Tutorial Problems- 2

1. Convert the following decimal numbers into IEEE 754 single-precision format and write your answers in hexadecimal form.

   a) 8E-39
   b) - 31.5

1.

This question can be solved using the following steps if it is a NORMAL number.

- **Step 1:** Determine the Sign bit (0: +ve numbers and 1: -ve numbers)
- **Step 2:** Determine the Exponent and store it as an 8-bit number in excess:127 format.
- **Step 3:** Determine the Mantissa (or fraction *f*), if it is between 1 and 2, then subtract 1 to store the remainder in binary fractions format.
- **Step 4:** Put everything together.

a) 8E−39

The number is smaller than the smallest magnitude that can be represented by a normal IEEE 754 number, hence it must be a subnormal number.

Step 1: Compute Sign Bit

Positive, hence sign bit, S, is 0

Step 2: Compute Exponent

IEEE 754 Exponent = 0 for subnormal numbers

Step 3: Compute Mantissa

8E-39/2 $^{-126}$ = 0.680564734 (less than 1)

Convert 0.680564734 into 23 bit binary fraction

round($0.680564734 \times 2^{23}$) = 5708991

Convert 5708991 into hexadecimal first

5708991 = 0x571CBF

= 101 0111 0001 1100 1011 1111B (23 bits binary fraction)

Step 4

Putting everything together

S	E	E	E	E	E	E	E	E	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1
0				0				5				7				1				C				B				F		

b) −31.5

Step 1: Compute Sign Bit

−31.5 is negative, hence sign bit, S, is 1

Step 2: Compute Exponent

floor($\log_2$ (31.5)) = floor (4.977) = 4

Exponent = 4

Since exponent in IEEE754 is excess 127 form

or 127 + 4 = 131

131 = 10000011B

Step 3: Compute Mantissa

$31.5 \times 2^{-4} = 1.96875$ (between 1 and 2)

Convert 0.96875 (after subtract 1) into 23 bit binary fraction

$\text{round}(0.96875 \times 2^{23}) = 8126464$

Convert 8126464 into hexadecimal first

8126464 = 0x7C0000

= 111 1100 0000 0000 0000 0000B (23 bits binary fraction)

Step 4

Putting everything together

S	E	E	E	E	E	E	E	E	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C				1				F				C				0				0				0				0			

2. Add the following 8-bit binary number pairs using 8-bit addition (result stored in eight bits only) and comment on whether there is an overflow if they were both unsigned numbers or they were both signed numbers.

   a) 0011 1111 and 1101 0011
   b) 0111 1111 and 0111 0011
   c) 1000 1111 and 1101 0011

2)

	Binary	Unsigned	Signed
a)	0011 1111B	63	63
+	1101 0011B	+ 211	− 45
	0001 0010B, C =1	18(wrong)	18 (correct)

The Carry flag, C, can be used to indicate Unsigned Overflow.

	Binary	Unsigned	Signed
b)	0111 1111B	127	127
+	0111 0011B	+ 115	+ 115
	1111 0010B, C = 0	242(correct)	−14 (wrong)

The Carry flag, C, cannot be used to indicate signed Overflow. A separate flag, V, is used to indicate sign overflow. V will be set to 1 in this case.

	Binary	Unsigned	Signed
c)	1000 1111B	143	− 113
+	1101 0011B	+ 211	− 45
	0110 0010B, C =1	98(wrong)	98 (wrong)

- Both Unsigned and Signed Overflow. C = V = 1.
- The overflow flag, V, will be set if there is a carry out of Bit 7 but not from Bit 6 (Part c) or if there is a carry out of Bit 6 but not from Bit 7 (Part b).
- The same principle can be extended to higher word length, for a 32-bit addition, the sign bit will be Bit 31 instead of Bit 7, and Bit 30 instead of Bit 6 should be checked for carry out.