Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

# КРИПТОГРАФІЯ

# КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

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

Виконав: ФБ-11 Тимощук Ілья

Київ – 2023

## Мета та основні завдання 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 використовувати вбудований генератор псевдовипадкових чисел мови програмування Python. В якості тесту перевірки на простоту використати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч.

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 A, 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 A і B – тобто, створити та зберегти для подальшого використання відкриті ключі (e, n) , (e , n ) та секретні d і d .

4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів A 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 A и B, перевірити правильність розшифрування. Скласти для A 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

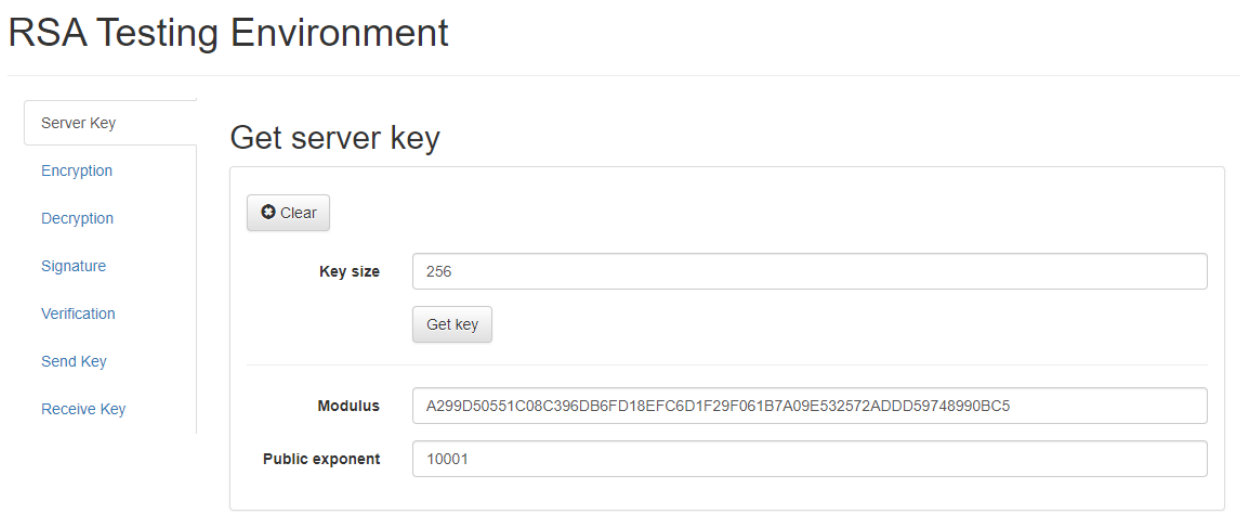
Кожна з наведених операцій повинна бути реалізована у вигляді окремої процедури, інтерфейс якої повинен приймати лише ті дані, які необхідні для її роботи; наприклад, функція Encrypt(), яка шифрує повідомлення для абонента, повинна приймати на вхід повідомлення та відкритий ключ адресата (і тільки його), повертаючи в якості результату шифротекст. Відповідно, програмний код повинен містити сім високорівневих процедур: GenerateKeyPair(), Encrypt(), Decrypt(), Sign(), Verify(), SendKey(), ReceiveKey().

## Хід роботи

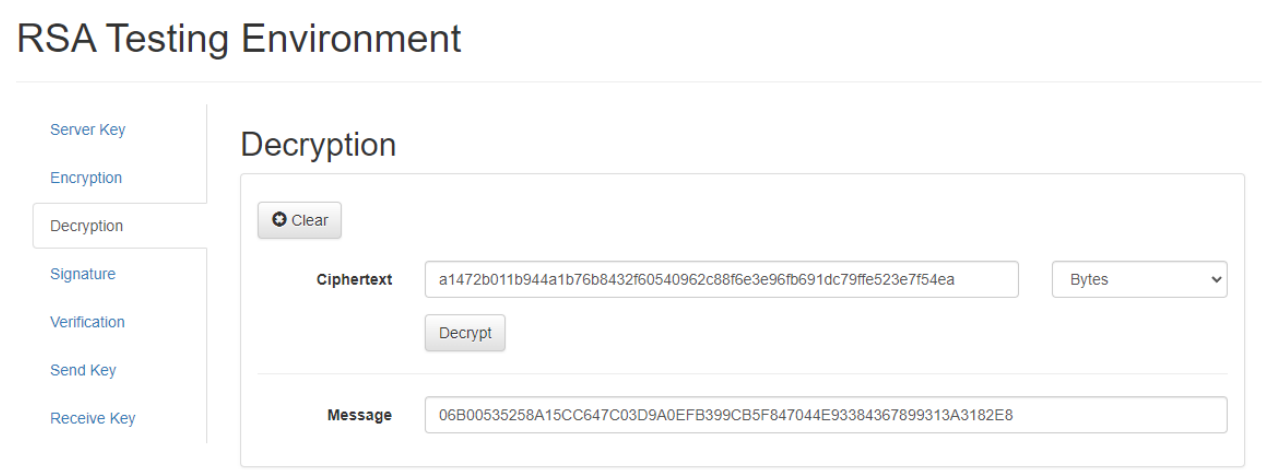
Значення кандидатів, що не пройшли тест перевірки простоти записані в **invalid_prime_numbers.txt**

