# ЛАБОРАТОРНАЯ РАБОТА № 5

# АЛГОРИТМИ ШИФРУВАННЯ З ВІДКРИТИМ КЛЮЧОМ

**Мета роботи:** ознайомитись з різними шифрами простої заміни (шифрами підстановки) та методами їх дешифрування.

# Короткі теоретичні відомості

## Система шифрування RSA

RSA (абревіатура від прізвищ Rivest, Shamir та Adleman) — криптографічний алгоритм з відкритим ключем, що ґрунтується на обчислювальній складності завдання факторизації великих цілих чисел.

Криптосистема RSA стала першою системою, придатною і для шифрування, і цифрового підпису.

Криптографічні системи з відкритим ключем використовують так звані односторонні функції, які мають таку властивість:

- якщо відомо x то f(x) то розрахувати просто;

- якщо відомо y = f(x), то для обчислення x немає простого (ефективного) шляху.

Під однобічністю розуміється не математично доведена односпрямованість, а практична неможливість обчислити зворотне значення, використовуючи сучасні обчислювальні засоби, за доступний для огляду інтервал часу.

В основу криптографічної системи з відкритим ключем RSA покладено складність завдання факторизації добутку двох великих простих чисел. Для шифрування використовується операція зведення в ступінь модуля великого числа. Для дешифрування (зворотної операції) за розумний час необхідно вміти обчислювати функцію Ейлера від великого числа, навіщо необхідно знати розкладання числа на прості множники.

У криптографічної системі з відкритим ключем кожен учасник має у своєму розпорядженні як відкритий ключ (англ. public key), так і закритий ключ (англ. private key). У криптографічної системі RSA кожен ключ складається з кількох чисел. Кожен учасник створює свій відкритий та закритий ключ самостійно. Закритий ключ кожен з них тримає в секреті, а відкриті ключі можна повідомляти будь-кому або навіть публікувати їх. Відкритий та закритий ключі кожного учасника обміну повідомленнями у криптосистемі RSA утворюють «узгоджену пару» у тому сенсі, що вони є взаємно зворотними, тобто:

Вибираються два простих числа p і q (такі що p нерівно q).

Обчислюється модуль N = p \* q.

Обчислюється значення функції Ейлера від модуля N: F(N)=(p−1)(q−1).

Вибирається число e, зване відкритою експонентою, число e повинно лежати в інтервалі 1 і обчислюється число d, зване секретною експонентою, таке, що d∗e=1(modF(N)), тобто мультиплікативно зворотне до числа e за модулем F(N ).

Разом отримуємо:

Пара (e, N) – відкритий ключ.

Пара (d, N) – закритий ключ.

## Шифрування та розшифрування

Припустимо, Боб хоче надіслати Алісі повідомлення m.

Повідомленнями є цілі числа в інтервалі від 0 до n – 1, взаємно прості з n.

Тобто , .

Алгоритм шифрування:

- зяти відкритий ключ (e, n) Аліси;

- взяти відкритий текст m;

- зашифрувати повідомлення за допомогою відкритого ключа Аліси.

Алгоритм розшифрування:

- прийняти зашифроване повідомлення c;

- взяти свій закритий ключ (d, n);

- застосувати закритий ключ для розшифрування повідомлення;

## Хід роботи

1. Статична модель згідно з варіантом зображено на рисунку 1.3.

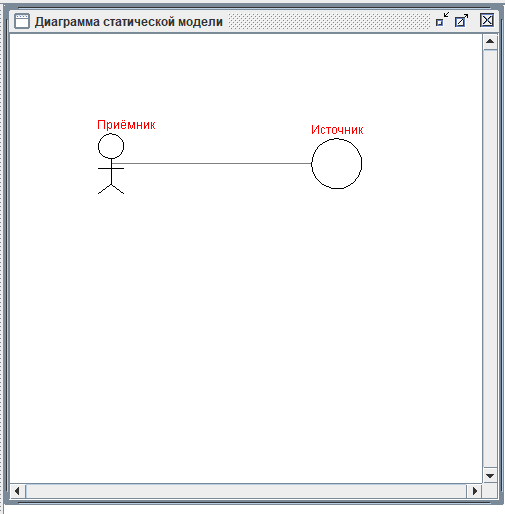


Рисунок 1.3 – Статична модель

1. Вікна статична модель, суб'єктна модель, діаграма взаємодії процесів, зображено на рисунку 1.4 – 1.6.

