

§ . 基础知识题 – 浮点数机内存存储格式(IEEE 754)理解



要求:

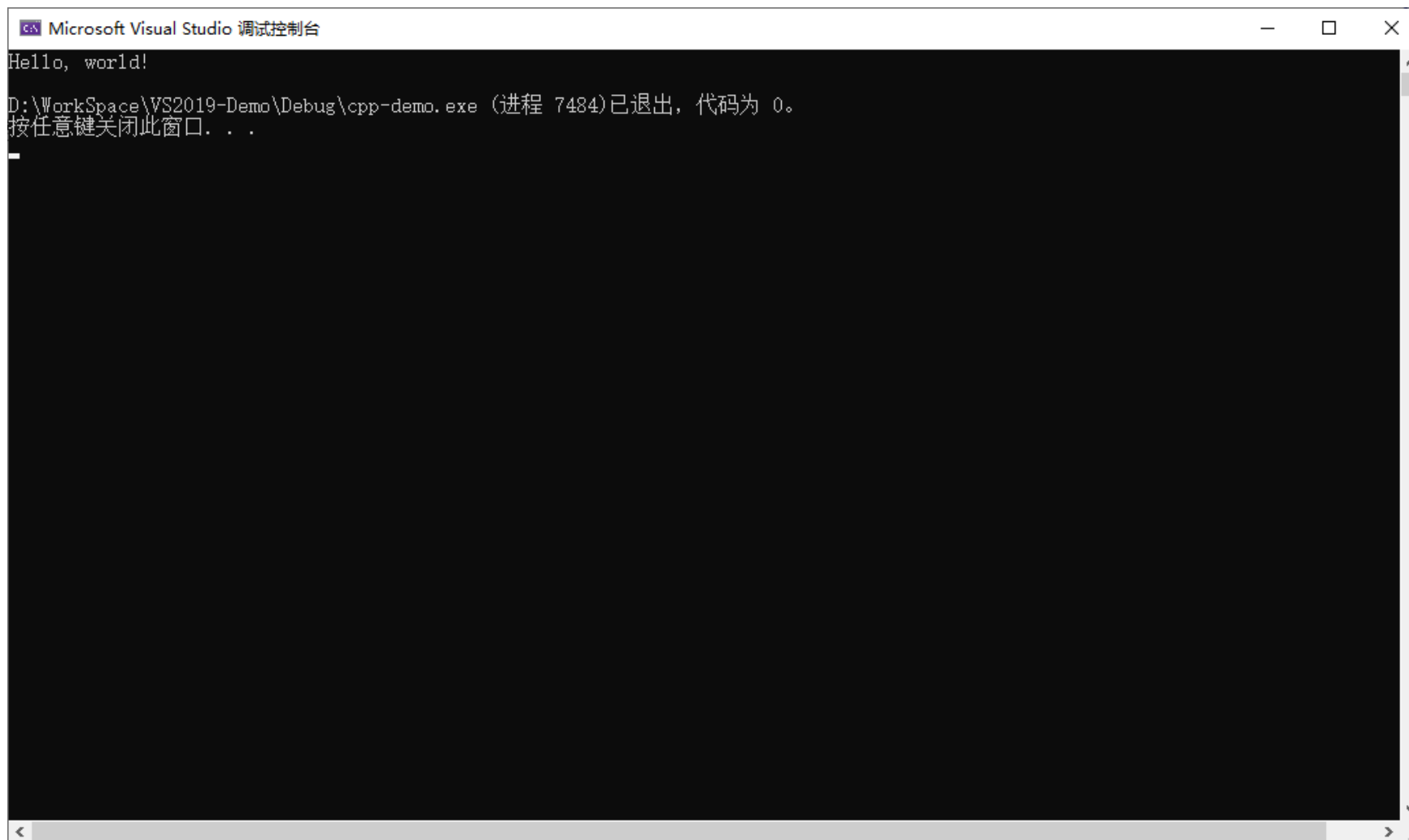
- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
- 4、转换为pdf后提交
- 5、**9月15日前**网上提交本次作业（在“文档作业”中提交）

§. 基础知识题 – 浮点数机内存储格式(IEEE 754)理解

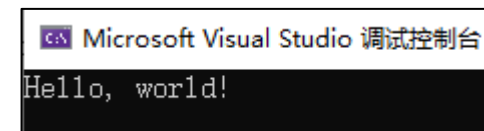


贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

A screenshot of the Microsoft Visual Studio debug console window. The window is titled "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 window is large and shows the full content of the console.