Вивід значень згенерованих ключів для A та B, а також перевірка функцій encrypt_message(), decrypt_message(), sign_message(), verify_signature(), send_key() та recive_key() для випадково згенерованих M та k. Логи цих та інших значень, виведених в консоль записані в **console_log.txt**


**Генерація публічного ключа сервера:**

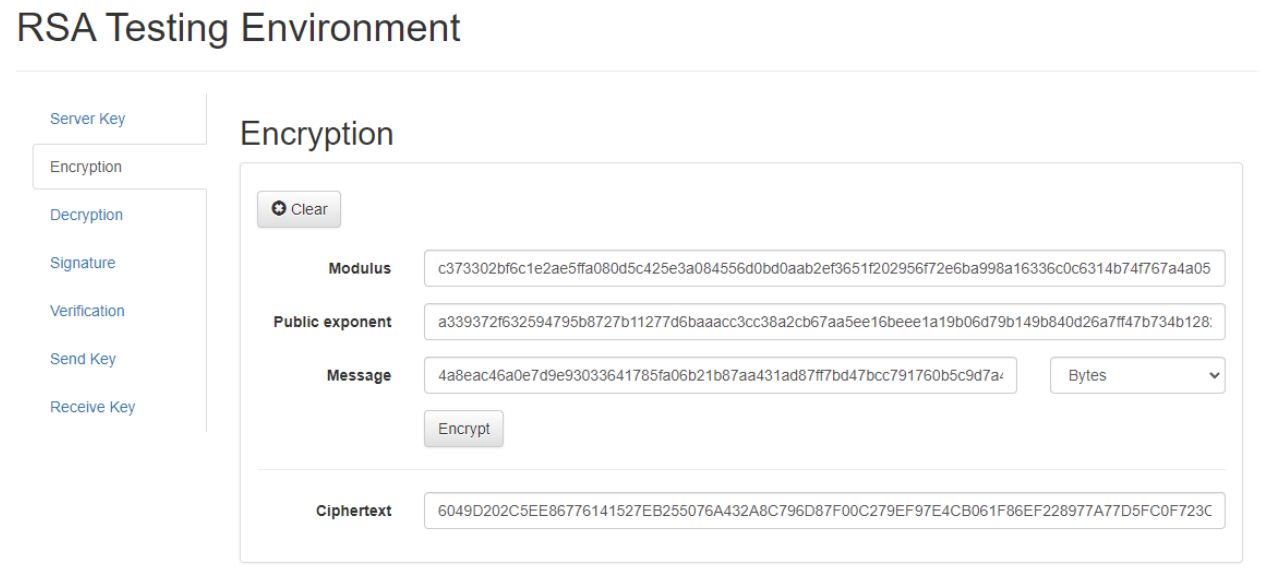


**Test #1 Decryption:**



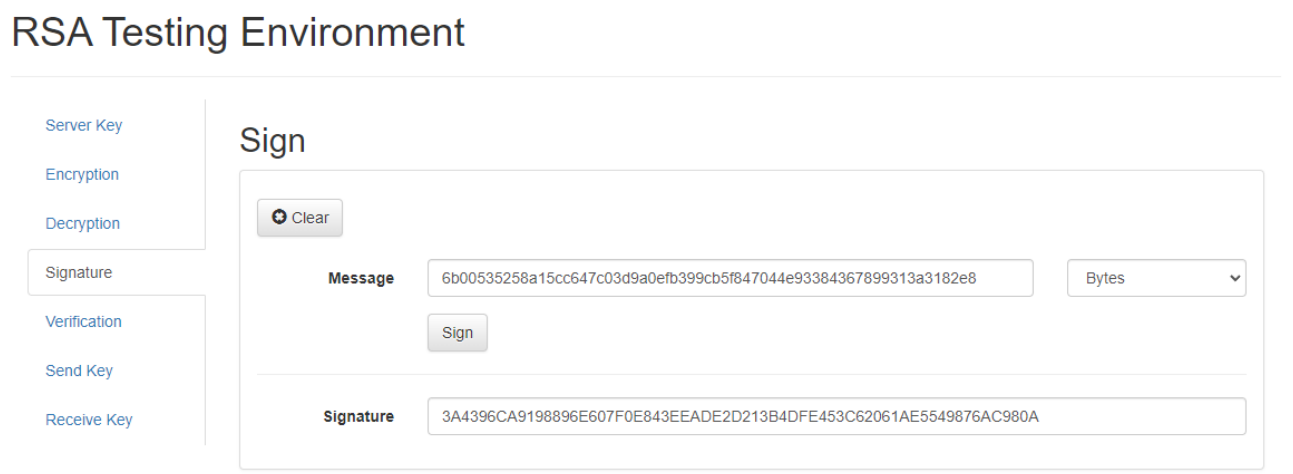

```
Test #1:
server_key_n :  735464792867813526204604252880223300687178632602844149151669471419389984798 13
server_key_e :  65537
test1_3_message :  302487811646286101170520373268399279089533964460938171751202335437602747 2616
server_key_n_hex :  a299d50551c08c396db6fd18efc6d1f29f061b7a09e532572addd59748990bc5
server_key_e_hex :  10001
test1_3_message_hex :  6B00535258A15CC647C03D9A0EFB399CB5F847044E93384367899313A3182E8
test1_3_message_hex (encrypted) :  a1472b011b944a1b76b8432f60540962c88f6e3e96fb691dc79ffe523e7f54ea
                                        TEST #1 COMPLETED!
```

**Test #2 Encryption:**




```
Test #2:
n_test2_4_5 :  102365446889206309087336591272010535239526667957028495690647385723945068573592385372734732746216755842100731777300920826238789823529803051064399364662015 19
e_test2_4_5 :  854870826158456710209153639691827226564930676983376415934814368662706071674189619090907108092520250924181459601568164506738466597965519087000092921396627 9
