Practice 8

Islambek Madizhan
p42isdam@uco.es

Exercise 1)
To complete this task we need to use the needed library(Cryptodome) by using this command: **pip install pycryptodome**
What Happens During Execution:
Input: The script uses the hardcoded string b"secret word" as plaintext.
Process:

- ☐ Generates a random key and IV.
- ☐ Encrypts the plaintext using AES in CBC mode.
- ☐ Encodes the IV and ciphertext in Base64.

Saves the results to a JSON file.
Output:

- ☐ Prints the encrypted data and the key.
- ☐ Creates a data.json file containing the IV and ciphertext.

The output:

```
{'iv': 'jCnr/ya29s6UH1zubYqZ1g==', 'ciphertext': '7HYJRpWZEkdsHMI7tnkMtg=='}
key = b07548cd670c8de59cced86770d99b31
bye.
```

Exercise 2)
During the execution of first script, we created ‘data.json’ file that looks like this:

```
Práctica 8 Encriptacón Moderna > {} data.json > ...
  1   {"iv": "jCnr/ya29s6UH1zubYqZ1g==", "ciphertext": "7HYJRpWZEkdsHMI7tnkMtg=="}
```

Now, this is what our script does:
Input the Key:

- ☐ The user enters the key in hexadecimal format (output from the encryption script).
- ☐

Load the Encrypted Data:

- ☐ Reads data.json to retrieve the IV and ciphertext.
- ☐ Decodes the Base64-encoded IV and ciphertext back into bytes.

Decrypt:

- ☐ Initializes the AES cipher using the provided key and IV.
- ☐ Decrypts the ciphertext.
- ☐ Removes padding using unpad.

Output:

- ☐ Displays the decrypted plaintext message.

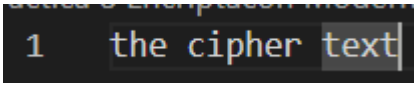
The output:

```
...   {'iv': 'jCnr/ya29s6UH1zubYqZ1g==', 'ciphertext': '7HYJRpWZEkdsHMI7tnkMtg=='}
      The message was:  b'secret word'
```

Exercise 3)
To complete this task, the following code was written:

```
import json
import sys
from base64 import b64encode
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
from Crypto.Random import get_random_bytes

if len(sys.argv) != 2:
    print("Usage: python encFICH.py cipher.txt")
    sys.exit(1)

input_filename = sys.argv[1]
try:
    with open(input_filename, 'rb') as file:
        data = file.read()
except FileNotFoundError:
    print(f"Error: File '{input_filename}' not found.")
    sys.exit(1)

key = get_random_bytes(16)
cipher = AES.new(key, AES.MODE_CBC)

ct_bytes = cipher.encrypt(pad(data, AES.block_size))

iv = b64encode(cipher.iv).decode('utf-8')
ct = b64encode(ct_bytes).decode('utf-8')

result = {
    "iv": iv,
    "ciphertext": ct
}

output_filename = f"{input_filename}.enc.json"
with open(output_filename, 'w') as outfile:
    json.dump(result, outfile)

print(f"Encryption complete. Encrypted file saved as '{output_filename}'.")
print(f"Key (hex): {key.hex()}")
```

Then cipher.txt file was created to encrypt. The content of cipher.txt:



After writing in the command line: **python enFICH.py cipher.txt,** these are the outputs:

```
C:\Users\2022\OneDrive\Рабочий стол\SEGU\Práctica 8 Encriptacón Moderna>python enFICH.py cipher.txt
Encryption complete. Encrypted file saved as 'cipher.txt.enc.json'.
Key (hex): f02b5a11c240d097c603bf8701287387
```

The output **cipher.txt.enc.json file:**

```
Práctica 8 Encriptacón Moderna > {} cipher.txt.enc.json > ...
  1  {"iv": "aDm3HkC0QUAQoTttUfVa1w==", "ciphertext": "0hyDdLvJgrGsI9DGazHH6w=="}
```

Once our data was encrypted and stored as json file, we can begin writing **deFICH.py** file that decrypts the content of encrypted file. It should require a key just as was done in previous exercises. The code:

```
import json
import sys
from base64 import b64decode
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad

if len(sys.argv) != 2:
    print("Usage: python decFICH.py <encrypted_file>")
    sys.exit(1)

input_filename = sys.argv[1]
try:
    with open(input_filename, 'r') as file:
        encrypted_data = json.load(file)
except FileNotFoundError:
    print(f"Error: File '{input_filename}' not found.")
    sys.exit(1)
except json.JSONDecodeError:
    print(f"Error: File '{input_filename}' is not a valid JSON file.")
    sys.exit(1)

key_input = input("Enter the key (hex): ")
try:
    key = bytes.fromhex(key_input)
except ValueError:
    print("Invalid key. Please enter a valid hexadecimal key.")
    sys.exit(1)

try:
    iv = b64decode(encrypted_data['iv'])
    ct = b64decode(encrypted_data['ciphertext'])
except KeyError:
    print("Error: JSON file missing required fields ('iv' or 'ciphertext').")
    sys.exit(1)
```

```
try:
    cipher = AES.new(key, AES.MODE_CBC, iv)
    plaintext = unpad(cipher.decrypt(ct), AES.block_size)
    print("Decryption successful. Decrypted content:")
    print(plaintext.decode('utf-8'))
except (ValueError, KeyError):
    print("Error: Decryption failed. Check the key and input file.")
```

To execute the script we use this command line:
python deFICH.py cipher.txt.enc.json
The output:

```
C:\Users\2022\OneDrive\Рабочий стол\SEGU\Práctica 8 Encriptacón Moderna>python deFICH.py cipher.txt.enc.json
Enter the key (hex): f02b5a11c240d097c603bf8701287387
Decryption successful. Decrypted content:
the cipher text
```

Exercise 4)
Summary of what I got from running the command lines

- ☐ Encrypted and decrypted files (secret.enc, secret_dec.txt).
- ☐ Hashes (MD5, SHA256).
- ☐ RSA private and public keys (private_key.pem, public_key.pem).
- ☐ Encrypted and decrypted content with RSA (secret_rsa.enc, secret_rsa_dec.txt).