Міністерство освіти і науки України

Національний технічний університет «ХПІ»

Навчально-науковий інститут комп’ютерних наук та інформаційних технологій

Кафедра комп’ютерної інженерії та програмування

# ЗВІТ

з лабораторної роботи № 4

з дисципліни «Сучасні технології безпечного програмування»

**«ВИКОРИСТАННЯ МНЕМОНІЧНИХ ФРАЗ ДЛЯ ФОРМУВАННЯ КЛЮЧІВ ШИФРУВАННЯ»**

Виконав:

студент гр. КН-Н922б

Кулик Д.І.

Перевірив:

Бульба С. С.

Харків – 2022

**Мета роботи:** Дослідити і реалізувати механізм використання мнемонічних фраз для формування ключів шифрування.

**Індивідуальне завдання**

- Використовуючи алгоритм bip39, створити seed генератора псевдовипадкових чисел за допомогою мнемонічної фрази та стосовні ключі шифрування.
- Зашифрувати текст
- Використовуючи раніше створену мнемонічну фразу, відновити ключі шифрування на дешифрувати текст. Вдосконалитись, що оригінальний та дешифрований тексти однакові.

З.І., для додаткових балів необхідно реалізувати підтримку україномовних мнемонічних фраз.

## Хід роботи

BIP описує реалізацію мнемонічного коду або мнемонічного речення — групи слів, які легко запам’ятати — для створення детермінованих гаманців. Він складається з двох частин: створення мнемоніки та перетворення її на бінарне початкове значення. Пізніше це початкове число можна використовувати для створення детермінованих гаманців за допомогою BIP-0032 або подібних методів.

Алгоритм створення ключів можна представити як:

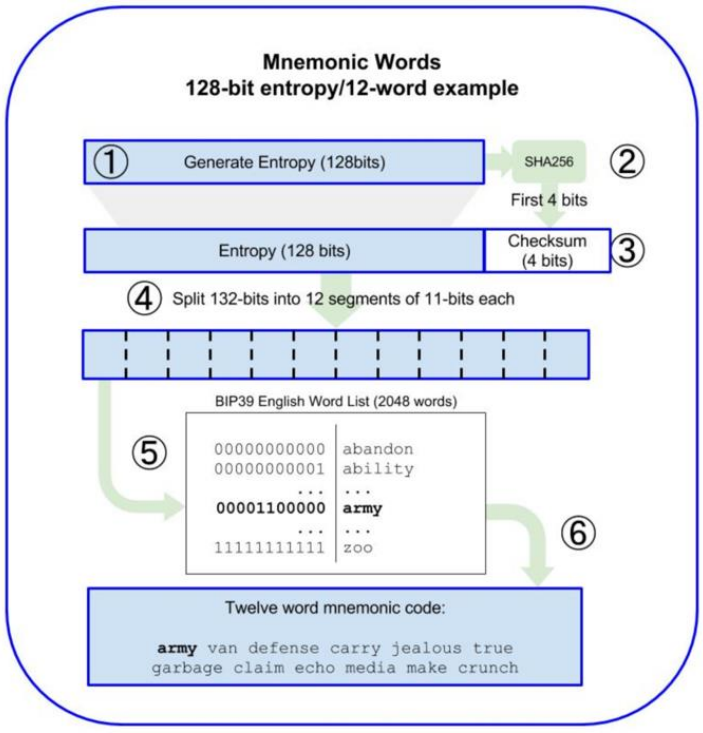


**Рисунок 1 –** Алгоритм створення ключів

**Важливі фрагменти програми**

```python
def get_2048_words(lang):
    """
    Функція повертає словник із парами <індекс, слово>
    де індекси приймають значення від 0 до 2047
    """
    if lang == 'en':
        df = pd.read_csv(
            './src/kulyk04/en_words.txt',
            names=['words']
        ).reset_index().set_index('words')

        nums = df.to_dict()['index']
    elif lang == 'ua':
        df = pd.read_csv(
            './src/kulyk04/ua_words.txt',
            names=['words']
        ).reset_index().set_index('words')

        nums = df.to_dict()['index']

    return nums
```

