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

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ № 4

Виконала:
студентка
групи ФБ-13,
Буєва Христина.

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

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 і рекомендації щодо 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 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq ≤ p1q1 ; p і q – прості числа для побудови ключів абонента A, 1 p 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) , (,) 1 n1 e та секретні d і d1 .

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 < n

Хід роботи

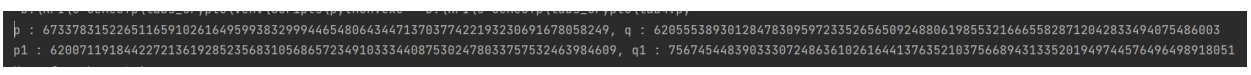
Завдання 1.

Було реалізовано функції gorner(), miller_rabin(), get_random_prime_num().

Завдання 2.

Реалізовано функцію get_pair_pq(). Кандидатів, що не пройшли було дуже багато, тому вивела їх лише один раз і записала в текстовий файл.

Результат виконання :



Завдання 3.

Реалізовано функцію generate_rsa_keypair(), яка приймає значення пари (p, q), а повертає відкритий та секретний ключ.

Результат виконання :

```
Keys for abonent A :
Open key : (417868542552431205372010421078509714363590279152070852218600359695754646663928756116630244058539928571292649967564755194696443319780637245118939860181887 47, 4718873277
Keys for abonent B :
Open key : (469236052106087962456954722035162661400051999487061393544677513670788416115734676913484691784328530173566170791743309356844769185451546767065049281627705 9, 2929540321
```

Завдання 4.

Реалізовано такі функції :

```python
#Task 4 ------------------------------------------------------------------------------------------------------------------
def encrypt (text, open_key):
    n, e = open_key
    encrypted_msg = pow(text, e, n)
    return encrypted_msg

def decrypt (encrypted_text, private_key):
    n = private_key[1]*private_key[2]
    d = private_key[0]
    decrypted_msg = pow(encrypted_text, d, n)
    return decrypted_msg

def sign(text, private_key):
    d, p, q = private_key
    sign = pow(text, d, p*q)
    return sign

def verify (d_text, open_key, sign):
    n, e =open_key
    m = pow(sign, e, n)
    if m == d_text:
        result = True
    else:
        result = False
    return result
```

Абонент А шифрує своє повідомлення та підписує. Також розшифруємо зашифроване повідомлення та перевіримо підпис (все теж саме зроблено для абонента Б):