例：有效贴图

A screenshot of the Microsoft Visual Studio debug console window, showing only the output text: "Hello, world!". The window is titled "Microsoft Visual Studio 调试控制台".



§. 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

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

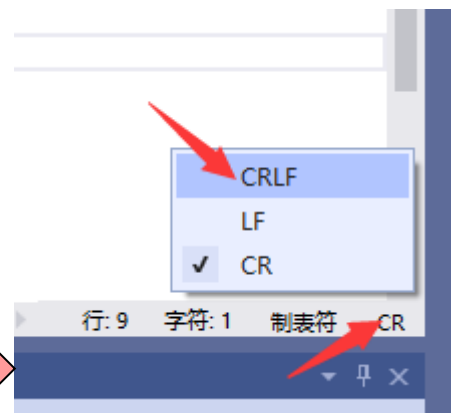
100 % 未找到相关问题 行: 9 字符: 1 制表符 CR

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 = 654.32f;
    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;
}
```

Microsoft
79
e9
f6
42

C:\N
7b
94
23
44

上例解读：单精度浮点数654.32f，在内存中占四个字节，四个字节的值依次为0x44 0x23 0x94 0x7b（按打印顺序逆向取）

转换为32bit则为：0100 0100 0010 0011 1001 0100 0111 1011

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 = 4.32e1;
    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

Micro
9a
99
99
99
99
99
45
40

上例解读：双精度浮点数4.32e1，在内存中占八个字节，八个字节的值依次为0x40 0x45 0x99 0x99 0x99 0x99 0x99 0x9a(逆向)

转换为64bit则为：0100 0000 0100 0101 1001 1001 1001 1001 1001 1001 1001 1001 1001 1010

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/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



§. 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)

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: 1234567.7654321

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 1001 1001 0110 1011 0100 0011 1110 (49 96 b4 3e)

(2) 其中: 符号位是 0

指数是 1001 0011 (填32bit中的原始形式)

指数转换为十进制形式是 147 (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 20 (32bit中的原始形式按IEEE754的规则转换)

1001 0011

- 0111 1111

= 0001 0100 (0x14 = 20)

尾数是 001 0110 1011 0100 0011 1110 (填32bit中的原始形式)

尾数转换为十进制小数形式是 0.1773755503845214844 (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 1.1773755503845214844 (加整数部分的1后)

001 0110 1011 0100 0011 1110 = $2^{-3} + \dots + 2^{-22}$

= 0.1773755503845214844 => 加1 => 1.1773755503845214844

$1.1773755503845214844 * 2^{20} = 1234567.75$ (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:

1234567 = 0001 0010 1101 0110 1000 0111 (整数部分转二进制为21位)

0.7654321 = 11000... (小数部分转二进制, 再要3位就够了)

1234567.7654321 = 0001 0010 1101 0110 1000 0111.110 = 1.0010 1101 0110 1000 0111 110 x 2^{20} (移20位)

符号位: 0

阶 码: $20 + 127 = 147 = 1001 0011$

尾 数: 0010 1101 0110 1000 0111 110 (23位)

001 0110 1011 0100 0011 1110 (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

1、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”)

A. 1234567.7654321 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

注：尾数为正、指数为正

学号：1850772.2770581

(1) 得到的32bit的机内表示是：_0100 1001 1110 0001 1110 1100 1010 0010__ (49 e1 ec a2)

(2) 其中：符号位是__0__

指数是__1001 0011__ (填32bit中的原始形式)

指数转换为十进制形式是__147__ (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__20__ (32bit中的原始形式按IEEE754的规则转换)

尾数是_ 110 0001 1110 1100 1010 0010 _ (填32bit中的原始形式)

