Challenge 1.

I have used a small script in Python to encrypt a cleartext. The script is below.

This file encrypts a cleartext into a cyphertext by using a password. In particular, I have used a simple and unsafe password: it is a combination of my own name “stefano” in lowercase combined with a number and an uppercase character (any arbitrary placement of these two extra characters is possible).
The challenge for you is to decrypt the cyphertext. Once decrypted, the cleartext will tell you what to write in the challenge in Moodle to prove that you solved it.
The script combines the password with a salt. In this case the salt is:

- Salt = b'\xd4\x1f\xceg\xe9\xafW\xad\xb7+Y\xc3\xd9t\xe1\xc6'

the encrypted token is:

- Cyphertext =
  b'gAAAAABgoqMJ17XcgGFW347sJ9q1cXjzd1Cl74v42sZVhmbGGer1_l1NFfZS
  M-FRCVpCaZ9-
  JYjy5Ut0Ycy4E1GHyUxCSEgROSw2HFsJjX43qZgk2AyMG1Vzfxx8V212x3WWws
  zfCV1rR2KWHvUyorQB-0asgI3NLcrZiLVjJSQHg2qOqqKNUyv-TQsR-EIo-
  GgI4FOnA1kyFymTQv2Vcjxq4zAtUO3-
  nssuxuVC_n27xefX4eRd_GrnonCvRL_0b_3KYt-pQp4iT_hcbvuEnuM--Ue-
  F_BjYg=='

Hint: I have used the package cryptography-3.4.7-cp36-abi3-win_amd64.whl. The package (or the right one for your configuration) can be downloaded from https://pypi.org/

```
import base64
import os
from cryptography.fernet import Fernet
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC

passwd = b"…………..."
cleartext = b"………………………"
salt = os.urandom(16)
print('salt = ',salt)
kdf = PBKDF2HMAC(
     algorithm=hashes.SHA256(),
     length=32,
     salt=salt,
     iterations=100000,
)
key = base64.urlsafe_b64encode(kdf.derive(passwd))
f = Fernet(key)
cyphertext = f.encrypt(cleartext)
print(‘cyphertext = ', cyphertext)
print(f.decrypt(cyphertext))
```