

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”  
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

**КРИПТОГРАФІЯ**  
**КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4**

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

Виконали:  
ФБ-32 Рибчук Нікіта  
ФБ-32 Луценко Євгеній

**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  $1 < p, q$  довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб  $pq \leq p_1q_1$ ;  $p < q$  – прості числа для побудови ключів абонента А,  $1 < p < q_1$  – абонента В.
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 В – тобто, створити та зберегти для подальшого використання відкриті ключі  $(e, n)$ ,  $(, )$  і  $n_1 e$  та секретні  $d$  і  $d_1$ .
4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і В. 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 М і знайти криптограму для абонентів А і В, перевірити правильність розшифрування. Скласти для А і В 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().

## Хід роботи

### Генерація ключів для А і В:

```
>>> 1. Генерація ключів...
[Абонент А]
Public Key (e) (Dec): 65537
Public Key (e) (Hex): 10001
-----
Modulus (n) (Dec): 7797209521010394527509602226917169357838896547104690495970780907151493221578392285224416476739982876480449997841036463216425803358694741345522987932893961
Modulus (n) (Hex): 94DFF75875605AF7C04F9C4E7878058D36A8917900A728CE35CD8E565632FABADF20AE048C55EAF301D7D936E1CB0B27D2539451FAADCFE221F115FD5ACC309
-----
Private Key (d) (Dec): 569053407067652120559049505643298000191394817962398868459469384910899065853008049270507842966090774190021283540662923703496076934443363161440370388598189
Private Key (d) (Hex): AD079A10C24F71EA3C368A5F276D7F8E167560E90B495A6D185F6DFFA682D3E9887A7C03400748BE390F57E291E38AE3C7C819E1F11CC65C69046039177DAD
-----
[Абонент В]
Public Key (e1) (Dec): 65537
Public Key (e1) (Hex): 10001
-----
Modulus (n1) (Dec): 8695013373964033401099152168039626208725997791178302089281123346052655347086332387443047430458457921714083058475079297557664410303547450730630646918083997
Modulus (n1) (Hex): A604570E250800106AEC4D2D2211C2EC0040AC732B360E95667614908C465A5CC9270A2AF1CAB991477E192CEF5BA544DDCBC2760957E838247D07FD838F590
-----
Private Key (d1) (Dec): 9592283243627581122033157109612123669208178632252199681862950382166864762550672054146349392870059976829304741227785223470515282109163092508989767121153
Private Key (d1) (Hex): 1D4DC6D00859088B58943D482A28586FAFDBBA94D998C1173D0E89D2926918A9D0FDEE8138C5850E2EAB81288A85A1F7228BAED2052E84F56E1338492657901
-----
```

### Шифрування та дешифрування повідомлення:

```
>>> 2. Шифрування тексту (A -> B)
Вхідний текст: Hello IPT
Текст як число (Dec): 1335473908826942034004
Текст як число (Hex): 48656C6CF20495054
-----
Зашифроване повідомлення (Ciphertext) (Dec): 4220158576372890102559659854729526857791254201761302755052231200457111629607240569754466766639276129681300388869794460411189955043179824973170143532566439
Зашифроване повідомлення (Ciphertext) (Hex): 5093849728E24E14E183315E4990C774E628E86A2165F80B9235613ACAC02687C73E85743D289CE0A764522C74C0303EC9DE9F7F9A74741FF24931E0647A7
-----
Розшифроване число (Dec): 1335473908826942034004
Розшифроване число (Hex): 48656C6CF20495054
-----
Результат: 'Hello IPT'
```

### Цифровий підпис та його перевірка:

```
>>> 3. Цифровий підпис повідомлення
Цифровий підпис S (Dec): 1101046468478866248085468644965944607533704786437868669681581240294774591259646819891771446120207284653904361555644865769019004179448272992927447558401155
Цифровий підпис S (Hex): 1505C09372CB4C374866C485423B8FF6D0011F843B84F1DC080877BDC202A1A3770955AA4A635D08E918759078253655F0E4C79D918E0923CC37001CB42CB3
-----
Перевірка підпису: УСПІХ
```

### Протокол обміну сеансовим ключем k:

```
>>> 4. Протокол обміну сеансовим ключем k
Згенерований ключ k (Dec): 2010825095786721942189778505447587253329979833470713543684512227738329046268930542340624055942288880187773308704829052265908067239137188011214551485268695
Згенерований ключ k (Hex): 2664855510290198F89730FF4305EB3D58AF4FB4EF989827EA25D08004962B4133889904E818D2C4385626A553D0B0096BAE37576ACA735659BF8E2555A842D7
-----
А відправив пакет (k1, S1)
k1 (Encrypted k) (Dec): 6024576455927503602208208508601788747879094912859930856511595804858211105424150328722265719076869414963201667029652336152079374205573770174131040852893
k1 (Encrypted k) (Hex): 7307842C7CF3939CDF3F463EAEAB416AFEF20844268CF7B158A39D237F8868F40D784F9CA6E76E6D09958FCFC712AACDC89587F75613EFC85C0984666069B9D
-----
S1 (Encrypted Signature) (Dec): 705802447728854987827674146997011292535589559467644345123601890664015435948398521627503473278955967487752113491202087319970303458265973778901201014630833
S1 (Encrypted Signature) (Hex): 86C2E7AE9E6E5F8AF5A9124E79B30C03AF05F3028D7655ECD6E481FC3CE84499E379E90C47155171A27023AD3A2D0880266C48C95E9BC34A4D100FC64E2981
-----
В отримав та обробив пакет
Отримане k (Dec): 2010825095786721942189778505447587253329979833470713543684512227738329046268930542340624055942288880187773308704829052265908067239137188011214551485268695
Отримане k (Hex): 2664855510290198F89730FF4305EB3D58AF4FB4EF989827EA25D08004962B4133889904E818D2C4385626A553D0B0096BAE37576ACA735659BF8E2555A842D7
-----
Автентичність відправника (A): ПІДТВЕРДЖЕНО
```