尾数转换为十进制小数形式是_ 0.7650339603424072 _ (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__ 1.7650339603424072 __ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

B. -7654321.1234567 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

注：尾数为负、指数为正

学号：-2770581.1850772

(1) 得到的32bit的机内表示是：__1100 1010 0010 1001 0001 1010 0101 0101__ (ca 29 1a 55)

(2) 其中：符号位是__1__

指数是__100 1010 0__ (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__010 1001 0001 1010 0101 0101__ (填32bit中的原始形式)

尾数转换为十进制小数形式是__0.3211160898208618__ (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__1.3211160898208618__ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

C. 0.001234567 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

注：尾数为正、指数为负

学号：0.001850772

(1) 得到的32bit的机内表示是：__0011 1010 1111 0010 1001 0101 1001 1010

(2) 其中：符号位是__0__

指数是__011 1010 1__ (填32bit中的原始形式)

指数转换为十进制形式是__117__ (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__-10__ (32bit中的原始形式按IEEE754的规则转换)

尾数是__ 111 0010 1001 0101 1001 1010 __ (填32bit中的原始形式)

尾数转换为十进制小数形式是__ 0.8951904773712158 __ (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__ 1.8951904773712158 __ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

D. -0.007654321 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

注：尾数为负、指数为负

学号：-0.002770581

(1) 得到的32bit的机内表示是：__ 1011 1011 0011 0101 1001 0010 1010 0011__

(2) 其中：符号位是__1__

指数是__ 011 1011 0 __ (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__ 011 0101 1001 0010 1010 0011__ (填32bit中的原始形式)

尾数转换为十进制小数形式是__ 0.4185374975204468 __ (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__ 1.4185374975204468 __ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

A. 1234567.7654321 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

注：尾数为正、指数为正

学号：1850772.2770581

(1) 得到的64bit的机内表示是：0100 0001 0011 1100 0011 1101 1001 0100 0100 0110 1110 1101 0010 0111 1001 0111

(2) 其中：符号位是__0__

指数是__100 0001 0011__ (填64bit中的原始形式)

指数转换为十进制形式是__1043__ (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__20__ (64bit中的原始形式按IEEE754的规则转换)

尾数是 1100 0011 1101 1001 0100 0100 0110 1110 1101 0010 0111 1001 0111 (填64bit中的原始形式)

尾数转换为十进制小数形式是__0.76503398614702234__ (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__1.76503398614702234__ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

B. -7654321. 1234567 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

学号：-2770581. 1850772

注：尾数为负、指数为正

(1) 得到的64bit的机内表示是：_1100 0001 0100 0101 0010 0011 0100 1010 1001 0111 1011 0000 1001 1100 0001 0101_

(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_ 0101 0010 0011 0100 1010 1001 0111 1011 0000 1001 1100 0001 0101_ (填64bit中的原始形式)

