НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

з лабораторної роботи № 4

із дисципліни «Криптографічні методи захисту інформації»

на тему

*Шифрування з відкритим ключем на основі алгоритму RSA*

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-01 | *ст. викладач Бай Ю. П.* |
| *Дюбакін Р. С.* |  |

Київ — 2022

ЗМІСТ

[Постановка задачі 2](#_Toc102990368)

[Основні теоретичні відомості з асиметричних криптосистем 3](#_Toc102990369)

[Математичне підґрунтя алгоритму RSA 3](#_Toc102990370)

[Опис алгоритму RSA 3](#_Toc102990371)

[Контрольний приклад 1 4](#_Toc102990372)

[Шифрування і розшифрування за алгоритмом RSA 5](#_Toc102990373)

[Контрольні питання 6](#_Toc102990374)

[Список літератури 7](#_Toc102990375)

[Додаток 1 8](#_Toc102990376)

[Додаток 2 9](#_Toc102990377)

***Мета роботи:*** розробити асиметричну криптосистему на основі алгоритму шифрування RSA.

# Постановка задачі

1. Скласти програму, яка дозволяє виконувати шифрування та розшифрування за алгоритмом RSA. Перевірити роботу програми на контрольних прикладах. Навести скріншоти детального покрокового виконання алгоритму.

1. *Контрольний приклад 1* ([RSA-encryption](https://brilliant.org/wiki/rsa-encryption/))

*p* = 11, *q* = 17, *e* = 3

*public key* {*e, n*}= {3*,* 187}

*private key* {*d, n*}= {107*,* 187}

*M* = 72

*C* = 183

*M’* = 72, *text* = *chr (72)* = {H}

*Контрольний приклад 2* ([RSA uk.wiki](https://uk.wikipedia.org/wiki/RSA))

*p* = 3557, *q* = 2579, *e* = 3

*public key* {*e, n*}= {3*,* 9173503}

*private key* {*d, n*}= {6111579*,* 9173503}

*M* = 111111

*C* = 4051753

*M’* = 1111111

2. Виконати дії ОДЕРЖУВАЧА і розшифрувати задане повідомлення, користуючись алгоритмом RSA. Необхідні результати занести до [Таблиця RSA](https://docs.google.com/spreadsheets/d/1IeqT1byTw2PgTOQXvXaozfia_Dh4pSM5RaWOLqZCtEM/edit?usp=sharing).

УВАГА! Числа ***n*** в стовпчику *D* мають бути унікальними.

# Основні теоретичні відомості з асиметричних криптосистем

Асиметричні криптосистеми — ефективні системи криптографічного захисту даних, які також називають криптосистемами з відкритим ключем. В таких системах для зашифровування даних використовують один ключ, а для розшифровування — інший (звідси і назва — асиметричні). Перший ключ є відкритим і може бути опублікованим для використання усіма користувачами системи, які шифрують дані. Розшифровування даних за допомогою відкритого ключа неможливе. Для розшифровування даних отримувач зашифрованої інформації використовує другий ключ, який є секретним (закритим). Зрозуміло, що ключ розшифровування не може бути визначеним з ключа зашифровування.

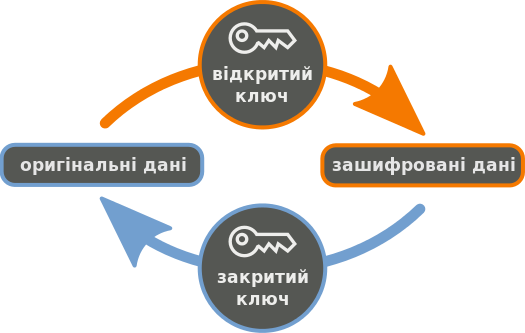
Головне досягнення асиметричного шифрування в тому, що воно дозволяє людям, що не мають наперед наявної домовленості про безпеку, обмінюватися секретними повідомленнями. Необхідність відправникові й одержувачеві погоджувати таємний ключ по спеціальному захищеному каналу цілком відпала. Прикладами криптосистем з відкритим ключем є Схема Ель-Гамаля (названа на честь автора, Тахера Ель-Гамаля), RSA (названа на честь винахідників: Рона Рівеста, Аді Шаміра і Леонарда Адлмана), Діффі-Геллмана і DSA, англ. Digital Signature Algorithm (винайдений Девідом Кравіцом).

