1.1:

a) 0b10001110=142=0x8E, 0xC3BA=0b1100001110111010=50106, 81=0b01010001=0x51, 0b100100100=292=0x124, 0xBCA1=48289=0x1011110010100001, 0=0b000000=0x0, 42=0x2A=0b00101010, 0xBAC4=0b1011101011000100=47812
b) 16 Ki, 8 Ti, 8 Mi, 256 Pi, 16 Ei, 4 Ti
c) 2^{11}, 2^{59}, 2^{18}, 2^{35}, 2^{26}, 2^{63}

2.2:

1) Unsigned: 255, 256. 2s Complement: 127, 128
2) Unsigned: 00000000, 00000011, -3 cannot represented with an unsigned 8 bit integer. 2s Complement: 00000000, -3=11111101, 3=00000011
3) Unsigned: 42=00101010, -42 cannot be represented with an unsigned 8 bit integer. 2s Complement: 42=00101010, -42=11010110
4) Base 36 ZZZZZZZZ= Base 10 2821109907455, since that's as large as the alphabet can go. This is a very open question, as you can technically continually make the base infinitely large, but with standard alphabet representation, this is as high as you can go.
5) Let x be a binary number, and y be the 1s complement of x. Then, x+y+1=0. This is due to the nature of 1s complement-since it is the inverse, flipped version of the original binary number, adding the two together will always result in having all 1s as the answer. For example, if x was 00110011, then y would be 11001100, and adding them together would result in the number 11111111. So, by adding 1 more to this number, we will always arrive at an answer where we have an overflow of 1, with the rest being zeroes. Here, that would be 00000000, with an extra 1 as an overflow.
6) Decimal is good for intuitive counting, as it is easily readable and understandable for us, likely because we have 10 fingers. It it useful for representing intuitive thinking. Binary is good for systems where binary choices of one option or the other is the quick and optimal way to go-like in computers, where it constantly chooses to do or not do something. Hexadecimal is good for representing very large numbers in a short, compact format.

3.1:

1) 2 bits because there are only 3 things so only 2 bits are needed
2) 2TiB of memory= 2^{41} bytes of memory, which needs 41 bits to be represented for all the bytes, where each byte has one shown value
3) 1, since there's only 1 value