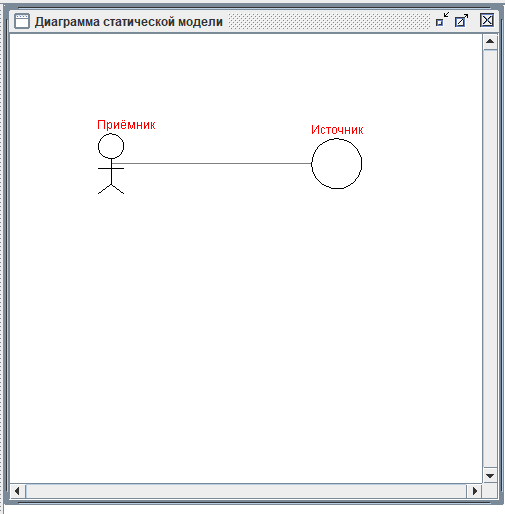


Рисунок 1.4 – Статична модель

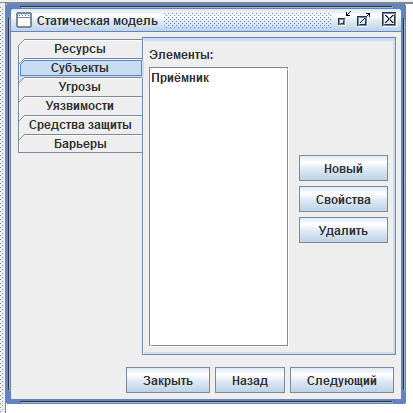


Рисунок 1.5 – Суб’єктна модель

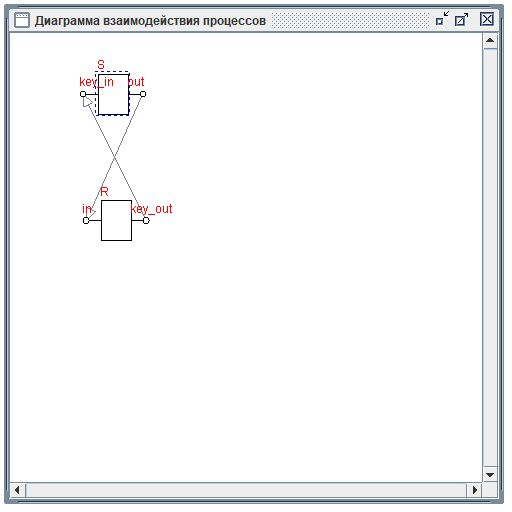


Рисунок 1.6 - Діаграма взаємодії процесів

Програма призначена для кодування та декодування слів шифром DES.

Метод makeKeys призначений для створення ключа шифрування. Код методу наведено в лістингу 1.7.

Лістинг 1.7. Метод makeKeys

long x = 1000;

long y = 10000;

long fi;

long p,q,e;

for ( p = x; !Numerical.isPrime( p ); p++ ) ;

for ( q = y + 2; !Numerical.isPrime( q ); q++ ) ;

rsa.setSecureKey(p,q);

logDataMessage("Секретный ключ","p=" + p +" q="+q);

fi=rsa.getFi();

for ( e = fi / 10; Numerical.gcd( e, fi ) != 1; e++ ) ;

rsa.checkOpenKey(e);

logDataMessage("Открытый ключ", "e=" + e+" n="+rsa.getN() );

long open\_key[]=new long[2];

open\_key[0]=e;

open\_key[1]=rsa.getN();

send("key\_out",open\_key);

Метод do\_send призначений для шифрування тексту. Код методу наведено в лістингу 1.8.

Лістинг 1.8. Метод do\_send

long message=55545435;

logDataMessage("Исходный текст",message);

long code = rsa.encode( message );

logDataMessage("Зашифрованный текст",code);

send("out",new Long(code));

Метод onRecieve Service використовується для отримання ключа. Код методу наведено в лістингу 1.9.

Лістинг 1.9. Метод onRecieve Service

long key[]=(long [])recv("key\_in");

rsa.setOpenKey(key[0],key[1]);

logDataMessage("Получен открытый ключ","e="+key[0]+" n="+key[1]);

Метод onRecieve Resurce призначений для розшифрування тексту. Код методу наведено в лістингу 2.1.

Long lcode=(Long)recv("in");

long code=lcode.longValue();

logDataMessage("Шифротекст",code);

long decode = rsa.decode( code );

logDataMessage("Расшифрованный текст",decode);

**Висновок:** ознайомились з системами шифрування RSA.