d_test2_4_5 :  318693226333418535651189478929294389333850536881554094920090930524706901059653540961380457755928320151895849962742541550842429532851011351833978899469479
n_test2_4_5_hex :  c373302bf6c1e2ae5ffa080d5c425e3a084556d0bd0aab2ef3651f202956f72e6ba998a16336c0c6314b74f767a4a052dca20ab0653be48cf2bd144e659fabaf
e_test2_4_5_hex :  a339372f632594795b8727b11277d6baaacc3cc38a2cb67aa5ee16beee1a19b06d79b149b840d26a7ff47b734b128224a321f14f1be5e68c44264f9a86a6e7c7
test2_4_message :  39048834987108131375727401698727842116996855376697845008745173530887650087850756997323066476662675123087095971106567145725613685178375968414634695024023 09
test2_4_message_hex :  4a8eac46a0e7d9e93033641785fa06b21b87aa431ad87ff7bd47bcc791760b5c9d7a49d42917c084b0554a698503a3976ce0314feee06ef7cdf85b85b2188f05
server_ciphertext :  5043030652236207148779660098229269277511583771056022097196145805912658988806428169599355724797035843619047211772934190364289734682358449728330700259339587
server_ciphertext_hex :  6049D202C5EE86776141527EB255076A432A8C796D87F00C279EF97E4CB061F86EF228977A77D5FC0F723C0733A96C413F332C69EBB61ABF670EBCF2001C2D43
Decrypted message from server:  39048834987108131375727401698727842116996855376697845008745173530887650087850756997323066476662675123087095971106567145725613685178375968414634695024023 09
                                        TEST #2 COMPLETED!
```