## Перевірка на сайті

Генерація ключа:

### RSA Testing Environment

Server Key

Encryption

Decryption

Signature

Verification

Send Key

Receive Key

#### Get server key

Clear

Key size256

Get key

ModulusC212EBB1BFFA7487D068F59EA6AE15B82D8058598FF4B99375495AB16F65AA6B

Public exponent10001

Сигнатура з сайту:

### RSA Testing Environment

Server Key

Encryption

Decryption

Signature

Verification

Send Key

Receive Key

#### Sign

Clear

MessageHello IPT

Text

Sign

Signature68B253CF9A1AF839211F3318F39CF59006C631451B93E3C4E01A647E01B49A78

Шифротекст згенерований нашим ключем із коду:

### RSA Testing Environment

Server Key

Encryption

Decryption

Signature

Verification

Send Key

Receive Key

#### Encryption

Clear

Modulus69DBB96494277D97714069C82222FB5E3A7F8D6AC43AD33BA983F9F5E587925F80D19DC86766DC26188BE

Public exponent10001

MessageHello IPT

Text

Encrypt

Ciphertext19373CD60F5D854F3FF284165473CE00AE26B84E69341688C45EDFF075825DB6812AF84C33967E352F5B1C

Верифікація на сайті для ключа згенерованого кодом:

## RSA Testing Environment

[Server Key](#)  
[Encryption](#)  
[Decryption](#)  
[Signature](#)  
[Verification](#)  
[Send Key](#)  
[Receive Key](#)

### Verify

✕ Clear

Message

Hello IPT

Text ▼

Signature

34BC3640C024EC319AE5060727E104FAE3A9EBABE1855E84D926BF046952001A06B5AA161EAF8AA016C

Modulus

69DBB96494277D97714069C82222FB5E3A7F8D6AC43AD33BA983F9F5E587925F80D19DC86766DC26188BE

Public exponent

10001

Verify

Verification

true

✓

Шифрування на сайті ключем із сайту:

## RSA Testing Environment

[Server Key](#)  
[Encryption](#)  
[Decryption](#)  
[Signature](#)  
[Verification](#)  
[Send Key](#)  
[Receive Key](#)

### Encryption

✕ Clear

Modulus

C212EBB1BFFA7487D068F59EA6AE15B82D8058598FF4B99375495AB16F65AA6B

Public exponent

10001

Message

Hello IPT

Text ▼

Encrypt

Ciphertext

860E6A65BDCDF89BF7BCC718B1466E31F74A37EA9EB39B378D63DDD139A63B5C

Перевірка шифрування кодом для ключа із сайту:

```
>>> Тест 1: Шифрування 'Hello IPT' (для Сайту)
860E6A65BDCDF89BF7BCC718B1466E31F74A37EA9EB39B378D63DDD139A63B5C
```

Перевірка підпису для ключа із сайту:

```
>>> Тест 2: Перевірка підпису сайту
Результат: УСПІХ
```

Перевірка дешифрування шифротексту отримано із сайту ключем із коду:

```
>>> Тест 3: Дешифрування шифротексту з сайту
Результат:
HEX: 48656C6C6F20495054
TEXT: 'Hello IPT'
```

Генерація підпису і перевірка його на сайті для нашого ключа із коду:

```
>>> Тест 4: Генерація вашого підпису
34BC3640C024EC319AE5060727E104FAE3A9EBABE1855E84D926BF046952001A06B5AA161EAF8AA016C5496590E13480EBF775929246E8AD0F408E0688A3C8
```

## RSA Testing Environment

[Server Key](#)  
[Encryption](#)  
[Decryption](#)  
[Signature](#)  
**Verification**  
[Send Key](#)  
[Receive Key](#)

### Verify

Clear

Message

Hello IPT

Text

Signature

34BC3640C024EC319AE5060727E104FAE3A9EBABE1855E84D926BF046952001A06B5AA161EAF8AA016C

Modulus

69DBB96494277D97714069C82222FB5E3A7F8D6AC43AD33BA983F9F5E587925F80D19DC86766DC26188BE

Public exponent

10001

Verify

Verification

true

## Висновки

У ході виконання комп'ютерного практикуму було досліджено, а також програмно реалізовано асиметричну криптосистему RSA, включаючи алгоритм генерації великих простих чисел на основі ймовірного тесту Міллера-Рабіна. Було розроблено функції шифрування, дешифрування та цифрового підпису, а також реалізовано протокол конфіденційного обміну сеансовими ключами.

Практична робота дозволила глибше зрозуміти математичні основи та принципи роботи RSA, які забезпечують високий рівень безпеки.