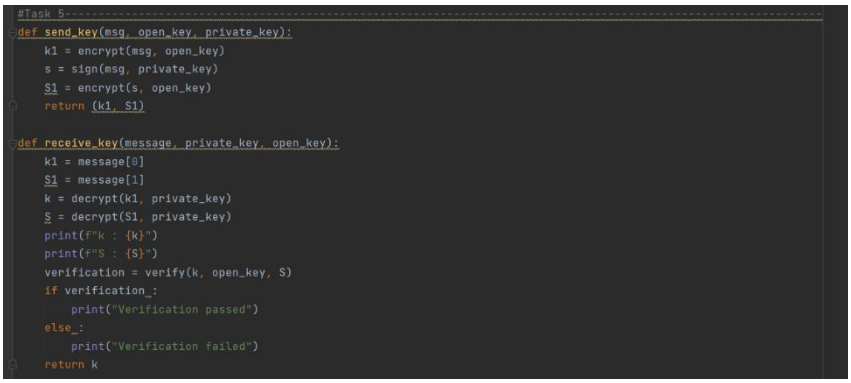
```python
#result task 4------------------------------------------------------------------
print("Шифроване та розшифроване повідомлення абонента А :")
msg_A = int("2947582035613856946202643891 63")
enc_msg_abonent_A = encrypt(msg_A, key[0])
print(f"Encrypted message : {enc_msg_abonent_A}")
dec_msg_abonent_A = decrypt(enc_msg_abonent_A, key[1])
print(f"Decrypted message : {dec_msg_abonent_A}")
print("Абонент А підписав повідомлення :")
signed_msg_A = sign(msg_A, key[1])
print(f"Sign : {signed_msg_A}")
print(f"Перевірка підпису : {verify(dec_msg_abonent_A, key[0], signed_msg_A)}")

print("Шифроване та розшифроване повідомлення абонента В :")
msg_B = int("1234567890947856123488226 1037")
enc_msg_abonent_B = encrypt(msg_B, key1[0])
print(f"Encrypted message : {enc_msg_abonent_B}")
dec_msg_abonent_B = decrypt(enc_msg_abonent_B, key1[1])
print(f"Decrypted message : {dec_msg_abonent_B}")
print("Абонент В підписав повідомлення :")
signed_msg_B = sign(msg_B, key1[1])
print(f"Sign : {signed_msg_B}")
print(f"Перевірка підпису : {verify(dec_msg_abonent_B, key1[0], signed_msg_B)}")
```

```
Шифроване та розшифроване повідомлення абонента А :
Encrypted message : 2584567458390432731645693716267486958358230919093692823750379316452574389321356230564038527929312027135572128156187165422054629839472011983602923872343815
Decrypted message : 2947582035613856946202643891 63
Абонент А підписав повідомлення :
Sign : 1620761409121875779369312749208182129838124338635240642846516745123287919042771423604108646427584572493633022571485589351064895883059630561591603722500994
Перевірка підпису : True
Шифроване та розшифроване повідомлення абонента В :
Encrypted message : 2980324672324401055982677446012751827880455630143504898249676472041945312586854448840733073583118937072091250936637262539565647433394853983735998115097947
Decrypted message : 1234567890947856123488226 1037
Абонент В підписав повідомлення :
Sign : 2122193554609666447487510871350695491552508697743166390988416228635633629513689362818102438148737658995957824456552566520366171551630307730739325457982900
Перевірка підпису : True
```

Завдання 5.

Протокол конфіденційності розсилання ключів зроблено аналогічно до вказівок в методичці :

Абонету а відомі свої ключі та відкритий ключ абонента Б. Абонент а формує повідомлення (k1, S1), яке відправляє абоненту Б. Абонент Б своїм пиватним ключем розшифровує повідомлення, та перевіряє підпис, маючи відкритий ключ абонента А.



```
#Task 5
def send_key(msg, open_key, private_key):
    k1 = encrypt(msg, open_key)
    s = sign(msg, private_key)
    S1 = encrypt(s, open_key)
    return (k1, S1)

def receive_key(message, private_key, open_key):
    k1 = message[0]
    S1 = message[1]
    k = decrypt(k1, private_key)
    S = decrypt(S1, private_key)
    print(f"k : {k}")
    print(f"S : {S}")
    verification = verify(k, open_key, S)
    if verification_:
        print("Verification passed")
    else_:
        print("Verification failed")
    return k
```

```
Абонент А підписав повідомлення :
Sign : 162076140912187577936931274920818212983812433863524064284651674512328791904277142360410864642758457249363302257148558935106489588305963056159160372250099 4
Перевірка підпису : True
Шифроване та розшифроване повідомлення абонента В :
Encrypted message : 29803246723244010559826774460127518278804556301435048982496764720419453125868544488407330735831189370720912509366372625395656474333948539837359981150979 47
Decrypted message : 12345678900947856123488226103 7
Абонент В підписав повідомлення :
Sign : 21221935546096664474875108713506954915525086977431663909884162286356336295136893628181024381487376589959578244565525665203661715516303077307393254579829 00
Перевірка підпису : True
Протокол конфіденційного розсилання ключів :
k : 2947582035613856946202643891 63
S : 162076140912187577936931274920818212983812433863524064284651674512328791904277142360410864642758457249363302257148558935106489588305963056159160372250099 4
Verification passed
```

