

Problem 1: Floating Point (floating)

In this problem, we will investigate how floating-point numbers are represented in memory. Recall that a float is a 32-bit value with a single sign bit, eight exponent bits, and 23 mantissa bits. Specifically, a floating point number x with sign bit 'sign', exponent e , and mantissa bits m_0, m_1, \dots, m_{22} can be written¹

$$x = (-1)^{\text{sign}} \cdot (1.m_{22}m_{21}m_{20}\dots m_0) \cdot 2^{e-\text{bias}}$$

where the mantissa is, of course, in base two. You will be given a list of N floating point values x_1, x_2, \dots, x_N . For each x_i , your program should write its binary representation to the output file as indicated below.

Suggested approach: You'll need to use bitwise operations, but you cannot do so on a floating-point number directly. Instead, you will need a way of considering a variable as either a float or an unsigned int. We will use a union, which is valid in this case because we assume the size of the two data types is the same.

```
union float_bits {
    float f;
    unsigned int bits;
};

// print_hex( 5.0f ) outputs "The float looks like 0x40a00000 in hex."
void print_hex( float f ) {
    union float_bits t;
    t.f = f;
    printf( "The float looks like 0x%x in hex.\n", t.bits );
}
```

Input Format

Line 1: One integer N

Lines 2... $N + 1$: Line $i + 1$ contains floating point number x_i

Sample Input (file floating.in)

```
3
1.5
0.15625
-7.333
```

¹Except for the case where x is a denormal floating point number, as discussed in class, in which case the (unbiased) exponent is -126 and mantissa is written $0.m_{22}m_{21}m_{20}\dots m_0$.

Lines 1 \dots N : Line i contains a representation of the floating-point number x_i , formatted as shown in the sample output.

$$\begin{aligned} &1.10000000000000000000 * 2^0 \\ &1.01000000000000000000 * 2^{-3} \\ &-1.1101010101001111110000 * 2^2 \end{aligned}$$

MIT OpenCourseWare

<http://ocw.mit.edu>

6.S096 Effective Programming in C and C++

IAP 2014

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.