Assessment 03 - Stream ciphers

Instructions
The assessment must be developed individually and submitted via Blackboard by **17/10/2025**. The submission should be a *zip* file containing all the developed code and a short report in *PDF* format describing the developed code and discussing the achieved results.

Activities

Q1: For the items in *Q1*, take as a basis the `encrypt` function of the *One-Time Pad* (OTP) below.

```python
def encrypt(cleartext: str, keystream: bytes) -> bytes:
    text = cleartext.encode()
    result = bytes([x ^ y for x, y in zip(text,keystream)])
    return result
```

Q1.a: Write the corresponding `decrypt(ciphertext, keystream)` function.
Q1.b: One of the main challenges for the practical use of the OTP is the generation of unique random *keystreams* of the same length as the message whose confidentiality is to be protected. Use the function `os.urandom()` to create the `pseudo-OTP` version, which uses a pseudorandom generator for the *keystream*. For further information regarding `os.urandom()`, consult this reference.
Q1.c: Item *Q1.b* addresses the challenge of generating secure unique *keystreams*; however, it is still necessary for both communicating parties to know the same key. Explore `Key Derivation Functions` (see an initial discussion here) as a way of addressing this issue. What new challenges arise?

Q2: Assume that you have managed to intercept the traffic of an industrial control system that receives encrypted commands (i.e., START, STOP, REMOVE, TRANSFER) using a *one-time pad scheme*. You know that only three *keystreams* are randomly used to encrypt all the commands. Propose a strategy of cryptanalysis that allows discovering the plaintext of each command in this capture file. Note that the chipertexts in this file are represented in hexadecimal.

Q3: The list below consists of six ciphertexts produced using the same keystream of the OTP. Knowing that this cipher is vulnerable when used in this way (see more here), reveal the cleartext corresponding to *Ciphertext 6*.
Note that for this question, it is easiest to use a combination of automated analysis plus human insight and even occasional guessing. As long as you can decrypt them, it doesn't matter how you do it.
Obs. The cleartexts are in Portuguese, with removed special characters and spaces. Also, consider only upper-case text encoded with *UTF-8*.

```
Ciphertext 1:
0x557c7d787a11694469521d0d82cd13fb86a97d8a5b73c5869d126e91e558d413561ab0a3095a892fe763bcf744a0c331948779a
0
Ciphertext 2:
0x576f7d78700c635774520e0793cc19e081a37e8a5164c090990f7896e342d31a4d16a7b505448e35f078aaf643adc0339c8679b
e4384693854070fba72d6deae347df382
Ciphertext 3:
0x516a606d7f0a67407f5a090393ca04e682a2649b4767c5879d027289e64ac6195616a2b100439832e26fa0fd44afda3f989468b
a4f817f2f55150faf66d7d6b72b78ff852320
Ciphertext 4:
0x576f7d78700c635774520e0793cc19e082a8659d5574c59a950f7e88f158db0a4513b4a119569129e76dabfd52a9c7239c817da
```

```
745966d2f4a1413a77cd2daa83469f38338260dfcf580df1fe20672d7e8bde19670f7a7b9c66d86072f084554b61455c933766ba2
1bde22d5cd27723cc1
Ciphertext 5:
0x51626c78701b6b577a5a0a1781d312e083a9749b5372ca908e0e7881e95bdd184105b0a218589921f06dbcf542a0de3387906fb
65e817c2b480f1bab60
Ciphertext 6:
0x5f6c657c6a117c4a6857080685cc13ef91a37c885d7ac19b880e6e91f75fd7125016a7b500
```