1. Consider a 16-bit floating-point representation based on the IEEE floating-point format, with one sign bit, seven exponent bits (k = 7), and eight fraction bits (n = 8). The exponent bias is $2^{7-1} - 1 = 63$.

   Fill in the table that follows for each of the numbers given except those marked "--", with the following instructions for each column:

   Hex: The four hexadecimal digits describing the encoded form.

   M: The value of the significand. This should be a number of the form x or x/y, where x is an integer, and y is an integral power of 2. Examples include: 0, 67/64, and 1/256.

   E: The integer value of the exponent.

   V : The numeric value represented. Use the notation x or x × 2z, where x and z are integers.

   You are recommended to fill in the blanks in fractions.

Description	Hex	M	E	V
-0				--
Smallest value > 1				
256				--
Largest denormalized				
-∞		--	--	--
Number with hex representation 3AA0	--			

2. Consider the 16-bit floating-point representation form in problem1 above. Calculate the product of (1 0101011 10011010)2 and (0 1000101 01010001)2, and then round the results with Round-to-Even rounding modes. (NOTE: Please give your steps detailed and the result should be in binary form)

3. We are running programs on a machine where values of type int have a 32-bit two's-complement representation. Values of type float use the 32-bit IEEE format, and values of type double use the 64-bit IEEE format.

   We generate arbitrary integer values x, y, and z, and convert them to values of type double as follows:

```
/* Create some arbitrary values */
int x = random();
int y = random();
```

```
int z = random();
/* Convert to double */
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. Answer with **Yes** or **No** Note that you cannot use an IA32 machine running gcc to test your answers, since it would use the 80-bit extended-precision representation for both float and double.

A. (doubleL) (float) x == dx

B. dx + dy == (double) (x + y)

C. dx + dy + dz == dz + dy + dx

D. dx * dy * dz == dz * dy * dx

E. dx / dx == dy / dy

4. Write down the byte codes of the following Y86 instructions (little-endian):

Y86 Instructions	Byte Codes
addl %edx, %eax	
subl %esi, %edi	
rrmovl %esp, %ebp	
rmmovl %ecx, 0x60(%ebx)	
pushl %eax	