I pledge my honor that I have abided by the Stevens Honor System.
Haig Emirzian

Quick Check 1.1

1a.) True, because binary numbers can also represent negative numbers. For example, 1010b is 6 in binary. However, if you take the leading 1 as a negative sign, then 1010b becomes -2.

1b.) False, since changing the negative number to signed would lead to a larger number than the signed positive number because the MSB of negative numbers is 1. For example, 0100 changed to a signed negative number is 1100, but a signed positive number is 0100.

2a.) **Unsigned**: 2^8 - 1 = 255; 0 because 255 in binary is 11111111. When you add 1 to the binary number it equates to 00000000.

Two's Complement: 2^7-1 = 127; -128 because 127 in binary is 01111111 and when you convert it to two's complement, you would turn all the 1's into 0's, vice versa, and add 1: 10000000. Since MBS is 1, the number can be calculated by plugging in n=8 bits into -2^{7-1} = -128.

2b.) **0 unsigned**: 0b0000
0 two's complement: 0b0000
1 unsigned: 0b0001
1 two's complement: 0b1111
-1 unsigned: Impossible because the category is unsigned for a negative number, not signed.
-1 two's complement: 1b1111

*To convert to two's complement, you must exchange each 1 for a 0 and each 0 for a 1. Then add 1, and you should have your two's complement.

2c.) **Unsigned**: 17 is 0b0001 0001; Impossible for -17 because the category is unsigned for a negative number, not signed.

Two's Complement: 17 is 0001 0001unsigned is the same as two's complement. -17 is 0001 0001 because the signed binary number for -17 is 1110 1111, and converting to two's complement would lead to exchanging all 1's to 0's, vice versa, and add 1. Therefore, the two's complement for -17 is 0001 0001.

3a.) 9 bits because 2^8 - 1= 255, which means it takes 9 digits in two's complement to reach 256 when adding the signed digit in the MBS position.

3b.) 7 bits because 56 in two's complement has 7 bits when considering the MBS that identifies that 56 is positive. 56: 011 1000

3c.) 8 bits because 127 in two's complement has 8 bits when considering the MBS that identifies that 127 is positive (127: 0111 1111). 8 bits because -127 in two's complement is 1000 0001.

3d.) 2^{40} * 12 * 8 because 2^{40} is equivalent to 1TB. Then multiply by 12 for 12TBs, and multiply by 8 to convert to bits.

Quick Check 1.2

1.) False, because you can only get an overflow error when two of the same signs are added and the sum of the two binary numbers is negative. This is because it extends past the range between -2^{d-1} and $2^{d-1} - 1$.

Quick Check 1.3

1.) You would have to shift twice. First, complete a logical shift left 4 spaces. So 1100 0101 would be 0101 0000. The empty spaces are replaced by 0's as a filler. Then you would have to complete a logical shift right 4 spaces, which would result 0000 0101. Again, the empty spaces are replaced by 0's as a filler.