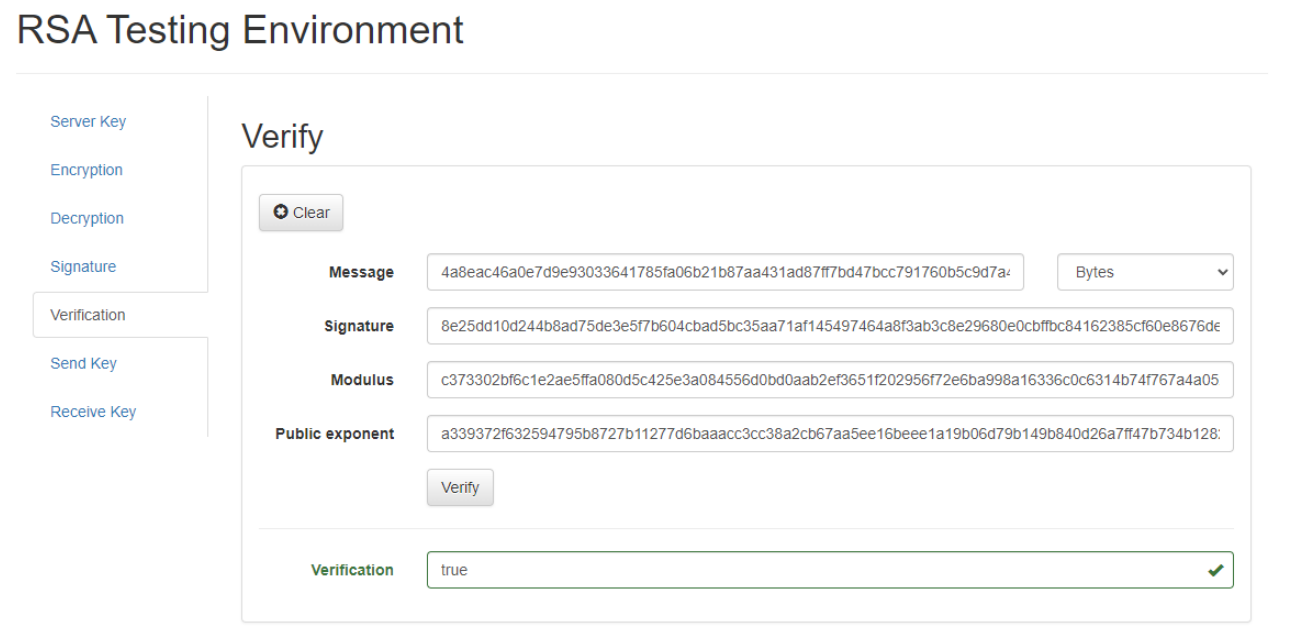
**Test #3 Sign:**




```
Test #3:
server_test3_signature:  263535646943638799332105710001245626665018398484225953269225526825440182661 22
server_test3_signature_hex:  3a4396ca9198896e607f0e843eeade2d213b4dfe453c62061ae5549876ac980a
Перевірка цифрового підпису вірна.
                                        TEST #3 COMPLETED!
```

Для тесту №3 було взято значення повідомлення test1_3_message з тесту №1.

