**КРИПТОГРАФІЯ**

**КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4**

**Вивчення криптосистеми RSA та алгоритму електронного підпису; ознайомлення з методами генерації параметрів для асиметричних криптосистем**

**Виконали:** ФБ-11 Мельниченко Богдан, Захаренко Нікіта

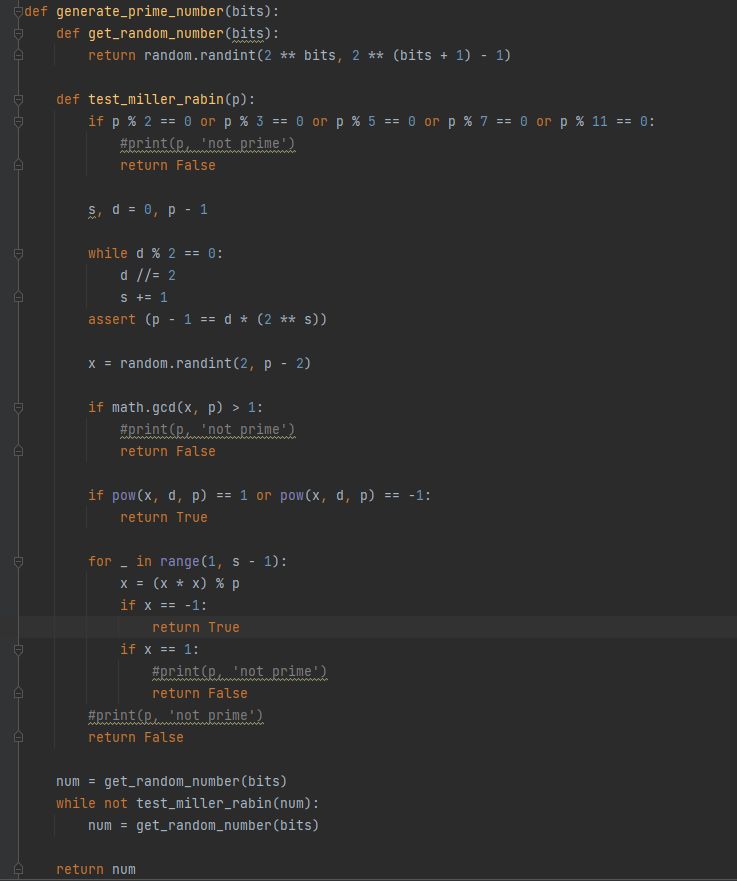
**Варіант:** 8

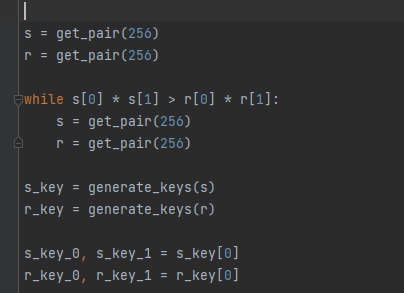
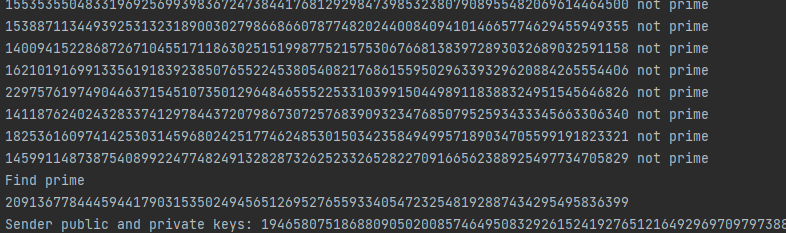
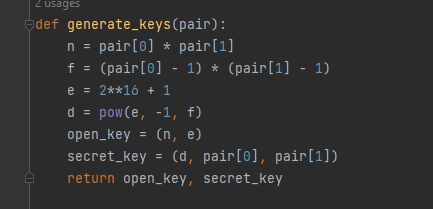
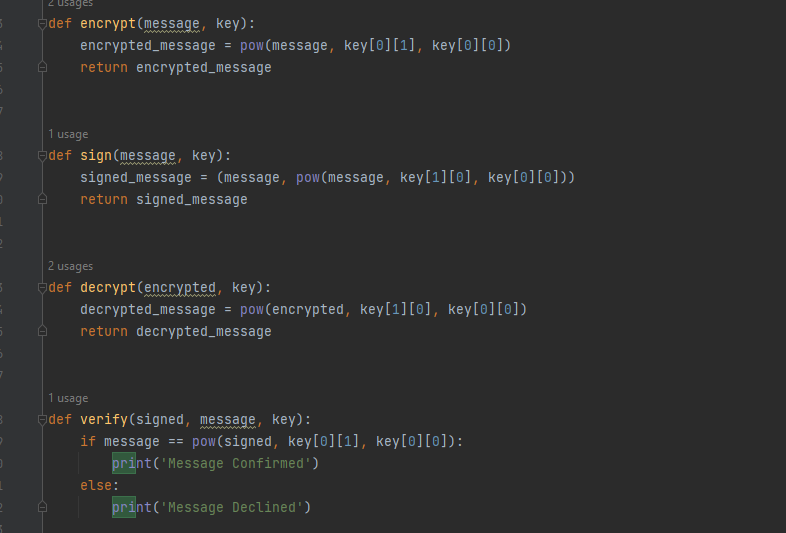
