

Міністерство освіти і науки України Львівський
національний університет імені Івана Франка
Факультет прикладної математики та інформатики
Кафедра програмування

Звіт до лабораторної роботи №5 з теми
“RSA шифр”

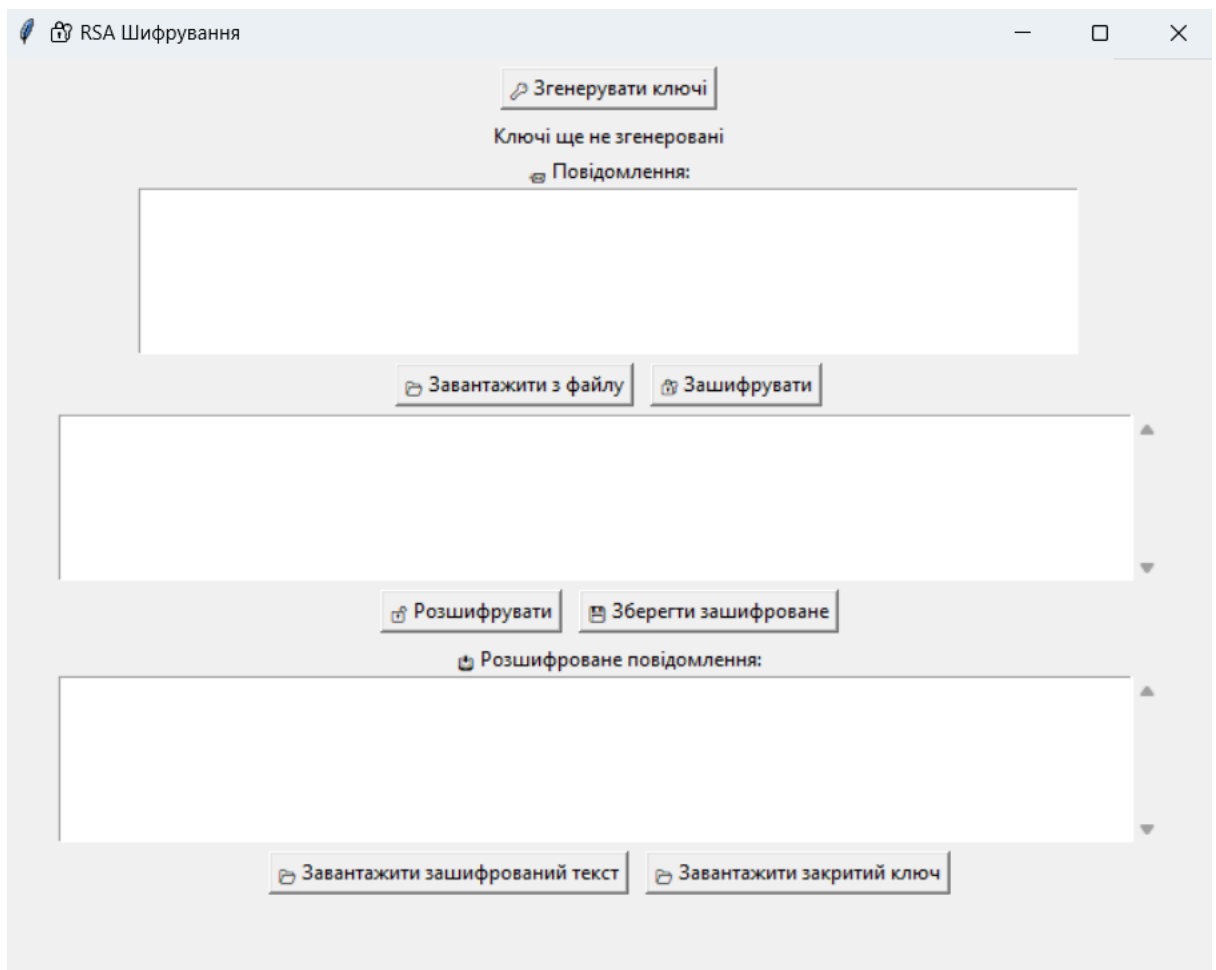
Підготував:
студент групи ПМІ-31
Урбанський Максим

Львів-2025

Мета: реалізувати шифр RSA.

Хід роботи

1. Реалізував інтерфейс за допомогою Python, tkinter.



Ми можемо згенерувати публічний і приватний ключі як і зберегти приватний ключ.

Можемо завантажити текст з файлу так і ввести вручну.

Можемо завантажити і зберегти зашифрований текст.

Підтримується і англійська і українська мови.

2. Розробив клас, де реалізував методи потрібні для роботи програми.

```
ui.py  model.py X
rsa > model.py > RSA
1  import random
2  from math import gcd
3
4  class RSA:
5      def __init__(self):
6          self.public_key = None
7          self.private_key = None
8
9      @staticmethod
10     def mod_pow(base, exponent, modulus):
11         result = 1
12         base %= modulus
13         while exponent > 0:
14             if exponent % 2 == 1:
15                 result = (result * base) % modulus
16                 base = (base * base) % modulus
17                 exponent //= 2
18         return result
19
20     @staticmethod
21     def mod_inverse(e, phi):
22         def egcd(a, b):
23             if a == 0:
24                 return b, 0, 1
25             g, y, x = egcd(b % a, a)
26             return g, x - (b // a) * y, y
27
28         g, x, _ = egcd(e, phi)
29         if g != 1:
30             raise Exception('Оберненого елемента не існує')
31         return x % phi
32
33     @staticmethod
34     def is_prime(n):
35         if n <= 1: return False
36         if n <= 3: return True
37         if n % 2 == 0 or n % 3 == 0: return False
```

3. Написав тести перевірити функціонал.

```
4 class TestRSA(unittest.TestCase):
5     def setUp(self):
6         self.rsa = RSA()
7         self.public_key, self.private_key, self.p, self.q = self.rsa.generate_keys()
8
9     def test_keys_are_not_none(self):
10        self.assertIsNotNone(self.public_key)
11        self.assertIsNotNone(self.private_key)
12
13    def test_keys_are_distinct(self):
14        self.assertNotEqual(self.p, self.q)
15
16    def test_mod_pow(self):
17        base, exp, mod = 4, 13, 497
18        expected = pow(base, exp, mod)
19        result = self.rsa.mod_pow(base, exp, mod)
20        self.assertEqual(result, expected)
21
22    def test_mod_inverse(self):
23        e = 17
24        phi = 3120
25        d = self.rsa.mod_inverse(e, phi)
26        self.assertEqual((e * d) % phi, 1)
27
28    def test_is_prime_true(self):
29        self.assertTrue(self.rsa.is_prime(101))
30
31    def test_is_prime_false(self):
32        self.assertFalse(self.rsa.is_prime(100))
33
34    def test_encrypt_decrypt(self):
35        message = "Привіт, RSA!"
36        encrypted = self.rsa.encrypt(message, self.public_key)
37        self.assertIsInstance(encrypted, list)
38        decrypted = self.rsa.decrypt(encrypted, self.private_key)
39        self.assertEqual(decrypted, message)
```

Висновок: на цій практичній роботі я навчився реалізовувати шифрування за допомогою RSA.