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

“КРИПТОГРАФІЯ”

Комп’ютерний практикум
Робота № 4

Виконав:

студент групи ФБ-93 Проценко О.А.

Перевірила:

Селюх П. В.

Київ – 2021

1. **Назва роботи**

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

   Мета

   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.

2. Особливості роботи:
   - 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.
   - За допомогою цієї функції згенерувати дві пари простих чисел p, q і p_1, q_1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $p*q > p_1 q_1$; p і q – прості числа для побудови ключів абонента A, p_1 і q_1 – абонента B.
   - 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_1,n_1) та секретні d і d_1.
   - 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.
   - 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 $0 < k < n$.
3. Робота виконана місті Сміла. Робота створена на мові програмування Python.
4. Робота над завданням.
   - Значення вибраних чисел p, q, p_1 , q_1 із зазначенням кандидатів, що не пройшли тест перевірки простоти, і параметрів криптосистеми RSA для абонентів A і B.

     Для тестування чисел на простоту використав тест Міллера-Рабіна. Обробив виключні ситуації. Створив дві додаткові функції для тесту ObjGeneratePrimeNumber (повертає потрібне просте число) та ObjGeneratePrimeCandidate (включає створення числа з бітів). Також всі числа обрані довжиною 256.

```
(A) personal key >>>
d >>> 29748502236411670172869249399615494940311247699757523680993727889259469764729094817940411192797797511228588591720399512819412139502623309315523095218814910
p >>> 58337203529073413672740794242890352876732063402799530610896734059336270979541
q >>> 617211825921760854359912018768228774544164616220138838904647219199713574415610

(A) common key >>>
e >>> 3013245804773202304089860889528768629350253243457294797672054079917712826807214268263330530841357372246740075796036385854489643631376206603747442253982011
n >>> 3600641190934879277785908267372653135848520674331582176331068905131857658554738978902136044079523497665891536232459216356851906194253793403636157134103501

(B) personal key >>>
d_1 >>> 3142974343663140106741998446801101542717341593883635016631160262066860120108488739328852976436512809091309802935900692227139409212184705641044452603199181
p_1 >>> 70333582774056243578084869148853190784070255591242231792120835779359653678331
q_1 >>> 10193302663371383834194337842385272572321677603936193019506512060133984173591 9

(B) common key >>>
e_1 >>> 4341758895073884952276550638760759053128066541108963253598222860586097053174818707067295303990550965603913343896525852732293666345189578444764277534518181
n_1 >>> 7169314966152391906195229853302148744167514631760556169459140772420125686345351217823775640718014297280739801469765982378272125325096485373205660674671189
```

Готові прості числа для обробки даних.

```
p_1 >>>  719147687139548435113871322941169428700513926637235899725345460787838505756 13
q_1 >>>  1037009737854986664624040890999941353739526950580222923921255552441100548043 01
p_1 >>>  9516894282126383705137856622522324624652580501499407300120589827123171364031 1
q_1 >>>  7563593453825246397109626152366421336354402176070790284057423803416607831451 7
p_1 >>>  7926823997607522943419498950347378753336777376039823590380714104709220333158 3
q_1 >>>  6419691588456540380202742373761291307733973687666714418986965161932497621913 7
p_1 >>>  9275478525284257502454345961745908082389281083865615054846143193392612562986 9
q_1 >>>  6129985487563578887660149459317139102022966048528068872829669674135422126809 7
p_1 >>>  7506789997490429083099781081968308524423944464803920015391545211544402512152 7
q_1 >>>  9956907693977531429298013146440920346957578737481921798234547299195469115881 3
p_1 >>>  9836806072791876828544992606188301509054740575979400489249795903370670076627 9
q_1 >>>  8914271691403689450506160023305252959214120304393144773525696657242311990225 9
```

Числа які відразу не пройшли перевірку $p*q > p_1 q_1$.

- Чисельні значення прикладів ВТ, ШТ;

```
Plain numbers >>> 178029809857546073197944684473531648471303144986899543605494488352273395056819734986471488896948202880400612111669212846 1

Encrypted info >>> 64666332026332376961336068854791781359816732612285754531005188977272617721352563108029220591723773727238891927125794747856754766571297417707917708521475 0

Indication for encrypted info >>> 10262392696188755337868406095164879928431330574132748894591856332637170419290283082088329449811307847086070869325051539280181966897709770479101306 3

The numbers is verified

Decrypted info >>> 178029809857546073197944684473531648471303144986899543605494488352273395056819734986471488896948202880400612111669212846 1
```

5. Висновки.

Детальніше розібрався в криптосистемі RSA. Провів аналіз алгоритму, зрозуміючи навіщо потрібні приватні та відкриті ключі. Отримав новий досвід з тестами для перевірки на просте число. Порівняв тест Міллера-Рабіна з решето Ератосфена. Отримав нові навички роботи з мовою програмування Python.