Лабораторна робота №4

Пошивак Н.
Боровков Н.
ФБ-02

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. 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().

Code output:

```
keys:
837361444779...
136667270239...
342853857522...
976638867817...

#           INFO
#    _Alice
# Public key:
> e =
203203423083113548892526943264740317868312250029426127579738765080039822658862608920077472200321804300733714864956562575066776841799703722036086577286513
> n =
114439902862148526424303469867075791073947171433411307102831471757998524842873560715949732624083777050707176260626896547751338220230254131378829357530801 7
# Private key:
> p =
837361444779912580115270700625836437987431157584289992405503150804725237539697
> q =
136667270239827286667200335026285924381710764834899607463314086710992961681937
> d =
378265199818494894806229864343846927224132536830429911111993741715276395019299026420018833007550245357097369718143723467527283653811474529913417264903057

#    _Bob
# Public key:
> e =
711080927201309088990962169133095955186454316830551159808298399491587987653184371557839184597619205586630920655091063740218677881465824274386240881674287
> n =
334844403237367538318913091927972007749792572274503619777644367819577472412778998964281904434958007561495153130655495492235088638695065942329247994607243 7
# Private key:
> p =
342853857522344304446723987302053816399092842932102941026960515128964152827377
> q =
976638867817157991643429848711028282083108660634245727173717560309181172681017
> d =
166182069649554363155554527144090735570702153901266588727333635568764551957678557458867961080093681071945594774556433916518973101056220276097572057680782 3
```

Message transfer process

> Generated message:

731078170727373018312976277883041327337475783271944873973178576368821869588627460443742590730284303335580025263602480558622239323630139560064420380715599

> Encrypted message:

882216645587015542473943831071817778536124301598005649194967958843019003607146255852488385837951594765333469479985929373957952100832323382074906004347 89

[+] Bob got the message

> Decrypted key:

665168542179528356494998852816343250559229463234264055710705365620140713547965789575446187832994408088640668700899040390659010855363993134796492643615168

[+] Bob have received the message.

[+] Bob have decrypted the message.

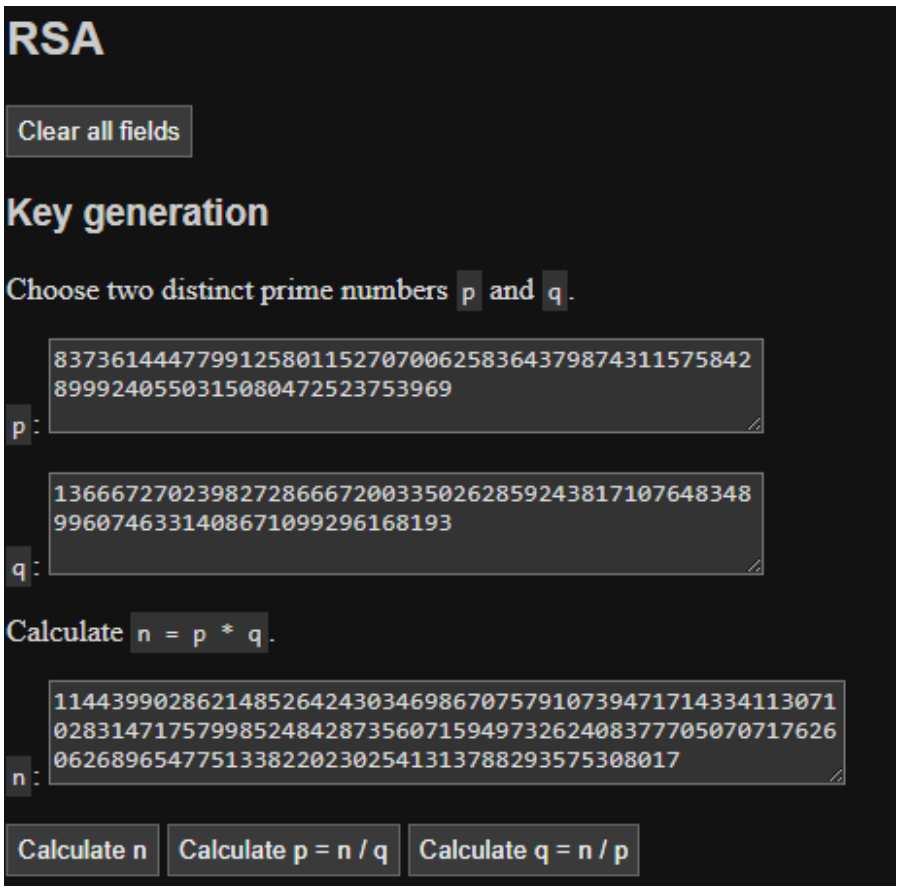
> Decrypted message:

