有效数位为15位
- ②float型数据有11位的指数，阶码最大补码11111111111，对应阶码取值1024，最多乘以 2^{2024} ， $2^{2024} \approx 1.7 \times 10^{308}$
所以最大只能是 1.7×10^{308}
- ③ $10^{15} < 2^{52} = 4503599627370469 < 10^{16}$ ，所以有些资料上说有效位数是15~16位
- ④第15/16位不同的例子如下

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4  int main()
5  {
6      double n = 1.111111123520189;
7      cout << setprecision(15) << n << endl;
8      return 0;
9  }
```

Microsoft Visual Studio 调试控制台

1.11111112352019

- 注：
- 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？

不是，float型(8 bit)的指数表示的结果为其指数的原码加上127的原码，double型(11 bit)的指数表示的结果为其指数的原码加上1023的原码。

(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;
}
```

①100.25为float型数据, 转化为2进制数后再进行转化时会产生高位截断，但截断数位均为0，不会产生误差，而1.2转化为二进制进行高位截断时，被截断的数位中有1，导致数据转换不完整，所以左侧产生warning而右侧没有。

②规律：double 所占字节多，精度高，而float 所占字节少，精度低，所以float 赋值给double 不会发生截断，double 赋值给float 可能发生截断，并且要根据数的具体值判断截断是否产生误差。