Exercise 1

Firstly, we convert the integral part and the fractional parts separately (Because $\mathbf{37.26_{10}}$ is a real number).

1. To hexadecimal (base 16 real number)

1.a. Integral part

Start with the source (37.26), **divide it by 16** and use the whole number part as the new source. The remainder is added to the left side and repeat this process until the result is 0.

Iteration	Source	Quotient	Remainder
1	37	2	5
2	2	0	2

Reading the remainders from bottom to top, we get the hexadecimal representation of the integer part: **25**.

1.b. Fractional Part

We use the fractional part (0.26) as the starting point, **multiply it by 16** and then use the new fractional part of the result as the next starting point.

Add the whole integral part of the result to the right side. Repeat until fraction is 0 or we have enough digits.

Iteration	Source	Result	Integral part of result	Fractional part of result
1	0.26	4.16	4	0.16
2	0.16	2.56	2	0.56
3	0.56	8.96	8	0.96
4	0.96	15.36	15_{10} = F_{16}	0.36
5	0.36	5.76	5	0.76
6	0.76	12.16	12_{10} = C_{16}	0.16 (repeat)

The fractions start repeating after 6 steps, so we stop there. This makes the fractional part in hexadecimal $\mathbf{428F5C_{16}}$.

Final Result: 37.26_{10} is equal to $\mathbf{25.428F5C_{16}}$.

2. To Binary (base 2 real number)

2.a. Integral part

To convert the source (37.26_{10}) to binary, we need to convert the integral part (37). Firstly, **divide it by 2**:

Iteration	Source	Quotient	Remainder
1	37	18	1
2	18	9	0
3	9	4	1
4	4	2	0
5	2	1	0
6	1	0	1

So, after reading the remainders from bottom to top, the integral part is **`100101`**$_2$.

2.b. Fractional part

Similar to converting to hexadicimal, we multiply by 2 instead of 16.

Iteration	Source	Result	Integral part of result	Fractional part of result	Destination
1	0.26	0.52	0	0.52	`0`
2	0.52	1.04	1	0.04	`01`
3	0.04	0.08	0	0.08	`010`
4	0.08	0.16	0	0.16	`0100`
5	0.16	0.32	0	0.32	`01000`
6	0.32	0.64	0	0.64	`010000`
7	0.64	1.28	1	0.28	`0100001`
...	...	...	...	...	...

It keeps going endlessly, so we've stopped after 7 steps. The fractional part in binary is **`0100001`**$_2$.

Final Result: 37.26_{10} is approximately **`100101.0100001`**$_2$.

Exercise 2

The sign bit

Because 37.26_{10} is *positive* → the sign is **0**.

The exponent

37.26_{10} when transformed into binary is $\mathbf{100101.0100001_2}$ or we can representation it like this:

$\mathbf{100101.0100001_2} = \mathbf{1.001010100001_2} \times \mathbf{2^5}$.

Therefore, the exponent is **5**. In *Excess_127*, the exponent is $\mathbf{5_{10}} + \mathbf{127_{10}} = \mathbf{132_{10}} = \mathbf{10000100_2}$.

The mantissa

We have the normalized binary representation as $\mathbf{1.001010100001_2} \times \mathbf{2^5}$ → the mantissa is $\mathbf{001010100001_2}$.

The mantissa is currently 12 bits long, so we add 11 more bits to the right to make it 23 bits long.

This gives us the final mantissa: $\mathbf{00101010000100000000000_2}$.

Final Result

Ultimately, the *Excess_127* representation of 37.26_{10} is $\mathbf{01000010000101010000100000000000_2}$, which is the total of these components:

Sign: **0**;

Exponent: $\mathbf{10000100_2}$;

Mantissa: $\mathbf{00101010000100000000000_2}$.

Exercise 3

Normalizing: $\mathtt{101.1010000}_2 \rightarrow \mathtt{1.011010000}_2 \times 2^2$.

The Shifter: **2**.

The Mantissa: $\mathtt{011010000}_2$.

Exercise 4

To perform the **32 - 25 = ?** using binary arithmetic and two's complement representation. There are several steps:

Step 1: Convert the numbers to Binary

- 32 in binary is $\mathtt{100000}_2$.
- 25 in binary is $\mathtt{011001}_2$.

Step 2: Find the Two's Complement of 25

To perform substaction, we need to find the $2'$s complement of 25.
- To do it, we add 1_{10} to the its one's complement (by invert the binary of 25).

Orginal	0	1	1	0	0	1
One's Complement	1	0	0	1	1	0
Add 1_{10}	0	0	0	0	0	1
Two's Complement	1	0	0	1	1	1

Now, the Two's complement of 25 is $\mathtt{100111}_2$.

Step 3: Add the Two's complement of 25 to 32, then convert the result back to Decimal (Base 10)

Firstly, we add the two's complement of 25 to 32:

32_{10}	100000
25_{10} ($2'$s complement)	100111
Sum	1000111

Because of working with 6-bit numbers, we only keep the rightmost 6 bits and discard any carry beyond these bits:
- Result (rightmost 6 bits): $\mathtt{000111}_2$.

Then, convert the binary result $\mathtt{000111}_2$ back to decimal:

$$1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 4 + 2 + 1 = 7.$$

Final Result: The binary substaction **32 - 25** using Arithmatic logic operation inside the CPU, with two's complement representation, the results is **7** (binary result $\mathtt{000111}_2$).

Exercise 5

Firstly, we write down the binary numbers in alignment:

$\boxed{\texttt{10001110}}$

$\boxed{\texttt{10111001}}$

Then, perform the XOR operation bit by bit and find the output:

- XOR (exclusive OR) outputs **`1`** when the bits are different, and **`0`** when the bits are the same.

Input 1	1	0	0	0	1	1	1	0
Input 2	1	0	1	1	1	0	0	1
Output	0	0	1	1	0	1	1	1

So, the result of $\mathbf{10001110_2}$ XOR $\mathbf{10111001_2}$ is $\mathbf{00110111_2}$.