Appendix 1. Multiplication rule for elements of Galois' field GF(256) (AES version).
The bonus problem 1 (1p) in Moodle is based on this appendix.

Galois Field GF(256) or GF(2^8) consists of integers 0 …. 256,
in 8 bit binary form 00000000 …. 11111111

	a	b
Base 10	212	105
Base 2 (8-bit form)	11010100	01101001
Polynomial form (degree 7)	$x^7+x^6+x^4+x^2$	$x^6+x^5+x^3+1$

The product of two numbers a*b (elements of GF(256) is calculated using following steps:
1) Multiply the polynomial representations using standard polynomial multiplication.
2) Reduce all coefficients of the product polynomial mod 2 (rule: odd → 1 and even → 0)
3) Divide the result of step 2 with a specific 8^{th} degree polynomial $q(x) = x^8 + x^4 + x^3 + x + 1$.
Polynomial q(x) is called the "field irreducible polynomial". *)
4) The remainder of the division is a 7^{th} degree polynomial. Reduce coefficients mod 2.
The result is the answer, which can be presented both in base 2 and base 10 number systems.

Step1	$(x^7+x^6+x^4+x^2)*(x^6+x^5+x^3+1) = x^{13} + 2x^{12} + x^{11} + 2x^{10} + 2x^9 + x^8 + 3x^7 + x^6 + x^5 + x^4 + x^2$ wolframalpha: expand $(x^7+x^6+x^4+x^2)*(x^6+x^5+x^3+1)$
Step2	Reducing coefficient mod 2 we get polynomial $x^{13} + x^{11} + x^8 + x^7 + x^6 + x^5 + x^4 + x^2$
Step3	Divide $(x^{13} + x^{11} + x^8 + x^7 + x^6 + x^5 + x^4 + x^2)/(x^8 + x^4 + x^3 + x + 1)$ and calculate the remainder. Remainder will be polynomial $-x^6 + x^5 + x^4 - x^3 + 2x^3 + x$ wolframalpha: remainder $(x^{13} + x^{11} + x^8 + x^7 + x^6 + x^5 + x^4 + x^2)/(x^8 + x^4 + x^3 + x + 1)$
Step 4	Reducing coefficients mod 2 we get the result (reducing rule: odd → 1 , even → 0) $x^6 + x^5 + x^4 + x^3 + x$
Base 2 form (8 bits)	01111010
Base 10	122

In practice using Galois' polynomial version of multiplication is quite complicated and it is difficult to code to a computer program. Surprisingly Python code for Galois' field multiplication has only 12 lines. The multiplication can be done without polynomials: every step can be implemented using only binary forms of numbers.

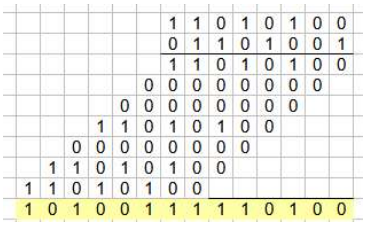
Step1 and Step2 can be implemented using binary multiplication in the following way.

Method is normal multiplication, taught already in primary school.

The last line 101001111110100 is exactly $x^{13} + x^{11} + x^8 + x^7 + x^6 + x^5 + x^4 + x^2$

Picture shows multiplication of 11010100 and 01101001:

Step1 and Step2 using binary form:

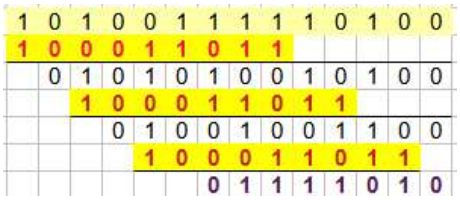


Step3 and Step4 are also done using binary prestentations:

The divisor, so called field irreducible polynomial $q(x) = x^8 + x^4 + x^3 + x + 1$ is in binary form 100011011.
It corresponds to modulus of modular arithmetics and can be used just like number 0: We can add number the modulus or its multiples to any number without changing it. (For example $n \equiv n + 12 \equiv n + 36 \pmod{12}$)

We can add the sequence 100011011 to eliminate 1's of the binary number starting from the left until the result is a 8-bit binary number without changing the result, the polynomial remainder mod q(x).

In our example we need only three additions of sequence 100011011 to reduce the answer to 8 bit number.



Python function galois_mult(a,b) created by chatGPT is short, only 12 lines.
It may be difficult to read, because it has rarely used logical operators like & ,< < and > >.

```
def galois_mult(a, b):
    p = 0
    hi_bit_set = 0
    for i in range(8):
        if b & 1 == 1:
            p ^= a
        hi_bit_set = a & 0x80
        a <<= 1
        if hi_bit_set == 0x80:
            a ^= 0x1b
        b >>= 1
    return p % 256 ;
```

Related to this topic Moodle workspace has one bonus problem , which gives 1 extra point, if answer is correct.
The method can be chosen freely between the polynomial and binary methods described above.

*) Polynomial q(x) with binary coefficients (0 or 1) is in Galois Theory called a Field Irreducible Polynomial, if it cannot be factored in the sense that it cannot be presented as a product q(x) = p(x)*r(x), where p(x) and r(x) would be lower degree polynomials with binary coefficients.