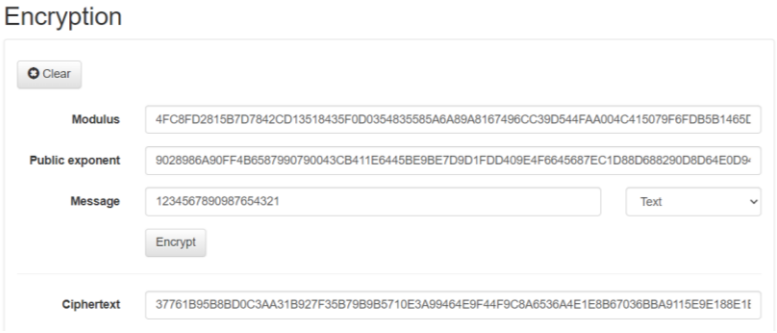
Далі наведений повний вивід, оскільки на скріншотах не все було видно, наприклад приватні ключі.

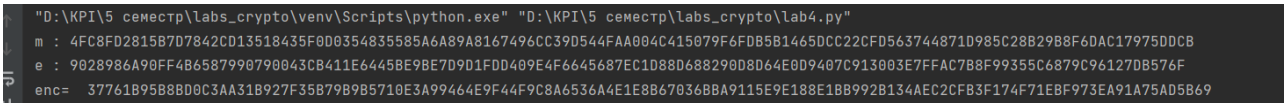
```
p :
673378315226511659102616495993832999446548064344713703774221932306916780582 49, q
: 62055538930128478309597233526565092488061985532166655828712042833494075486003

p1 :
62007119184422721361928523568310568657234910333440875302478033757532463984609,
q1 :
75674544839033307248636102616441376352103756689431335201949744576496498918051

Keys for abonent A :

Open key :
(41786854255243120537201042107850971436359027915207085221860035969575464663928 75
61166302440585399285712926499675647551946964433197806372451189398601818874 7,
47188732770409573358714355064473419752184245292288217414745132085970690584093244
22300433156119305175174520278988493600834509652704151839851667811673926 23),
Private key :
(36168694441643298074084728514113432715026362509719911842174261220637404680027 33
77757126705239768602425044149941020167320905120804543413119124805958609025 5,
673378315226511659102616495993832999446548064344713703774221932306916780582 49,
62055538930128478309597233526565092488061985532166655828712042833494075486003)

Keys for abonent B :

Open key :
(46923605210608796245695472203516266140005199948706139354467751367078841611573 46
76913484691784328530173566170791743309356844769185451546767065049281627705 9,
29295403211530997713553426745955628557254410911420553710329509904436076671798427
60017410381971343788694027569260029499513074603873387841059005610513594159),
Private key :
(73148521018912966353617895452736767200675192992009599647278300974471492559839 91
```

```
026723742417551631901558821943054488334457045922020912668981557711780658639,
620071191844227213619285235683105686572349103334408753024780337575324639846 09,
756745448390333072486361026164413763521037566894313352019497445764964989180 51)

Шифроване та розшифроване повідомлення абонента A :

Encrypted message :
258456745839043273164569371626748695835823091909369282375037931645257438932135 62
305640385279293120271355721281561871654220546298394720119836029238723438 15

Decrypted message : 294758203561385694620264389163

Абонент A підписав повідомлення :

Sign :
162076140912187577936931274920818212983812433863524064284651674512328791904277 14
236041086464275845724936330225714855893510648958830596305615916037225009 94

Перевірка підпису : True

Шифроване та розшифроване повідомлення абонента B :

Encrypted message :
298032467232440105598267744601275182788045563014350489824967647204194531258685 44
488407330735831189370720912509366372625395656474333948539837359981150979 47

Decrypted message : 12345678900947856123488226 1037

Абонент B підписав повідомлення :

Sign :
212219355460966644748751087135069549155250869774316639098841622863563362951368 93
628181024381487376589959578244565525665203661715516303077307393254579829 00

Перевірка підпису : True

Протокол конфіденційного розсилання ключів :

k : 294758203561385694620264389163

S :
162076140912187577936931274920818212983812433863524064284651674512328791904277 14
236041086464275845724936330225714855893510648958830596305615916037225009 94

Verification passed
```