**Рисунок 1 –** Зчитування слів зі словників

```python
def get_seed(mnemonic: str, passphrase: str = "") -> bytes:
    """
    Функція, що генерує початкове двійкове число за допомогою функції
    pbkdf2_hmac(). Кількість ітерацій встановлено на 2048, а HMAC-SHA512
    використовується як псевдовипадкова функція.
    Довжина отриманого ключа становить 512 біт (= 64 байти).
    """
    decode_phrase(mnemonic)
    mnemonic = normalize_string(mnemonic)
    passphrase = "mnemonic" + normalize_string(passphrase)
    mnemonic_bytes = mnemonic.encode("utf-8")
    passphrase_bytes = passphrase.encode("utf-8")
    stretched = hashlib.pbkdf2_hmac(
        "sha512", mnemonic_bytes, passphrase_bytes, PBKDF2_ROUNDS
    )
    print(f'GENERATED SEED       : {stretched.hex()}')
    return stretched
```

**Рисунок 3 –** Генерація чисел

```python
def encrypt(text, mnemonic):
    rsa = RSA.generate(1024, randfunc=PRNG(get_seed(mnemonic)))
    public_key = rsa.public_key().export_key('PEM')
    print(f'PUBLIC KEY          : {public_key}')
    cipher = PKCS1_OAEP.new(RSA.import_key(public_key))
    encrypted_text = cipher.encrypt(text)

    return encrypted_text
```

**Рисунок 4 –** Функція шифрування

```python
def decrypt(encrypted_text, mnemonic):
    rsa = RSA.generate(1024, randfunc=PRNG(get_seed(mnemonic)))
    private_key = rsa.export_key('PEM')
    print(f'PRIVATE KEY          : {private_key}')
    cipher = PKCS1_OAEP.new(RSA.import_key(private_key))
    decrypted_text = cipher.decrypt(encrypted_text)

    return decrypted_text
```