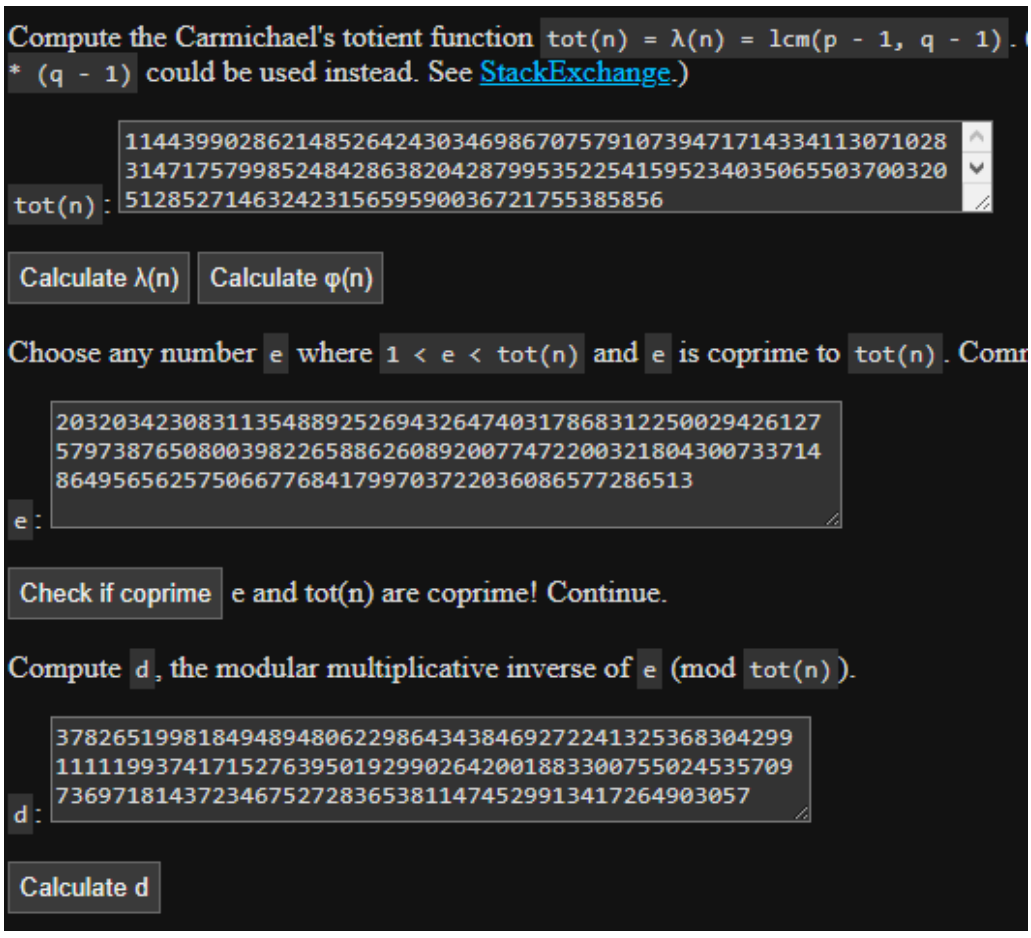
731078170727373018312976277883041327337475783271944873973178576368821869588627460443742590730284303335580025263602480558622239323630139560064420380715599

