**作业——1**

1. 请证明补码加法公式：[ｘ]补＋[ｙ]补＝[ｘ＋ｙ]补（mod 2w）。 [\*]补表示整型数据\*的补码表示，机器字长为W。
2. 将8位无符号数130转换为8位浮点数（exp域宽度为4 bits, frac域宽度为3bits）

Exp = ?

Frac = ?

1. We are running programs on a machine with the following characteristics:

* Values of type int are 32 bits. They are represented in two’s complement （**补码**）, and they are right shifted arithmetically. Values of type unsigned are 32 bits.
* Values of type float are represented using the 32-bit IEEE floating point format, while values of type double use the 64-bit IEEE floating point format.

We generate arbitrary values x, y, and z, and convert them to other forms as follows:

*/\* Create some arbitrary values \*/*

*int x = random();*

*int y = random();*

*int z = random();*

*/\* Convert to other forms \*/*

*unsigned ux = (unsigned) x;*

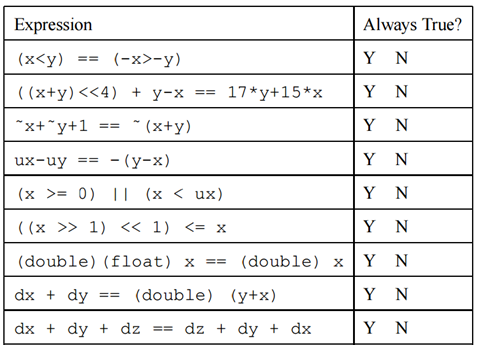
*unsigned uy = (unsigned) y;*

*double dx = (double) x;*

*double dy = (double) y;*

*double dz = (double) z;*

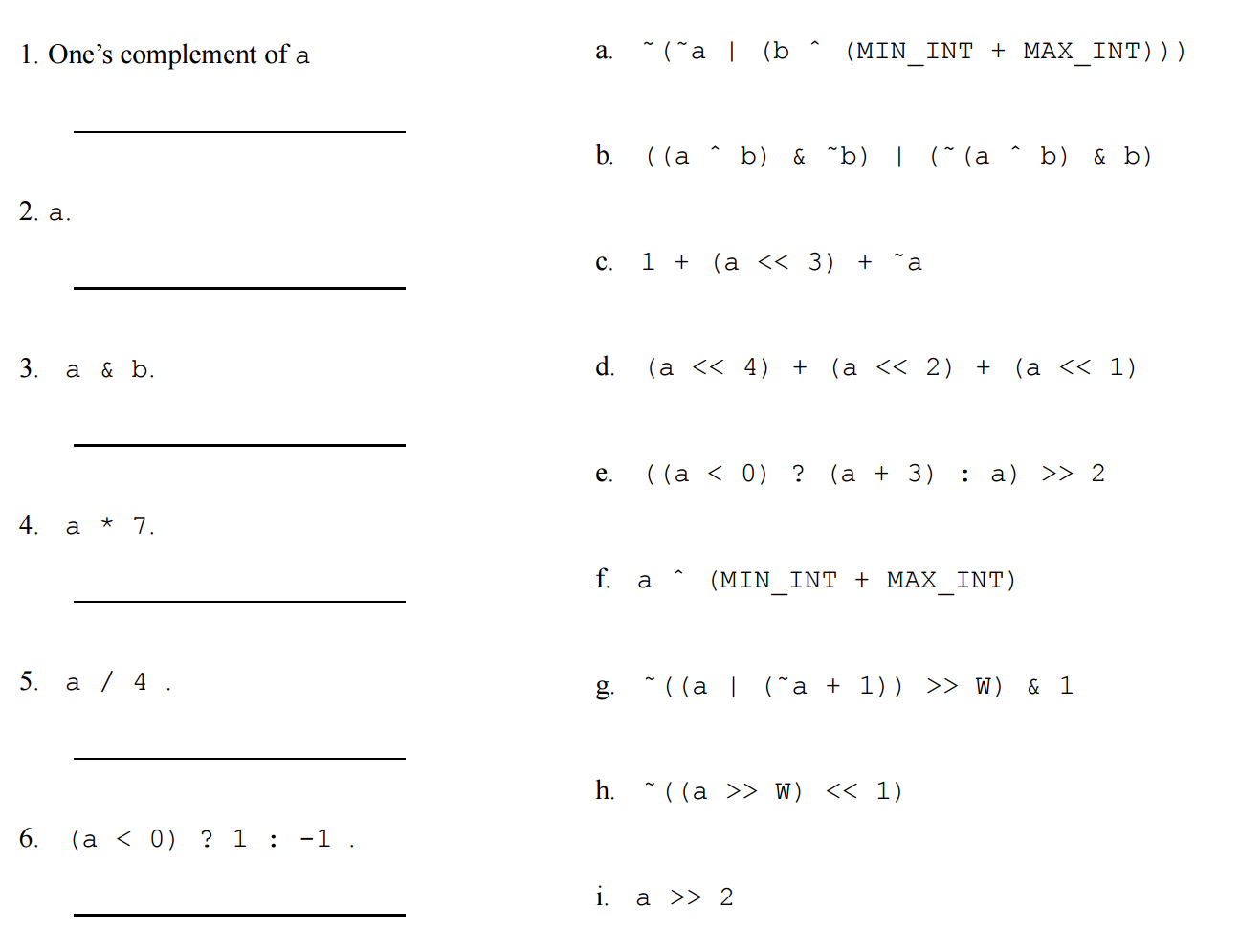
For each of the following C expressions, you are to indicate whether or not the expression always yields 1.



1. In the following questions assume the variables **a and b are signed integers** and that the machine uses two’s complement representation. Also assume that MAX\_INT is the maximum integer, MIN\_INT is the minimum integer, and W is one less than the word length (e.g., W = 31 for 32-bit integers). Match each of the descriptions on the left with a line of code on the right (write in the letter).

//1's Complement：反码，即按位取反

//2's Complement：补码



1. Match each of the assembler routines on the left with the equivalent C function on the right.