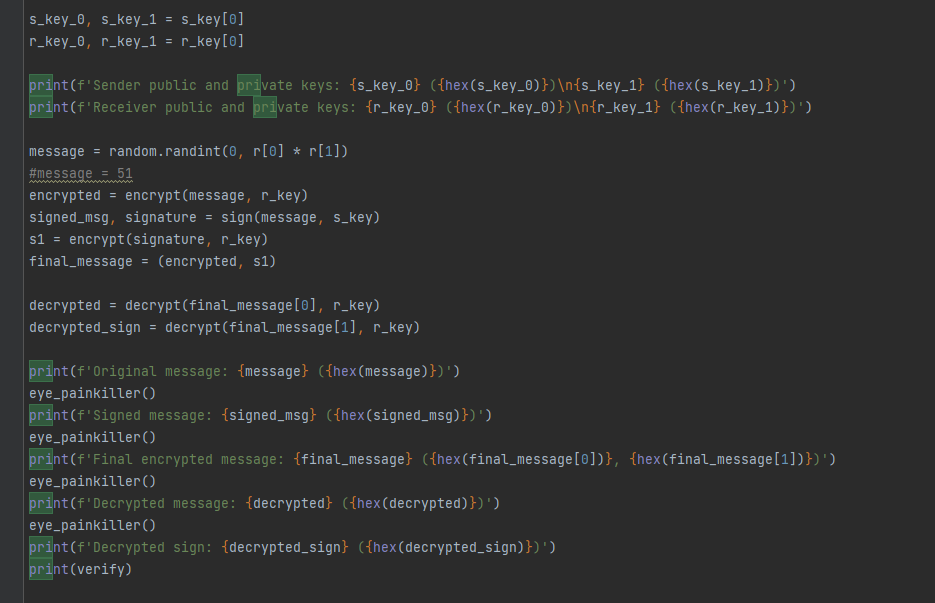
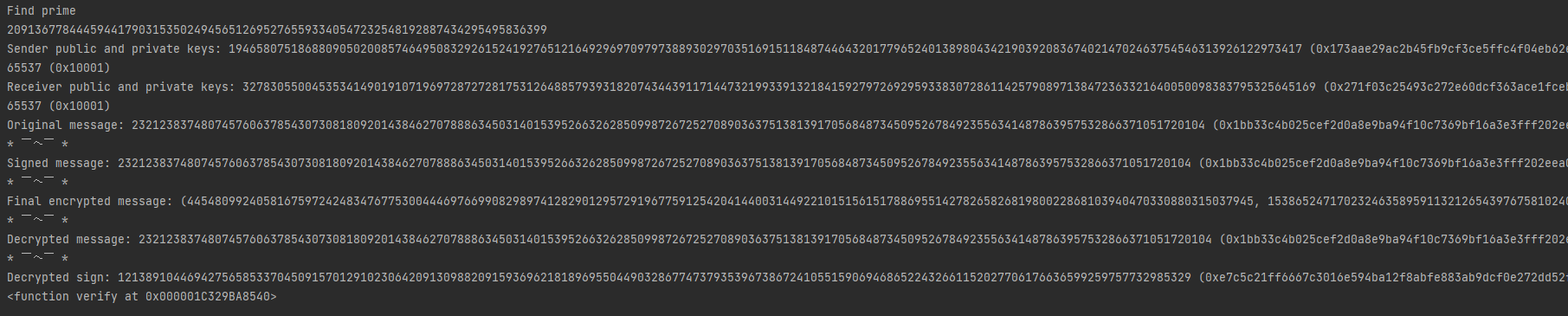
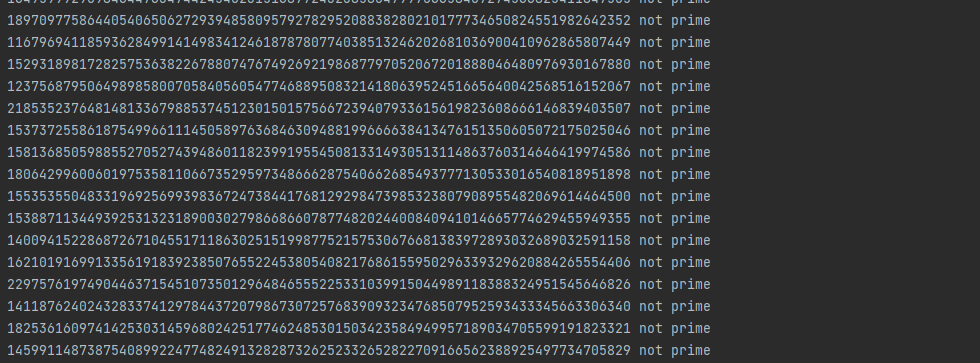
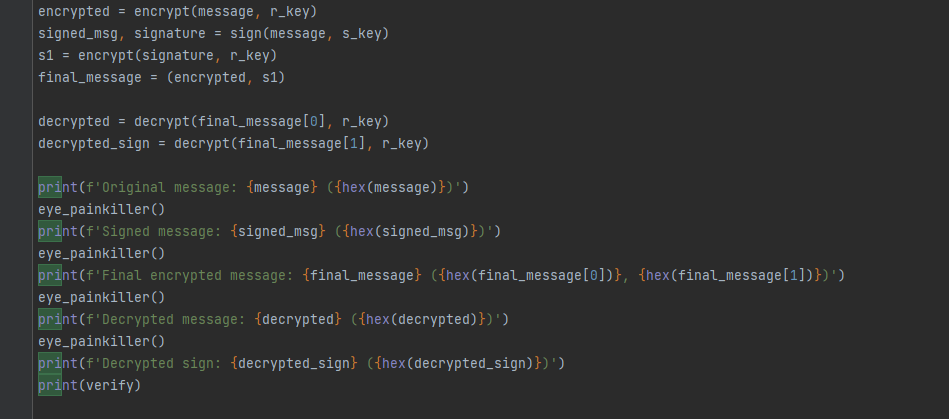
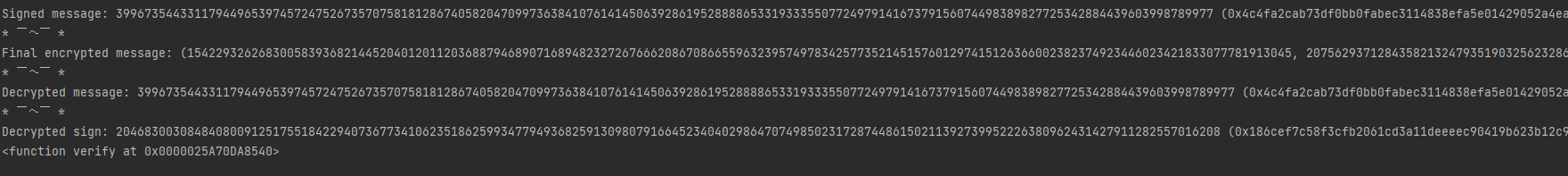
**Мета роботи:**

Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів

**Порядок виконання роботи**:

1. Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч, використання готових реалізацій тестів не дозволяється.
2. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p1 , q1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq < p1q1; p і q – прості числа для побудови ключів абонента А, p1 і q1 – абонента B.
3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p, q) та відкритий ключ (n, e). За допомогою цієї функції побудувати схеми RSA для абонентів А і B – тобто, створити та зберегти для подальшого використання відкриті ключі (e, n), (e1, n1) та секретні d і d1.
4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і B. Кожна з операцій (шифрування розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті ключові дані, які необхідні для її виконання. За допомогою датчика випадкових чисел вибрати відкрите повідомлення M і знайти криптограму для абонентів А и B, перевірити правильність розшифрування. Скласти для А і B повідомлення з цифровим підписом і перевірити його.
5. За допомогою раніше написаних на попередніх етапах програм організувати роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA. Протоколи роботи кожного учасника (відправника та приймаючого) повинні бути реалізовані у вигляді окремих процедур, на вхід до яких повинні подаватись лише ті ключові дані, які необхідні для виконання. Перевірити роботу програм для випадково обраного ключа 0 < k < n.



* Функція get\_random\_number(bits): Генерує випадкове число у заданому діапазоні біт. Число вибирається між 2^bits і 2^bits+1, включаючи крайні точки.
* Функція test\_miller\_rabin(p): Перевіряє, чи є число p простим, використовуючи тест Міллера-Рабіна. Спочатку вона виключає числа, кратні деяким малим простим числам (2, 3, 5, 7, 11). Після цього виконується тест Міллера-Рабіна, який є ефективним імовірнісним тестом для визначення простоти числа.
* Головна частина: Генерує випадкове число num з використанням get\_random\_number(bits) і перевіряє його на простоту за допомогою test\_miller\_rabin(num). Якщо число не є простим, генерація і перевірка повторюються до тих пір, поки не буде знайдено просте число.
* **2**. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p1 , q1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq p1q1 ; p і q – прості числа для побудови ключів абонента А, 1 p і q1 – абонента B.
* 
* Output
* 
* **3.** Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p,q) та відкритий ключ (n,e) . За допомогою цієї функції побудувати схеми RSA для абонентів А і B – тобто, створити та зберегти для подальшого використання відкриті ключі (e,n) , ( , ) 1 n1 e та секретні d і d1
* 
* **4.** Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і B. Кожна з операцій (шифрування, розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті ключові дані, які необхідні для її виконання.
* 
* **5**. За допомогою раніше написаних на попередніх етапах програм організувати роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA. Протоколи роботи кожного учасника (відправника та приймаючого) повинні бути реалізовані у вигляді окремих процедур, на вхід до яких повинні подаватись лише ті ключові дані, які необхідні для виконання. Перевірити роботу програм для випадково обраного ключа 0 k n.
* 
* 
* Sender public and private keys: 19465807518688090502008574649508329261524192765121649296970979738893029703516915118487446432017796524013898043421903920836740214702463754546313926122973417 (0x173aae29ac2b45fb9cf3ce5ffc4f04eb62e3f02699c65bde7d5d27e6db29a9bf88c3147dc8659bd3e54bff69a09959df246e15c7d1d24931d36250af22450a8e9)
* 65537 (0x10001)
* Receiver public and private keys: 32783055004535341490191071969728727281753126488579393182074344391171447321993391321841592797269295933830728611425790897138472363321640050098383795325645169 (0x271f03c25493c272e60dcf363ace1fceb18dea486a278307ab6ffda6925051877b2e2b72f83aac9fbe82a91971b370aad87efbb4c04829a72e192d6fb7e923171)
* 65537 (0x10001)
* Original message: 23212383748074576063785430730818092014384627078886345031401539526632628509987267252708903637513813917056848734509526784923556341487863957532866371051720104 (0x1bb33c4b025cef2d0a8e9ba94f10c7369bf16a3e3fff202eea0ae427120970fb5d6d21466a615cd42dd977b5569eb29b1fb9964676ac3cb8b3c2b79c5b47e0da8)