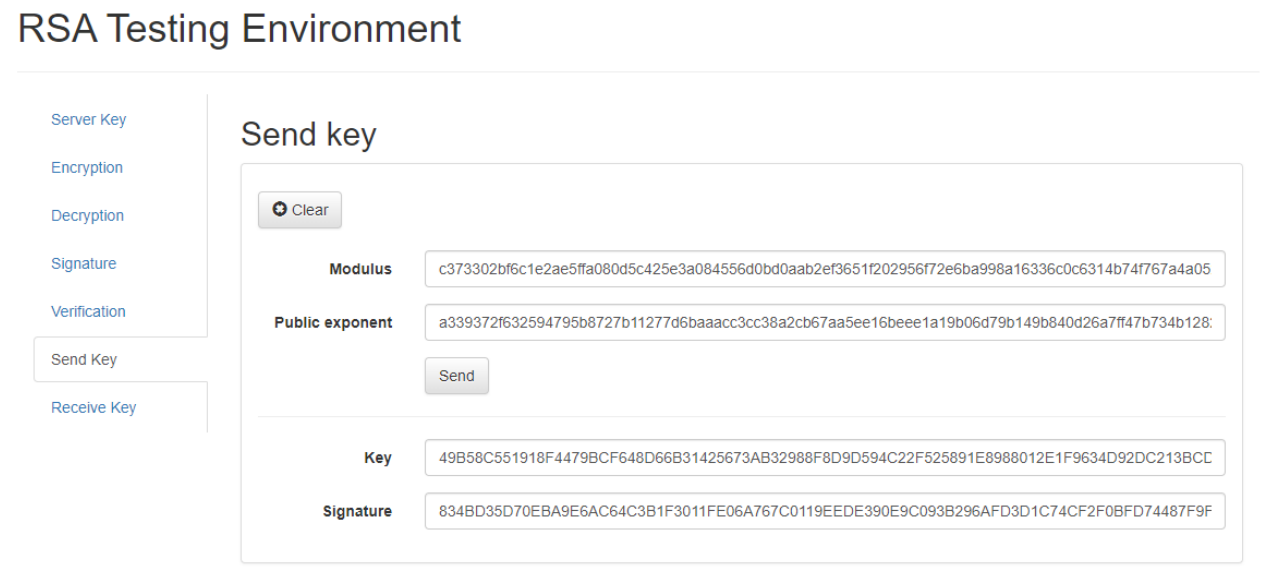
**Test #4 Verify:**




```
Test #4:
Цифровий підпис: 744488984547883647912604929274881504760119068551992179755685741601646761613428172963544600524046675963914417551874915132772733563313376271261507159065862 0
signed_test2_4_message:  744488984547883647912604929274881504760119068551992179755685741601646761613428172963544600524046675963914417551874915132772733563313376271261507159065862 0
signed_test2_4_message_hex:  8e25dd10d244b8ad75de3e5f7b604cbad5bc35aa71af145497464a8f3ab3c8e29680e0cbffbc84162385cf60e8676de3c5baacff691a244ddad5e6672588323c
                                        TEST #4 COMPLETED!
```

Для тесту №4 було взято значення повідомлення test2_4_message, d_test2_4_5 та n_test2_4_5 з тесту №2.