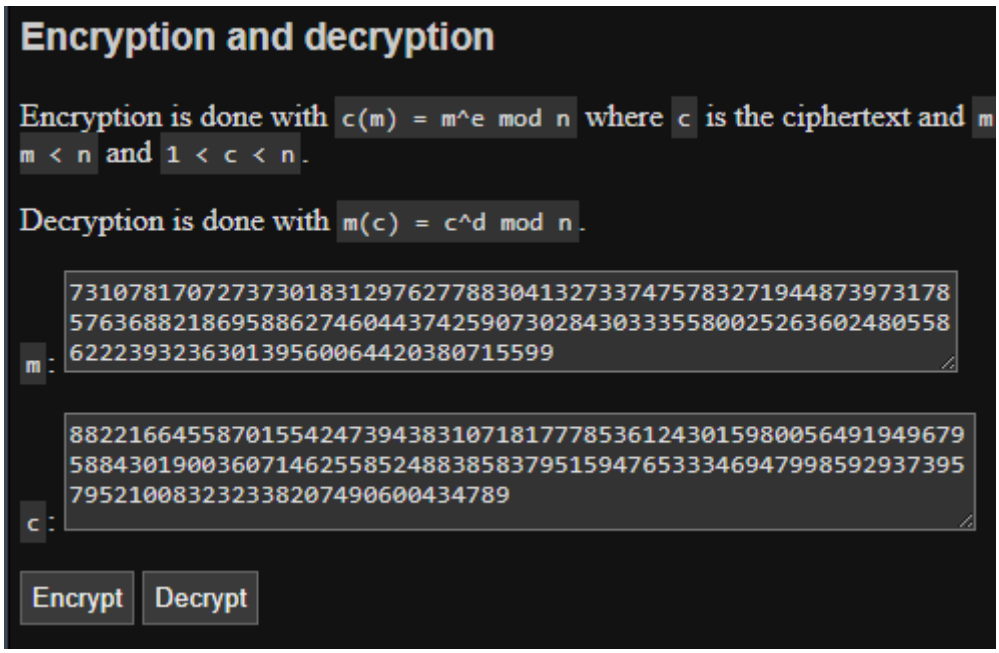
[+] Decrypted message is equal to original.

Process finished with exit code 0

Перевірка:


RSA
Clear all fields
Key generation
Choose two distinct prime numbers p and q.
p :
837361444779912580115270700625836437987431157584289992405503150804725237539 69
q :
136667270239827286667200335026285924381710764834899607463314086710992961681 93
Calculate n = p * q.
n :
114439902862148526424303469867075791073947171433411307102831471757998524842873560715949732624083777050707176260626896547751338220230254131378829357530801 7
Calculate n
Calculate p = n / q
Calculate q = n / p


Compute the Carmichael's totient function tot(n) = λ(n) = lcm(p - 1, q - 1). (
* (q - 1) could be used instead. See StackExchange.)
tot(n) :
114439902862148526424303469867075791073947171433411307102831471757998524842863820428799535225415952340350655037003205128527146324231565959003672175538585 6
Calculate λ(n)
Calculate φ(n)
Choose any number e where 1 < e < tot(n) and e is coprime to tot(n). Comr
e :
203203423083113548892526943264740317868312250029426127579738765080039822658862608920077472200321804300733714864956562575066776841799703722036086577286513
Check if coprime
e and tot(n) are coprime! Continue.
Compute d, the modular multiplicative inverse of e (mod tot(n)).
d :
378265199818494894806229864343846927224132536830429911111993741715276395019299026420018833007550245357097369718143723467527283653811474529913417264903057
Calculate d



Усі значення збігаються із результатами програми