

§ . 基础知识题 – 浮点数机内存存储格式(IEEE 754)理解



要求:

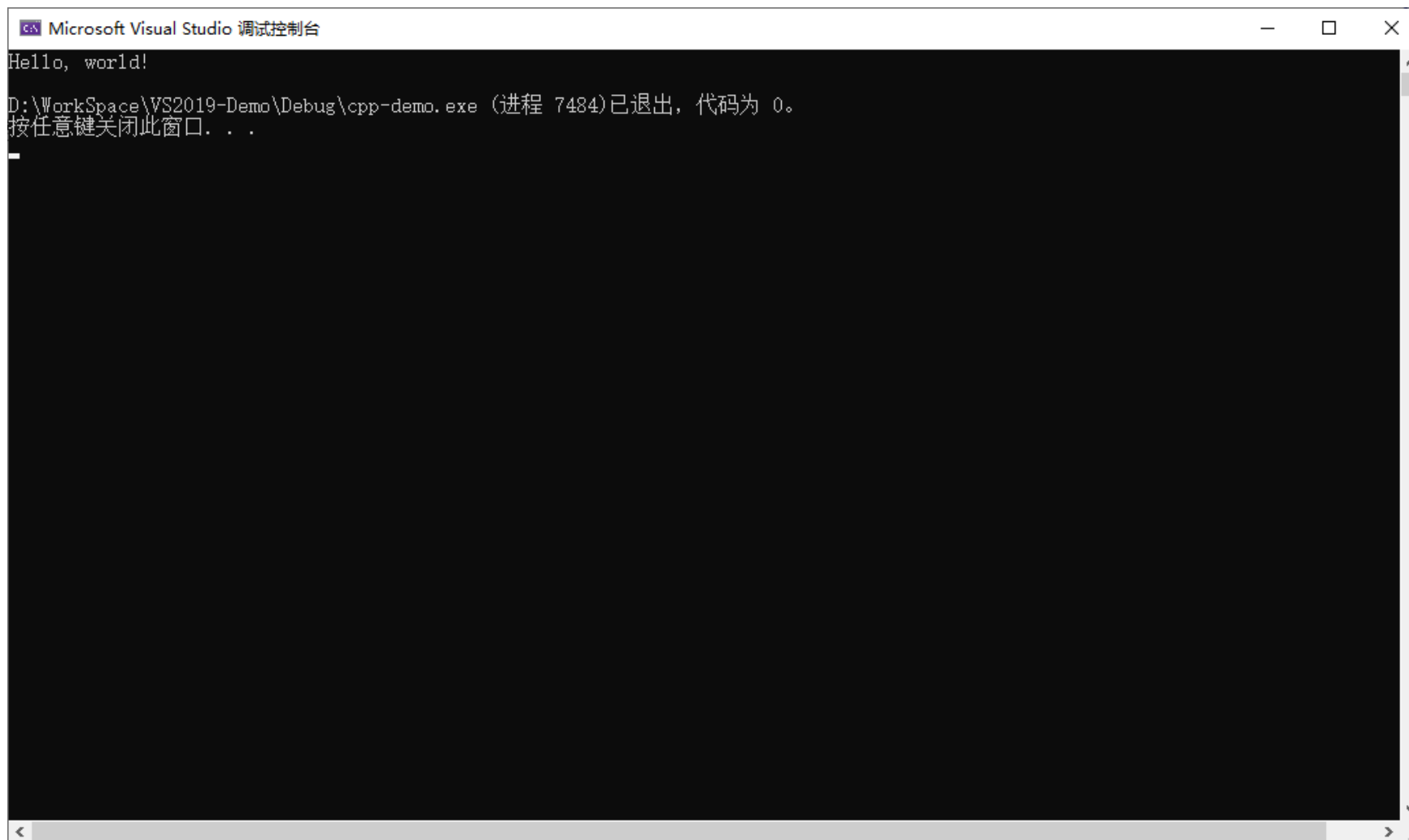
- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
- 4、转换为pdf后提交
- 5、**3月14日前**网上提交本次作业（在“文档作业”中提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

§. 基础知识题 – 浮点数机内存储格式(IEEE 754)理解

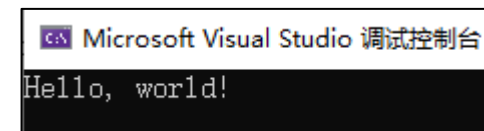


贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

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 screenshot is a full window capture, including the title bar and window controls, which is considered an invalid example according to the requirements.

例：有效贴图

A screenshot of the Microsoft Visual Studio debug console window, showing only the output text: "Hello, world!". This is a cropped version of the previous screenshot, focusing only on the effective output, which is considered a valid example.



