CSA answer sheet for HW4 GG2961

Q.1

A. 0110 1110 + 1001 1111

- Unsigned addition: 110 + 159 = 269. 269 > 2^8 - 1 => overflow has occurred since 269
cannot be represented with 8 bits
- Signed addition: 110 - 97 = 13. 13 < 2^7 - 1 => no overflow has occurred since 13 can
be represented with 8 bits in signed form

B. 1111 1111 + 0000 0001

Unsigned addition: 255 + 1 = 256 > 2^8 - 1 => overflow has occurred
- Signed addition: -1 + 1 = 0, 0 can be represented in 29s complement form in 8 bits => no
overflow has occurred

C. 1000 0000 + 0111 1111

Unsigned addition: 128 + 127 = 255 = 2^8 -1 => no overflow has occurred since 255 can
be represented with 8 bits
- Signed addition: - 128 + 127 = -1, -1 can be represented in 29s complement form in 8 bits
=> no overflow has occurred

D. 0111 0001 + 0000 1111

Unsigned addition: 113 + 15 = 128 < 2^8 -1 => no overflow has occurred since 128 can
be represented with 8 bits
- Signed addition: 113 + 15 = 128, cannot be represented in 29s complement form => an
overflow will occur

Q.2

To represent the result of the multiplication, we will need at least 16 bits
- ABhex = 171, EFhex = 239
- 239 = (256 - 17), 17 = (16 + 1)
- 256 = 2^8, 16 = 2^4
- To obtain the result,

- Left shift 171 four times and add 171 to it to obtain 171 x 17, let
this value be a
- Left shift 171 eight times to obtain 171 x 256, let this value be b
- Subtract a from b

Q.3
32 bits => Single Precision Floating Point
- Binary representation of 0xdeadbeef:
- 1101 1110 1010 1101 1011 1110 1110 1111
- S: 1 => -1
- Exponent: 10111101 => 189, Exponent Bias = 189 - 127 = 62
- Fraction: 01011011011111011101111 => 1 + 2^-2 + 2^-4 + 2^-5 + 2^-7
+ 2^-8 + 2^-10 +
2^-11 + 2^-12 + 2^-13 + 2^-14 + 2^-16 + 2^-17 + 2^-18 + 2^-20 + 2^-21
+ 2^-22 + 2^-23
=
1.35738933086395262997999
- Decimal representation: -1.35738933086395262997999 * 2^62 =
-6,259,853,398,707,797,984.9229547264623

Q.4
Number is positive, sign bit is zero
- 78 in binary representation: 1001110
- 0.75 in binary representation: 110
- 78.75 = 1001110.110 = 1.00111011 x 2^6
- Exponent, 32 bits = 6 + 127 = 133 = 10000101
- Exponent, 64 bits = 6 + 1023 = 1029 = 10000000101
- 32 bit : 01000010100111011000000000000000
-64bit:01000000010110000101000000000000000000000000000000000000000000
00

Q.5
Number is positive, sign bit is zero
- 78 in base16 representation: 4E
- .75 in base16 representation: .C
- 78.75 in base16 representation: 4E.C = 4.EC * 16
- 78.75 in binary representation: 01001110.110
- Shift right by two hexadecimal digits: 0.01001110110 * 16^2
- Bias of 64 added to exponent of 2, 66: 1000010 binary rep
- 32 bit rep with base 16: 01000010010011101100000000000000

Q.6

a)
-1.3625 * 10^-1 = -0.13625
- Number is negative => sign bit is turned on
- 0.13625 binary representation: 0.001000101110000101
- = 1.000101110000101 * 2^-3
- Exponent = -3 + 15 = 12
- 16 bit representation: 10110000101110000101
- Bits marked in red will not be stored due to the limited precision
of 10 bits, truncation
error
- In case of a 32 bit representation, -0.13625 could9ve been
represented perfectly
- Range of numbers in single precision : 2^(-126) to 2^(+127)
- Range of numbers in single precision : 2^(-14) to 2^(+15)

(b) Calculate the sum of 1.6125 ×10
1
(A) and 3.150390625 ×(10)^−1
(B) by
hand, assuming operands A and B are stored in the 16- bit half precision described in
problem a. above Assume 1 guard, 1 round bit, and 1 sticky bit, and
round to the nearest even. Show all the steps.
1.6125 * 10^1 = 16.125
- Binary representation: 10000.001 = 1.0000001 * 2^4 (A)
- Exponent with bias: 15+4 = 19, binary representation: 10011
- 16 bit representation: 0100110000001000
- 3.150390625 ×10^-1 = 0.3150390625
- Binary representation: 0.0101000010100110011 = 1.01000010100110011
* 2^-2 (B)
- Exponent with bias: 15 -2 = 13, binary representation: 1101
- 16 bit representation: 0011010100001010
We can9t directly add the binary representations because they don9t
have the same exponents,
we can shift (A) to left by 6 bits
- 1.0100001010
- + .00010000001 (Truncation error)
- 1.0101001010 * 2^4
- Already normalized, no errors while adding the numbers. I do not
know what to do with
the Truncation and representation errors

Q.7

Name	Mantissa bits	Exponent bits	Exponent bias	Smallest positive number
Single precision	23	8	127	2^(-126-(23))
fp16	10	5	15	2^(-14-(10))
bfloat16	7	8	127	2^(-126-(7))

Q.8

Number	Binary	Decimal
0	0 000 000	0.0
-0.125	1 000 000	-0.125
Smallest positive normalized number	0 001 000	0.25
largest positive normalized number	0 110 111	15
Smallest positive denormalized number > 0	0 000 001	0.125
largest positive denormalized number > 0	0 000 111	0.10

Smallest normalized positive:
S = 0, E = 001(2^(1-3) = 2^-2), F = 000
Binary: 0 001 000
Largest positive normalised:
S = 0, E = 110 (max value), F = 111
Binary: 0 110 111
Smallest denormalized positive number:
S = 0, E = 000, F = 001
Binary: 0 000 001
Largest positive denormalized number:
S = 0, E = 000, F = 111
Binary: 0 000 111

b.

Let a = 1 110 111, b = 0 110 111, and c = 0 000 001. Then (a + b) + c = c, because a and b cancel each other, while a + (b + c) = 0, because b + c = b (c is very small relative to b and is lost in the addition).