**Рисунок 5 –** Функція дешифрування

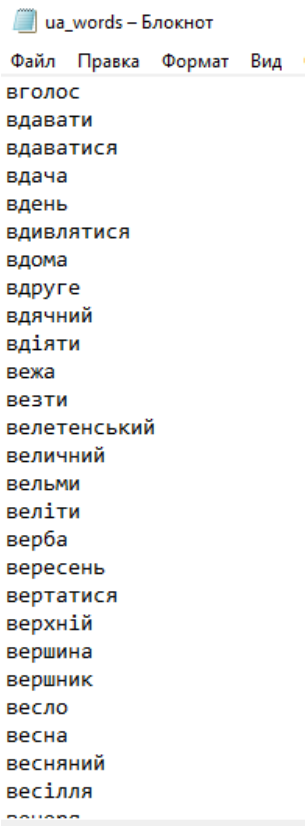


**Рисунок 6 –** Словник українських слів

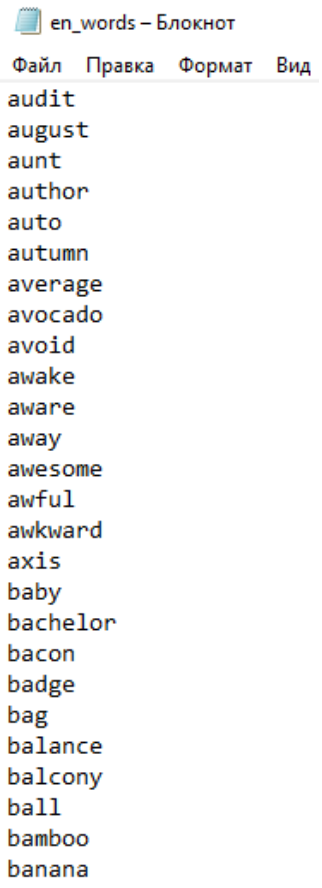


**Рисунок 7 –** Словник англійських слів

## Результати роботи програми

```
(stbp_venv) C:\tmp\stbp>python kulyk04\main.py
ORIGINAL TEXT        : b'Daniil Kulyk'
Mnemonic             : каміння каміння каміння каміння каміння каміння каміння каміння каміння каміння каміння мисливець

------------------------------------------------------------------------------------------------------------------------

----------------------------------------------------ENCRYPTION----------------------------------------------------
------------------------------------------------------------------------------------------------------------------------

GENERATED SEED       : abedc45093e798f8f04e4109b79be098e428f5aebe12b754e144b5a7b607560f7a5139c8689cabc51d6febd33d660ec6d
ec1c8d412214c552fbd9b202b2411ce
PUBLIC KEY           : b'-----BEGIN PUBLIC KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC1UtYNtVlarhw1x8Lx8wVeUk2h\ne
aIyhqXBe/1ayR6TlHx6BHlgs8mWCaGmoE15z/URfGnJs+ZG+EXlUjM1O44SzREb\nDfZ9hbpWDnKL9CwMWB5YM+gYQ7+j9I7tz0OGoLOuMXPDB8iuqppSNg0
dOeWhpLf0\nXnVEyR8z/cxT69mQNQIDAQAB\n-----END PUBLIC KEY-----'
ENCRYPTED TEXT       : 1adeffa601c96ac43353847c72c934907b2b4f3c1daf6a90daf55cb22b48f978a25b5020a081d1a4d60982b58a69d450e
25f706634127d17dcf6bf6ff79199fb808737ec9230f149df39a748aa78336ee3d97a8baa757ba5a45579134afe6e10f28ab3e048582450e686c426f
8fe7b283e1f83020e345f35dfa4ff2c826775fa
------------------------------------------------------------------------------------------------------------------------

----------------------------------------------------DECRYPTION----------------------------------------------------
------------------------------------------------------------------------------------------------------------------------

GENERATED SEED       : abedc45093e798f8f04e4109b79be098e428f5aebe12b754e144b5a7b607560f7a5139c8689cabc51d6febd33d660ec6d
ec1c8d412214c552fbd9b202b2411ce
PRIVATE KEY          : b'-----BEGIN RSA PRIVATE KEY-----\nMIICXAIBAAKBgQC1UtYNtVlarhw1x8Lx8wVeUk2heaIyhqXBe/1ayR6TlHx6BH
lg\ns8mWCaGmoE15z/URfGnJs+ZG+EXlUjM1O44SzREbDfZ9hbpWDnKL9CwMWB5YM+gY\nQ7+j9I7tz0OGoLOuMXPDB8iuqppSNg0dOeWhpLf0XnVEyR8z/c
xT69mQNQIDAQAB\nAoGAGhM5/gmsuloca215LtMMXBaI3GZB+XMchhj/A521fHKXffKDw/DG1gE3kl8u\nFQzfSK70BZIN1PniD0rFW2vfj72Yny6UDAaJLR
mEUO2HvqLRTZ9us33djbJVV7Lb\nLPnjduT4MsXj9xoZZqf4mRg7RPh424Vs/1eL2VL9A/snWycCQQDRSJKeElMF/a5b\ncZQUetyvn4Zw7cFhkr9FYkCLbc
kjHWwgaVZagu1In7bBR+v9F4pj2ZIk54KZF+5F\nRZ8ARRADAkEA3cyBO77F4lIJIOiXOyxadzyK7VSF/w/vgwv4raca/5CPR6ZgLelA\nTR4TVLzn381Bwd
8hgC18F3e31ohIc+q1ZwJBANAeL09qQY2vm/frlqpXE6bCKNuc\nU7ZITFUfQ7hj0KVO2PvSRMF1eanCyeajMxZYlpJ9BkwUvVao6ZWRinCmPSkCQGtX\nPr
AQaqpWM27GwfH0TC/qj/296z9+kdalI4u/FLTcXEs7vM5bVTA9PvPCnKFWL93I\nw+jJmoWs/b6IpC/dRj8CQDLZGw16HxWrZSYlMUNOToUjIhF4FT+Cck/n
g03ONXQ6\nqoKL4MWTJ5ZetWfbvneNU9WB5aOa6K25c3Cpds5GTAA=\n-----END RSA PRIVATE KEY-----'
DECRYPTED TEXT       : b'Daniil Kulyk'
------------------------------------------------------------------------------------------------------------------------
------------------------------

(stbp_venv) C:\tmp\stbp>
```