§. 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

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      cout << "Hello, World." << endl;
7      return 0;
8  }
9
```

100 % 未找到相关问题 行: 9 字符: 1 制表符 CR

输出

显示输出来源(S): 生成

1>—— 已启动生成: 项目: demo-CPP, 配置: Debug Win32 ——

1>demo.cpp

1>D:\WorkSpace\VS2019-demo\demo-CPP\demo.cpp(1,1): warning C4335: 检测到 Mac 文件格式: 请将源文件转换为 DOS 格式或 UNIX 格式

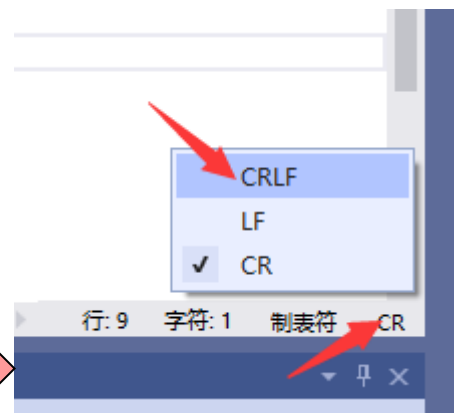
1>D:\WorkSpace\VS2019-demo\demo-CPP\demo.cpp(1,10): warning C4067: 预处理器指令后有意外标记 - 应输入换行符

1>MSVCRTD.lib(exe_main.obj) : error LNK2019: 无法解析的外部符号 _main, 函数 "int __cdecl invoke_main(void)" (?invoke_main@YAHXZ) 中

1>D:\WorkSpace\VS2019-demo\Debug\demo-CPP.exe : fatal error LNK1120: 1 个无法解析的外部命令

1>已完成生成项目 "demo-CPP.vcxproj" 的操作 - 失败。

输出 错误列表



§. 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×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. 2351050.0501532 (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 0111 1111 0010 1000 (4a 0f 7f 28) (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 1111 0111 1111 0010 1000 (填32bit中的原始形式)

尾数转换为十进制小数形式是0.12106800079345703125 (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1.12106800079345703125 (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. -0501532.2351050 (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 1000 1111 0100 1110 0011 1000 1000 (c8 f4 e3 88) (不是手算，用P.4方式打印)

(2) 其中：符号位是_____1_____

指数是_____1001 0001_____ (填32bit中的原始形式)

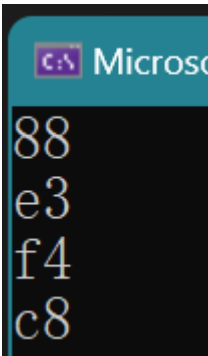
指数转换为十进制形式是_____145_____ (32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是_____18_____ (32bit中的原始形式按IEEE754的规则转换)

尾数是_____111 0100 1110 0011 1000 1000_____ (填32bit中的原始形式)

尾数转换为十进制小数形式是_-0.91319370269775390625_____ (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_-1.91319370269775390625_____ (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. 0.002351050 (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 0001 0100 0001 0011 (3b 1a 14 13) (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)

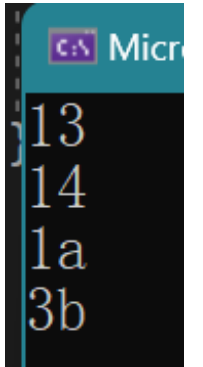
指数转换为十进制形式是____118____ (32bit中的原始形式按二进制原码形式转换)

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____001 1010 0001 0100 0001 0011____ (填32bit中的原始形式)

尾数转换为十进制小数形式是____0.20373761653900146484375____ (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____1.20373761653900146484375____ (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. -0.000501532 (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 1010 0000 0011 0111 1001 0011 1110 (ba 03 79 3e) (不是手算，用P.4方式打印)

(2) 其中：符号位是 1

指数是 0111 0100 (填32bit中的原始形式)

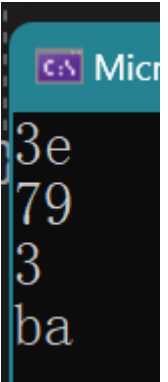
指数转换为十进制形式是 116 (32bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是 -11 (32bit中的原始形式按IEEE754的规则转换)

尾数是 000 0011 0111 1001 0011 1110 (填32bit中的原始形式)

尾数转换为十进制小数形式是 -0.0271375179290771484375 (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 -1.0271375179290771484375 (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. 2351050.0501532 (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 1110 1111 1110 0101 0000 0110 0110 1011 0110 1011 1000 1001 (41 41 ef e5 06 6b 6b 89) (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)