Перевірка на сайті :

Encryption : бачимо, що зашифровані повідомлення збігаються



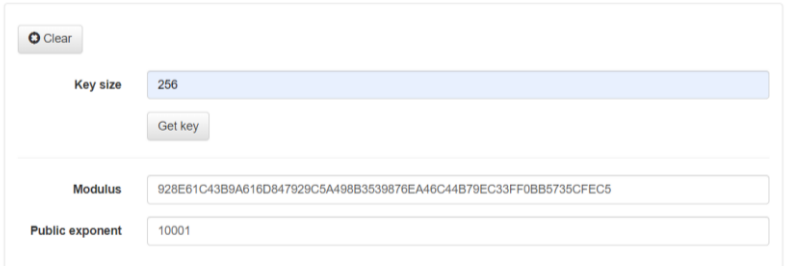


Зашифрувала у себе повідомлення ключем з серверу :

```
msg =  "12345678909876543 21"
```

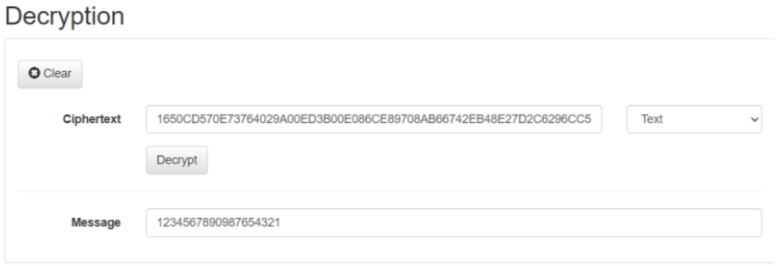
```
Encrypted for server =  1650CD570E73764029A00ED3B00E086CE89708AB66742EB48E27D2C6296CC55F
```

Get server key



Перевіримо decryption : розшифровано правильно

Decryption



Перевіримо підпис :

Підписуємо повідомлення на сервері та перевіримо в себе : результат перевірки True. Далі підпишемо повідомлення самі і перевіримо на сайті :

Sign


```
"D:\KPI\5 семестр\labs_crypto\venv\Scripts\python.exe" "D:\KPI\5 семестр\labs_crypto\lab4.py"
m : 4FC8FD2815B7D7842CD13518435F0D0354835585A6A89A8167496CC39D544FAA004C415079F6FDB5B1465DCC22CFD563744871D985C28B29B8F6DAC17975DDCB
e : 9028986A90FF4B6587990790043CB411E6445BE9BE7D9D1FDD409E4F6645687EC1D88D688290D8D64E0D9407C913003E7FFAC7B8F99355C6879C96127DB576F
enc=  37761B95B8BD0C3AA31B927F35B79B9B5710E3A99464E9F44F9C8A6536A4E1E8B67036BBA9115E9E188E1BB992B134AEC2CFB3F174F71EBF973EA91A75AD5B69
Encrypted for server =  1650CD570E73764029A00ED3B00E086CE89708AB66742EB48E27D2C6296CC55F
True
My sign =  4DAC4D7B503AE70A8A4A4CB563ED93EA8D0F5C5EB6FC69ADCFB646452D45735E49B3339FB85626F7B0E24C1CC54B3F35DC6A8054302B37996938C3C6A688ECD9
```

Verify

Висновки : під час виконання лабораторної роботи я ознайомилася з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практично ознайомилася з системою захисту інформації на основі криптосхеми RSA, а також організувала з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.