尾数转换为十进制小数形式是_ 0. 32111605886325845 _____ (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. 32111605886325845 _____ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

C. 0.001234567 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

学号:0.001850772

注：尾数为正、指数为负

(1) 得到的64bit的机内表示是：_0011 1111 0101 1110 0101 0010 1011 0011 0100 1101 1001 0111 0010 1110 1111 1110 _____

(2) 其中：符号位是__0_____

指数是__011 1111 0101_____ (填64bit中的原始形式)

指数转换为十进制形式是__1013_____ (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__ -10 _____ (64bit中的原始形式按IEEE754的规则转换)

尾数是1110 0101 0010 1011 0011 0100 1101 1001 0111 0010 1110 1111 1110 (填64bit中的原始形式)

尾数转换为十进制小数形式是0.89519052800000010 _____ (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1.89519052800000010 _____ (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8bit中间加一个空格或- (例：“11010100 00110001” 或 “11010100-00110001”)

D. -0.007654321 (此处假设学号是1234567，各人换成自己的学号，按1234567做的0分!!!)

学号:-0.002770581

注：尾数为负、指数为负

(1) 得到的64bit的机内表示是：_1011 1111 0110 0110 1011 0010 0101 0100 0101 1001 0010 0110 0011 1111 1000 1101_____

(2) 其中：符号位是_1_____

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_ 0110 1011 0010 0101 0100 0101 1001 0010 0110 0011 1111 1000 1101 (填64bit中的原始形式)

尾数转换为十进制小数形式是__ 0.41853747199999991 _____ (64bit中的原始形式按二进制原码形式转换)

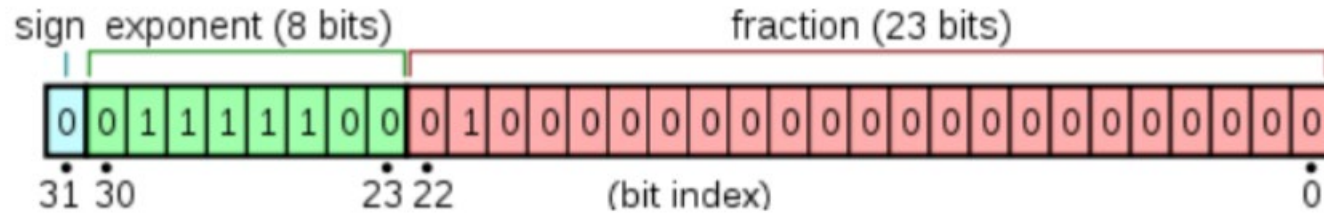
尾数表示的十进制小数形式是__ 1.41853747199999991 _____ (加整数部分的1)



§. 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？



32位浮点数内存占用示意图, 共使用了32个小格子

分为三段表示：其中

sign: 符号位，即图中蓝色的方块，0为正，1为负，

biased exponent: 偏移后的指数位，即图中绿色的方块表示 2^{n-127} 次方，第一位不为符号位，用-127来表示指数的正负

fraction: 尾数位，即图中红色的方块，该数为unsigned数字，表示 $M \cdot 2^{(n-127)}$

浮点数F的值可以表示为 $F = (-1)^{\text{符号位}} \cdot \text{尾数} \cdot 2^{(\text{指数}-127)}$



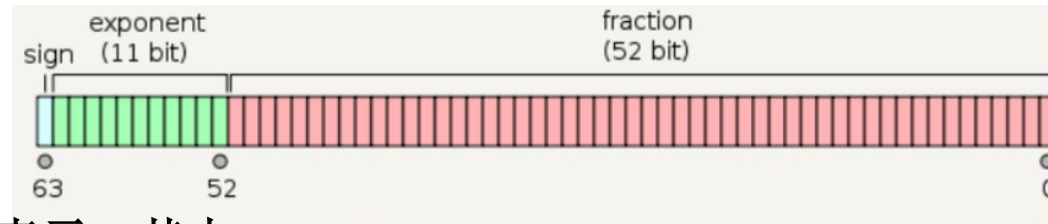
(2) 为什么float型数据只有7位十进制有效数字？为什么最大只能是 3.4×10^{38} ？
有些资料上说有效位数是6~7位，能找出6位/7位不同的例子吗？

float型最大为01111111 01111111 11111111 11111111 十进制转化约是： $3.4 * 10^{38}$

Float的尾数有23位： $2^{23} = 8388608$ ，一共七位，这意味着最多能有7位有效数字，但绝对能保证的为6位，也即float的精度为6~7位有效数字

比如说1234567.7654321转换为float再转回来就是 $1.1773755503845214844 * 2^{20} = 1234567.75$ 则只有6位有效数字

(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？



双精度类似单精度 也分为三段表示：其中

sign: 符号位，即图中蓝色的方块，0为正，1为负，

biased exponent: 偏移后的指数位，即图中绿色的方块表示 2^{n-1023} 次方，第一位不为符号位，用-1023来表示指数正负

fraction: 尾数位，即图中红色的方块，该数为unsigned数字，表示 $M * 2^{(n-1023)}$

浮点数F的值可以表示为 $F = (-1)^{\text{符号位}} * \text{尾数} * 2^{(\text{指数}-127)}$



(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？

最大的double : 01111111 11101111 11111111 11111111 11111111 11111111 11111111 11111111

转换的十进制数大约是： $1.7 * 10^{308}$

Double的位数有52位： $2^{52} = 4503599627370496$ ，一共16位，同理，double的精度为15~16位如：

比如说 98765432101234567890，double型为01000100 00010101 01101010 10010101 00110100
11011011 10001000 00110010，转化为十进制为98765432101234573893.2...，则有十五位有效数字