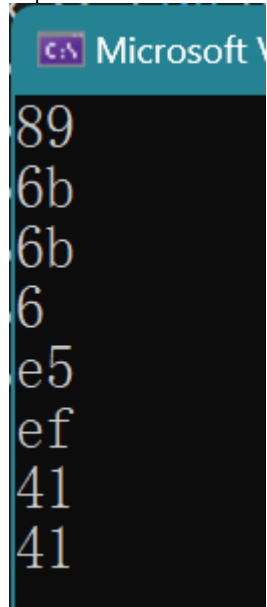
指数转换为十进制形式是____1044____ (64bit中的原始形式按二进制原码形式转换)

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 0001 1110 1111 1110 0101 0000 0110 0110 1011 0110 1011 1000 1001 (填64bit中的原始形式)

尾数转换为十进制小数形式是 0.1210680247083664173857187051908113062381744384765625
(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 1.1210680247083664173857187051908113062381744384765625
(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. -0501532.2351050 (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 0001 1110 1001 1100 0111 0000 1111 0000 1011 1111 0101 1101 0111 1001 (c1 1e 9c 70 f0 bf 5d 79) (不是手算，用P.5方式打印)

(2) 其中：符号位是_____1_____

指数是_____100 0001 0001 _____ (填64bit中的原始形式)

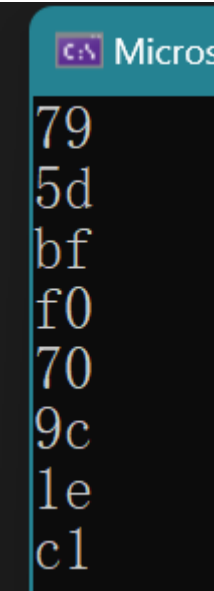
指数转换为十进制形式是_____1041_____ (64bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是_____18_____ (64bit中的原始形式按IEEE754的规则转换)

尾数是_____1110 1001 1100 0111 0000 1111 0000 1011 1111 0101 1101 0111 1001_____ (填64bit中的原始形式)

尾数转换为十进制小数形式是 -0.9131936458778382448286947692395187914371490478515625
(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 -1.9131936458778382448286947692395187914371490478515625 (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. 0.002351050 (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 0010 1000 0010 0101 1011 1000 1111 0111 0010 1100 1111 (3f 63 42 82 5b 8f 72 cf) (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)

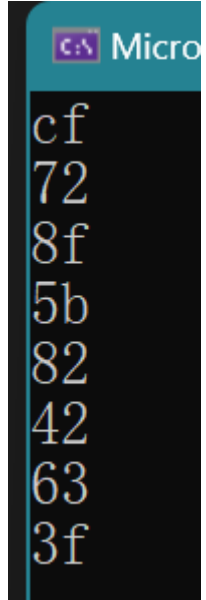
指数转换为十进制形式是____ 1014 ____ (64bit中的原始形式按二进制原码形式转换)

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 0011 0100 0010 1000 0010 0101 1011 1000 1111 0111 0010 1100 1111 (填64bit中的原始形式)

尾数转换为十进制小数形式是 0.2037375999999999631739910910255275666713714599609375
(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 1.2037375999999999631739910910255275666713714599609375
(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. -0.000501532 (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 0100 0000 0110 1111 0010 0111 1100 0100 1101 1001 1101 0011 0000 1001 (bf 40 6f 27 c4 d9 d3 09) (不是手算，用P.5方式打印)

(2) 其中：符号位是 1

指数是 011 1111 0100 (填64bit中的原始形式)

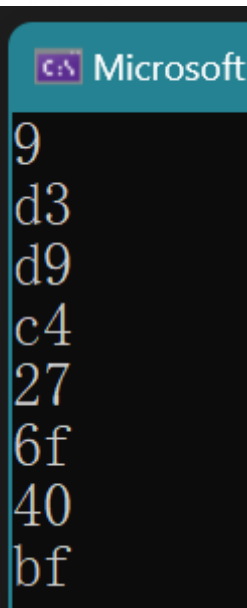
指数转换为十进制形式是 1012 (64bit中的原始形式按二进制原码形式转换)

指数表示的十进制形式是 -11 (64bit中的原始形式按IEEE754的规则转换)

尾数是 0000 0110 1111 0010 0111 1100 0100 1101 1001 1101 0011 0000 1001 (填64bit中的原始形式)

尾数转换为十进制小数形式是 -0.0271375359999999066218379084602929651737213134765625 (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 -1.0271375359999999066218379084602929651737213134765625 (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？

float型数据的32bit依规则分为3部分来表示一个单精度浮点数：1bit符号位、8bit指数位、23bit尾数位。尾数的正负由符号位决定，0为正，1为负；尾数是将浮点数（可表示范围内）转化为2进制小数后，表示为 $1.**\cdots \times 2^x$ 的形式，隐藏高位1，尾数补0后将小数点后23位储存在尾数部分；指数由指数位的值偏置表示，对于float型偏置值为127，若此值转为10进制后大于127为正，小于则为负；指数即上文提到的科学计数法表示中2的指数x，+127后转为2进制储存在指数位中。

(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？

单精度浮点数中尾数位占23bit，再加上隐藏高位的小数点前的1，共可表示24位有效数字，同时 $10^7 < 2^{24} = 16,777,216 < 10^8$ ，故只能精确表示7位十进制数；同时，由于储存机制，计算机存储一个浮点数时是将此数转为与其最接近的“精度”，而这一精度为不均匀的离散性分布。

32位浮点数指数部分取值最大为127（1111 1111用来表示 non-number），而尾数部分最大值为 $1.111\cdots$ 约为10进制中的2，所以float型数据最大为 2×2^{127} ，即 3.4×10^{38} 。

由程序可见f1, f2精确度不到7位

f3, f4精确度为7位

