Exercises about representation of information

Add a few explanations to demonstrate how to perform each conversion. For example, from decimal to binary we use powers and then explain the corresponding operations.

1. Convert from decimal to binary:

To do this part we have different types of methods, but in this case I use the next:

I use the powers of 2 depending on it is possible to minus a power of 2 we got the number 1 and if we can´t do this we must put a zero.
I put the 8 first powers of two:

256 128 64 32 16 8 4 2 1

128 then we put a 1
128+64=192 then we put a 1
192+32=224 then we put a 1
224+16=240 this is more than 234 then we put a 0
224+8=232 we put a 1
232+4=236 this is more than 234 then we put a 0
232+2=234 then we put a 1 and to complete the binary number we have to add a zero on the right.

   a. 234 -> 1110 1010
   b. 555 -> 0010 0010 1011
   c. 12321-> 0011 0000 0010 0001
   d. 152-> 1001 1000
   e. 32768-> 1000 0000 0000 0000

2. Convert from binary to decimal:
   a. 1000 0000 0 -> 128
   b. 1011110100 -> 756
   c. 10011101 -> 157
   d. 11111111111-> 2047

3. Convert from hexadecimal to binary:
   a. 45A0

0100010110100000

   b. CF

11001111

   c. AAB2

1010101010110010

   d. 3020

0011000000100000

4. Convert from binary to hexadecimal:
    a. 110001000

188

    b. 100010110

116

5. Complete the following conversions related to octal numeral system:
    a. Convert the numbers from exercise 4 to octal.

610//426

    b. Convert the octal 3020 to binary.

11 000 010 000

6. Fill in the gaps, using all the conversions you need. You have to write the steps to transform each number.

BINARY	DECIMAL	HEXADECIMAL	OCTAL
100001	**33**	21	41
1111 1111	255	**FF**	377
1111 1111	255	FF	**377**
10 0001	33	21	41

33→ 33/2
1 16/2
0 8/2
0 4/2
0 2/2
0 1 -->1000 01

10 0001→ 21 hexadecimal

100 001 → 41 octal

7. How many bits do you need to represent the following numbers in binary?
    a. hexadecimal:

4B -→ 1001011 ->7bits

4AA-→ 10010101010 -> 1byte and 3bits

FF4FA→1111 1111 0100 1111 1010 -> 2 bytes and 4 bits

345F→11 0100 0101 1111-> 1byte and 6 bits

b. decimal: 100, 256, 255, 32, 31, 3, 4350, 1024, 45, 2^{30}, 63

100→ 1100100 -> 7 bits. This can be explained because each number counting from the first one from the left is a bit and 8 numbers is a byte.

256→ 1 0000 0000 -> 1byte and 1 bit

32→ 1 00000 -> 5 bits

31 → 1 111 11 -> this is the same like 2^5 -1 = 31 -> 6 bits

3 → 11 -> 2bits

4350 →

4350/2
0 2175/2
1 1087/2
1 543/2
1 271/2
1 135/2
1 67/2
1 33/2
1 16/2
0 8/2
0 4/2
0 2/2 0 1 **1000 0111 1111 0 ->1 byte and 5bits**

1024 → this is equal to 2^{11} then -> 1 0000 0000 00 ->1 byte and 3 bits

45→ 1 0 1 1 0 1 -> 6bits

2^{30} → 1 0000 0000 0000 0000 0000 0000 0000 0 -> 3 bytes and 6 bits

63 → this is equal to 2^7-1 then -> 1 1111 11 -> 7bits.

8. Solve the following parts using ASCII extended (8 bits).
   a. Write a random text, which contains letters, numbers, and other alphanumeric characters.

SYSTEMS22!

b. Encode to hexadecimal, according ASCII table.

In this case I used the ASCII table to convert each character:

53 | 59 | 53 | 54 | 45 | 4D | 53 | 32 | 32

c. Convert to binary.

Finally using the table to convert to binary, taking 4 numbers for each number, for instance: 53 to binary we take 0101 for 5 and 0011 for the number 3. Leaving 101 0011.

0	0	0	0	**0**
0	0	0	1	**1**
0	0	1	0	**2**
0	0	1	1	**3**
0	1	0	0	**4**
0	1	0	1	**5**
0	1	1	0	**6**
0	1	1	1	**7**
1	0	0	0	**8**
1	0	0	1	**9**
1	0	1	0	**A**
1	0	1	1	**B**
1	1	0	0	**C**
1	1	0	1	**D**
1	1	1	0	**E**
1	1	1	1	**F**

101 0011 | 101 1001 | 101 0100 | 100 0101 | 100 1101 | 101 0011 | 11 0010 | 0011 0010