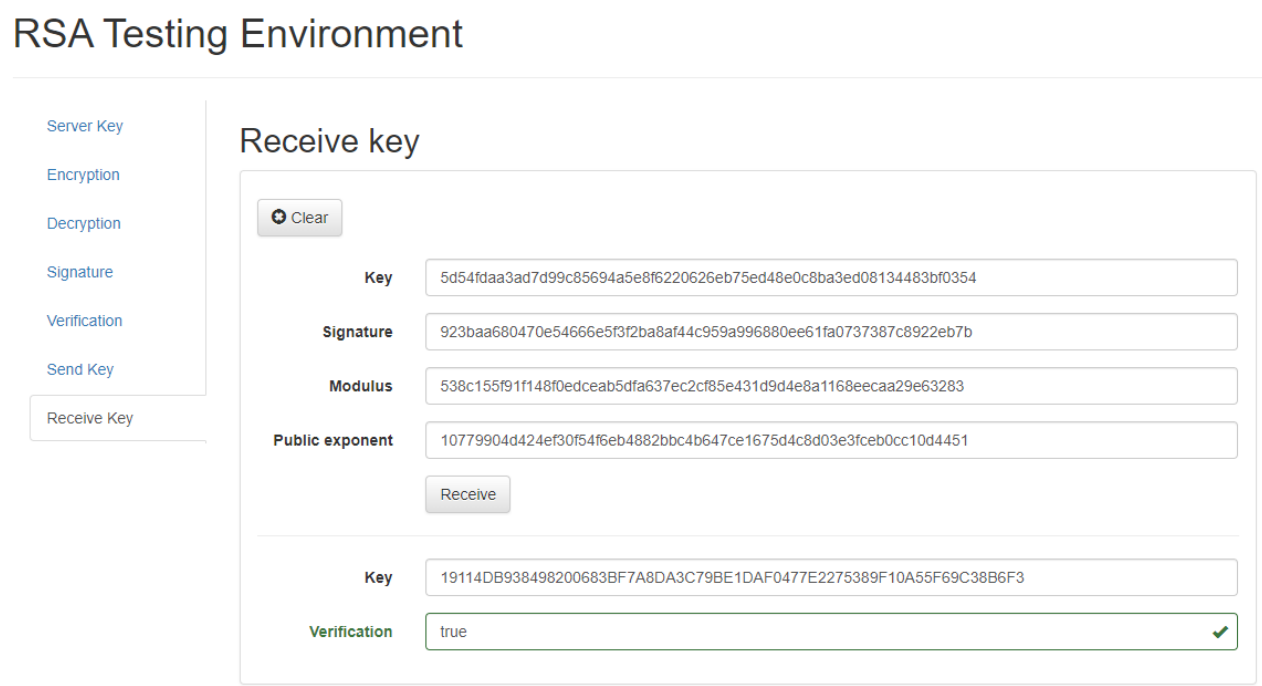
**Test #5 Send key:**

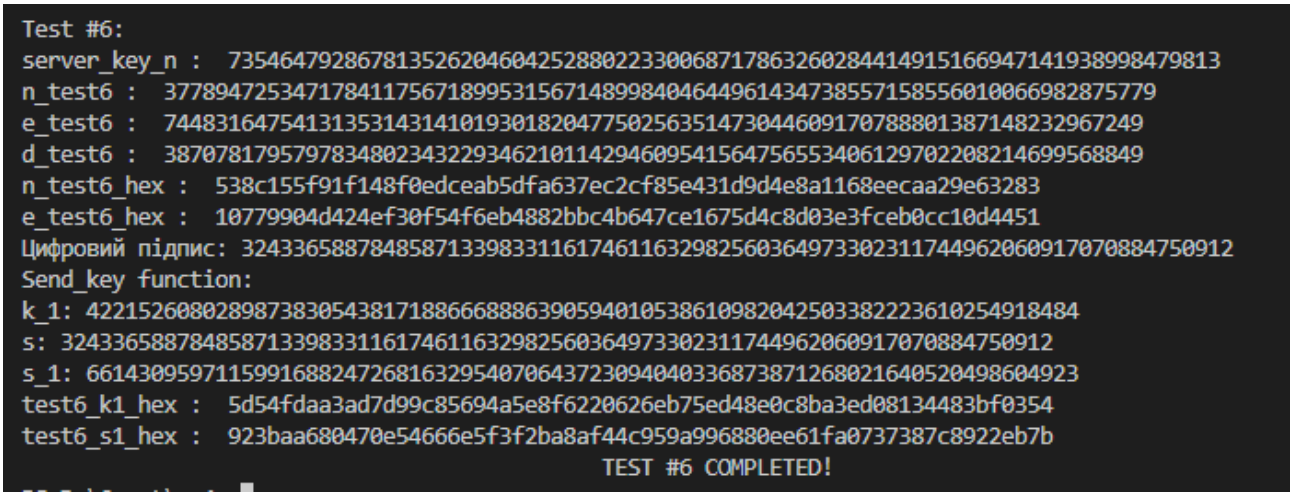



```
Test #5:
test5_k1:  38604626103972358397328908463628861399749734797758217278194626828968475297853950415244165454024106771280838283433687816859446397856241914803499113780135 50
test5_s1:  68765396486941261910824913562077959687401196519572081365691135763584964352911500650426476361387459293549818281018843538190446602056759064840952284315743 84
test5_k1_hex:  49b58c551918f4479bcf648d66b31425673ab32988f8d9d594c22f525891e8988012e1f9634d92dc213bcd2cff414279f44b834b624a500221ac6c1b5e2fad6e
test5_s1_hex:  834bd35d70eba9e6ac64c3b1f3011fe06a767c0119eede390e9c093b296afd3d1c74cf2f0bfd74487f9f6d8e344449450c24505bc3f20edbe0e5fa3514e82570
Recive_key function:
k_decrypted: 13011951155952883268
s_decrypted: 66624261137959800224273245802215687234107287142033330335718089333639489278 8
Перевірка цифрового підпису вірна.
                                TEST #5 COMPLETED!
```

Для тесту №5 було взято значення d_test2_4_5 та n_test2_4_5 з тесту №2.

**Test #6 Recive key:**






**Труднощі, які виникли під час виконаня роботи**

Під час перевірки функції Recive Key на сайті, ключ ніяк не підтверджувався, допоки я не згадав, що для перевірки треба згенерувати нові ключі, щоб **n_test6 < server_key_n.**

**Висновок**: в цій лабараторній роботі було вивчено тести перевірки чисел на простоту і методи генерації ключів для асиметричної криптосистеми типу RSA; практичне ознайомлено з системою захисту інформації на основі криптосхеми RSA, організовано з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчено протокол розсилання ключів.