**Рисунок 8 –** Результат виконання програми для ua мнемонічних фраз

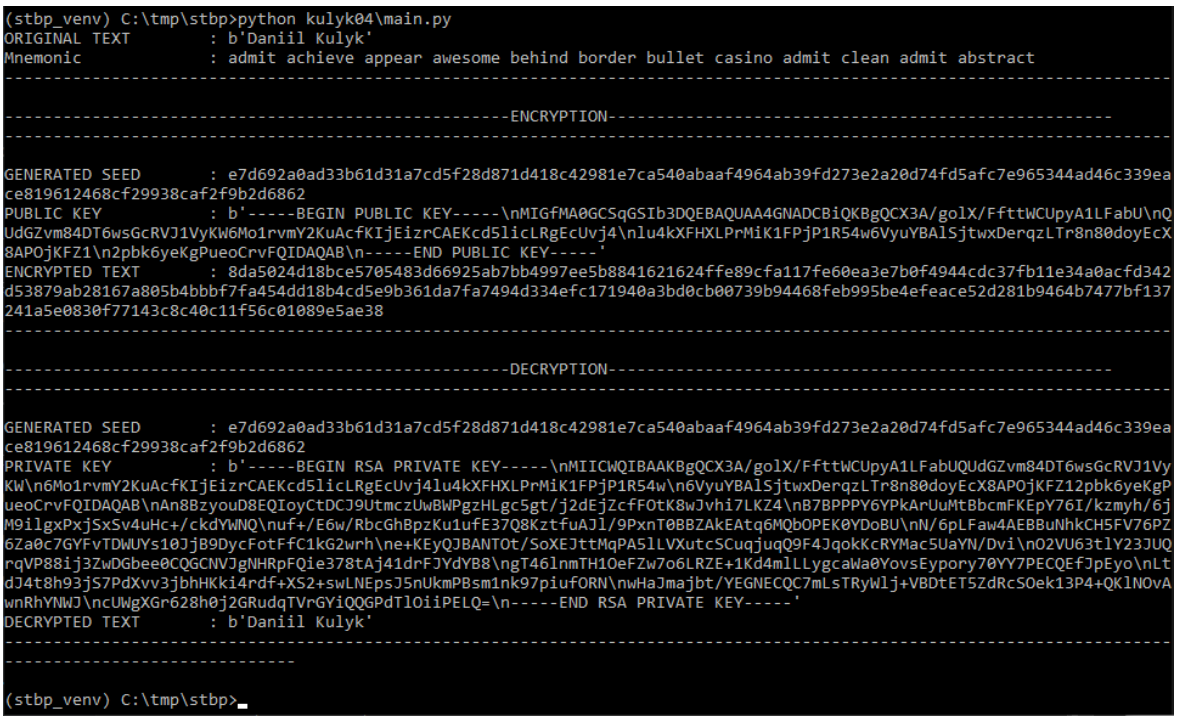


**Рисунок 9 –** Результат виконання програми для en мнемонічних фраз

**Висновки:** в результаті виконання лабораторної роботи було досліджено і реалізовано механізм використання мнемонічних фраз для формування ключів шифрування RSA.