```
float f1 = 1024.001; //均为7位有效数
float f2 = 1024.002;
float f3 = 1024.003;
float f4 = 1024.004;
printf("%.17f\n", f1); //精确度到7位应为 1024.001xxxxxxxxxxxxxx
printf("%.17f\n", f2); //精确度到7位应为 1024.002xxxxxxxxxxxxxx
printf("%.17f\n", f3); //精确度到7位应为 1024.003xxxxxxxxxxxxxx
printf("%.17f\n", f4); //精确度到7位应为 1024.004xxxxxxxxxxxxxx
return 0;
```

Microsoft Visual Studio 调试控制台

```
1024.000976562500000000
1024.001953125000000000
1024.003051757812500000
1024.004028320312500000
C:\Users\C10H15N\Desktop
```



§. 基础知识题 - 浮点数机内存储格式(IEEE 754)理解

3、总结

(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？

double型数据的64bit依规则分为3部分来表示一个双精度浮点数：1bit符号位、11bit指数位、52bit尾数位。尾数的正负由符号位决定，0为正，1为负；尾数是将浮点数（可表示范围内）转化为2进制小数后，表示为 $1.**\cdots \times 2^x$ 的形式，隐藏高位1，尾数补0后将小数点后52位储存在尾数部分；指数由指数位的值偏置表示，对于double型偏置值为1023，若此值转为10进制后大于127为正，小于则为负；指数即上文提到的科学计数法表示中2的指数x，+1023后转为2进制储存在指数位中。

(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？

双精度浮点数中尾数位占52bit，再加上隐藏高位的小数点前的1，共可表示53位有效数字，同时 $10^{15} < 2^{53} = 16,777,216 < 10^{16}$ ，故只能精确表示15位十进制数；同时，由于储存机制，计算机存储一个浮点数时是将此数转为与其最接近的“精度”，而这一精度为不均匀的离散性分布。

64位浮点数指数部分取值最大为1023（111 1111 1111用来表示 non-number），而尾数部分最大值为 $1.111\cdots$ 约为10进制中的2，所以float型数据最大为 2×2^{1023} ，即 1.7×10^{308} 。

由程序可见d1, d4精确度不到16位

d2, d3精确度为16位

```
double d1 = 1048576.000000001; //均为16位有效数字
double d2 = 1048576.000000002;
double d3 = 1048576.000000003;
double d4 = 1048576.000000004;
printf("%.34lf\n", d1); //精确度到16位应为1048576.000000001xxxxxxxxx
printf("%.34lf\n", d2); //精确度到16位应为1048576.000000002xxxxxxxxx
printf("%.34lf\n", d3); //精确度到16位应为1048576.000000003xxxxxxxxx
printf("%.34lf\n", d4); //精确度到16位应为1048576.000000004xxxxxxxxx
```

Microsoft Visual Studio 调试控制台

```
1048576.0000000009313225746154785156250000
1048576.0000000020954757928848266601562500
1048576.0000000030267983675003051757812500
1048576.0000000039581209421157836914062500
```

C:\Users\C10H15N\Desktop\Homework\Coding\w



§. 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，是以偏置法标识出来的，将计算所得指数加上偏置值后转为2进制，存储在指数位中。

(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;
}
```

1.2并不可以准确的表示为一个float型常量，在赋值过程中会舍去部分数据，所以编译器报warning；1.25可以精确表示为一个float型常量，复制过程中舍去的只有0，所以不加后缀F的情况下也不会报warning。

如果double型常量赋值给float型变量时，没有加上后缀F，且其值不能精确地表示为float型的值，则会报warning；如果double型常量赋值给float型变量时，加上后缀F，或者其值可以精确地表示为float类型的值，则不会报warning。