Tema 6 – Lab SSI

Ex 1:

```
coefficients = []
states = []
n = int(input())

for i in range(n):
    x = int(input())
    coefficients.append(x)

for j in range(n):
    y = int(input())
    states.append(y)

max_period = pow(2, n) - 1

for i in range(n, max_period + 1):
    x = 0
    for j in range(n):
        print(n - j - 1, i - n + j)
        x += (coefficients[n - j - 1] * states[i - n + j])
    states.append(x % 2)

print(states)
```

Ex 2:

a) b'\x88\x10\x86\xe2\xf3\xaai)\x9fz\xcb\xf0h4\xa4\xec\x88\x10\x86\xe2\xf3\xaai)\x9fz\xcb\xf0h4\xa4\xec\x88\x10\x86\xe2\xf3\xaai)\x9fz\xcb\xf0h4\xa4\xec'

b) c) Este folosit modul de operare ECB. In general, poate fi recomandat datorita simplitatii sale, dar in cazul nostru, avand un pattern ‘test’ repetat, el va cripta identic toate blocurile de ‘test’, fiind foarte usor pentru un atacator sa observe tiparul.

c) Dimensiune cheie: 16; Dimensiune bloc: 48

d) Am folosit modul de operare CCM ce suporta blocuri de orice dimensiune.

Ex 3:

```
from Crypto.Cipher import DES

key1 = '\x10\x00\x00\x00\x00\x00\x00\x00'.encode()
```

```
key2 = '\x20\x00\x00\x00\x00\x00\x00\x00'.encode()

def get_key(x: int):
    return (16 * x).to_bytes(1, byteorder="little") + b'\x00\x00\x00\x00\x00\x00\x00'

cipher1 = DES.new(key1, DES.MODE_ECB)
cipher2 = DES.new(key2, DES.MODE_ECB)

plaintext = "Provocare MitM!!".encode()
crypted = dict()

#Generez dictionarele cheie ghicita: text criptat

for i in range(16):
    key = get_key(i)
    cipher = DES.new(key, DES.MODE_ECB)
    ciphertext = cipher.encrypt(plaintext)
    crypted[ciphertext] = key

encrypted_text = b"G\xfd\xdfpd\xa5\xc9'C\xe2\xf0\x84)\xef\xeb\xf9"

#Le iau pe rand si vad unde se potriveste cu textul cautat
for i in range(16):
    key = get_key(i)
    cipher = DES.new(key, DES.MODE_ECB)
    ciphertext = cipher.decrypt(encrypted_text)
    if ciphertext in crypted:
        print(f"Cheie gasita!!\nKey1 = {crypted[ciphertext]}, key2 = {key}")
```

Am folosit in total 16+16=32 de chei si am facut 2^(nr_de_chei) + 1 criptari/decriptari.