* \* ￣へ￣ \*
* Signed message: 23212383748074576063785430730818092014384627078886345031401539526632628509987267252708903637513813917056848734509526784923556341487863957532866371051720104 (0x1bb33c4b025cef2d0a8e9ba94f10c7369bf16a3e3fff202eea0ae427120970fb5d6d21466a615cd42dd977b5569eb29b1fb9964676ac3cb8b3c2b79c5b47e0da8)
* \* ￣へ￣ \*
* Final encrypted message: (4454809924058167597242483476775300444697669908298974128290129572919677591254204144003144922101515615178869551427826582681980022868103940470330880315037945, 15386524717023246358959113212654397675810240989088856234752446455670802644304183981481938247851748786444107964243548506519286527375375140621184371945655812) (0x550ea84697fe40f17672b8b31ee81d6bcb85a87ffb77a13107134104e6bb3400d2763ec2069b32bbcc9b05478f761bca8e4d49a96b43ec91be6dea61a1a318f9, 0x125c7c424bb27666132e75d6747d60de986d8c5b18e04e2eedc99eac3313c617f1aeaf0cdf7168011985edc03325581b97275a965c21a57d2185e610fbbe84e04)
* \* ￣へ￣ \*
* Decrypted message: 23212383748074576063785430730818092014384627078886345031401539526632628509987267252708903637513813917056848734509526784923556341487863957532866371051720104 (0x1bb33c4b025cef2d0a8e9ba94f10c7369bf16a3e3fff202eea0ae427120970fb5d6d21466a615cd42dd977b5569eb29b1fb9964676ac3cb8b3c2b79c5b47e0da8)
* \* ￣へ￣ \*
* Decrypted sign: 12138910446942756585337045091570129102306420913098820915936962181896955044903286774737935396738672410551590694686522432661152027706176636599259757732985329 (0xe7c5c21ff6667c3016e594ba12f8abfe883ab9dcf0e272dd52f7eac19e5e91fcae4ff89e3ccbb8fe19b94dc00833c3c9edbfea127f5594f4c7dde39a804c79f1)
* <function verify at 0x000001C329BA8540>
* Декілька кандидатів, які не пройшли перевірку:
* 
* Опис кроків протоколу конфіденційного розсилання ключів з підтвердженням справжності: 0 < message < n
* 
* Шифрування та підпис повідомлення:
* 
* Шифруємо підпис та отримуємо фінальне повідомлення: (k1, s1). Отримувач за допомогою секретного ключа розшифровує повідомлення і підпис та проводить автентифікацію.
* 
* 
* \* ￣へ￣ \*
* Signed message: 3996735443311794496539745724752673570758181286740582047099736384107614145063928619528888653319333550772497914167379156074498389827725342884439603998789977 (0x4c4fa2cab73df0bb0fabec3114838efa5e01429052a4eacc013eb23880eac02c6b757c454c07b1feb5b83e242ac69af1f44c8c4f6839978735115a8b835a9159)
* \* ￣へ￣ \*
* Final encrypted message: (15422932626830058393682144520401201120368879468907168948232726766620867086655963239574978342577352145157601297415126366002382374923446023421833077781913045, 20756293712843582132479351903256232869592047721864497832094281711429730210710859886361566206689359384079340273219105145670684002992324916677307601305180190) (0x12679b96e73af866f8aae05d2cf647d6f5b6cbbcd78b89cf95852c59fbde3482d8e70d1d7df732d4e05a9dd9eed89b06a912e2e5c91cafb47f420f70ed36f5dd5, 0x18c4ea679c845401a85707fe3ac037811f381083ca285d5600817cc87c5140246eac73e1e2dfe6f840d6c7d1e35ec97f04e0b581aafdccbf944cb3ad627c7981e)
* \* ￣へ￣ \*
* Decrypted message: 3996735443311794496539745724752673570758181286740582047099736384107614145063928619528888653319333550772497914167379156074498389827725342884439603998789977 (0x4c4fa2cab73df0bb0fabec3114838efa5e01429052a4eacc013eb23880eac02c6b757c454c07b1feb5b83e242ac69af1f44c8c4f6839978735115a8b835a9159)
* \* ￣へ￣ \*
* Decrypted sign: 20468300308484080091251755184229407367734106235186259934779493682591309807916645234040298647074985023172874486150211392739952226380962431427911282557016208 (0x186cef7c58f3cfb2061cd3a11deeeec90419b623b12c9a10e8c2e65db407408488a785eaba1d0cfc6b9c257f7eeb0e6ca0c8bd62219ecc238346e008c8acd5890)
* <function verify at 0x0000025A70DA8540>

**Висновок**: Отже, під час виконання цієї лабораторної роботи, ми ознайомились із різними методами перевірки простоти чисел та з методами створення ключів для асиметричної криптосистеми RSA. Ми практично вивчили систему захисту інформації, яка ґрунтується на криптосхемі RSA. Ми створили засекречений зв'язок та електронний підпис за допомогою цієї системи, а також вивчили протокол розподілу ключів. Ми також реалізували протокол конфіденційного розподілу ключів за допомогою мови програмування **Python3** на відкритих каналах зв'язку, з підтвердженням автентичності відправника.