**Aim**

Write C code to generate additive shares of multiplication of two random 32-byte numbers using correlated oblivious transfers.

**Background**

In a secure Multi-Party Computation (MPC) for ECDSA signatures, there is a need for converting multiplicative shares to additive shares using some mechanism. The mechanism is termed as the Multiplicative-to-Additive (MTA) protocol and it can be realized using the Oblivious Transfer (OT) technique. The OT in its basic form, is used for exchanging information between 2 parties without revealing the knowledge of what data was exchanged, but can be extended to produce a MtA protocol.

Learn about OT, and Co-related OT (COT) and their implementation using the Elliptic Curve Cryptography (ECC). Do an implementation of COT to realize a MtA protocol.

**Implementation requirements**

The implementation should accept two integers a and b as input (i.e., multiplicative shares) and produce two integers c and d as output (i.e., additive shares) which when added would produce the product of input integers (in other words a \* b = c + d under the Finite Field of the chosen curve). The integers should be sampled from the Finite Field of the chosen Elliptic Curve (preferably use secp256k1 curve for the implementation).

Additionally, the implementation should also perform the check that the values generated by the implemented program hold the relation as mentioned above (i.e. a \*b= c + d under the Finite Field of the chosen curve). Use XOR for encryption and sha256 for hashing.

Use relevant data structures to hold and process all the integers in the Finite Field of the chosen Elliptic Curve. Refer to Appendix A.3.3 for implementation from Cypherock whitepaper. Use Trezor's ECDSA library for point- multiplication and point-addition functionality wherever required. Assume the relevant input format to the program and output all necessary information to standard output (stdout).

Provide the source code for the implemented program, test inputs to the program, and corresponding output to the provided input.