Рисунок 1 – Принцип роботи асиметричної системи

# Математичне підґрунтя алгоритму RSA

В основі алгоритму RSA полягає складність задачі факторизації великих чисел. – це розклад числа на прості множники.

Взаємно прості числа — це цілі числа, які не мають спільних дільників, окрім 1, або, інакше кажучи, якщо їх найбільший спільний дільник дорівнює 1.

**Теорема 1.** **Мала теорема Ферма**

Якщо р – просте число, то mod p = 1 для будь-якого х, взаємно простого з p.

**Визначення.** Функцією Ейлера φ(n) називається число додатних цілих чисел, менших вiд n і взаємно-простих з n.

**Зауваження.** Якщо n ‒ просте число, то φ(n) = n ‒ 1.

**Теорема 2.**

Якщо n = p⋅ q, де p і q – прості числа, p ≠ q, то φ(n)=(p – 1)⋅ (q – 1).

|  |  |  |
| --- | --- | --- |
| *n = p⋅ q* |  | *φ(n)=(p – 1)⋅ (q – 1)* |

**Теорема 3.**

Якщо n = p⋅ q (p і q – прості числа, p≠ q), x – взаємно просте з p і q, то xφ(n)mod n = 1.

|  |  |
| --- | --- |
|  | *xφ(n)mod n = 1* |

**Наслідок 1.**

Якщо n = p⋅ q (p і q – прості числа, p≠ q), e – взаємно просте з φ(n), то відображення Ee,n: x→mod n буде взаємно однозначним.

**Наслідок 2.**

Якщо n = p⋅ q (p і q – прості числа, p≠ q), e – взаємно просте з φ(n), то існує таке ціле d, що e∙d mod φ(n) = 1.

e∙d mod φ(n) = 1

**Як працює алгоритм?**

Шифрування:

m = mod n

Розшифрування:

c = mod n

**mod n** = mod n = mod n

Оскільки виконується умова e∙d mod φ(n) = 1, то можна записати e∙d = k·φ(n) + 1

Тоді:

mod n = mod n;

За теоремою Ейлера:

mod n = 1

Тоді

mod n = m

Отже

mod n = m

# Опис алгоритму RSA

|  |  |
| --- | --- |
| Крок | Опис операції |
| 1 | Обрати два довільних простих числа *p* і *q (p ≠ q)* |
| 2 | Обчислити добуток *n = p* ∙ *q* |
| 3 | Обчислити функцію Ейлера *φ*(*n*) *=* (*p – 1*) ∙ (*q – 1*) |
| 4 | Обрати відкриту експоненту *e: 1 < e < φ(n), e -* взаємно просте з *φ(n)* |
| 5 | Обчислити секретну експоненту *d:* ***(e∙d) mod* φ*(n) = 1*** |
| 6 | Зберегти закритий ключ {*d*, *n*} |
| 7 | Опублікувати відкритий ключ {*е*, *n*} |
| 8 | Одержати від відправника / викладача зашифроване повідомлення - послідовність ***ci*** (в даному випадку в якості ***mi*** були використані індекси літер в англійському алфавіті) |
| 9 | Розшифрувати задане повідомлення ***mi* = (*ci*)*d* mod *n*** |
| 10 | Відповідь |

# Контрольний приклад 1

[RSA-encryption](https://brilliant.org/wiki/rsa-encryption/)

Виконати приклад. Додати скріншот, що містить усі проміжні результати.

*p* = 11, *q* = 17, *e* = 3

*public key* {*e, n*}= {3*,* 187}

*private key* {*d, n*}= {107*,* 187}

*M* = 72

*C* = 183

*M’* = 72, *text* = *chr (72)* = {H}

Зображення, що містить текст

Автоматично згенерований опис

# Контрольний приклад 2

*p* = 3557, *q* = 2579, *e* = 3

*public key* {*e, n*}= {3*,* 9173503}

*private key* {*d, n*}= {6111579*,* 9173503}

*M* = 111111

*C* = 4051753

Зображення, що містить текст

Автоматично згенерований опис*M’* = 111111

# Шифрування і розшифрування за алгоритмом RSA

*Таблиця 1.*

|  |  |  |
| --- | --- | --- |
| Крок | Опис кроку | Результат |
| 1 | Обрати два довільних простих числа *p* і *q*  *p ≠ q*; 1 < *p, q* < 200 | **P = 53**  **Q= 67** |
| 2 | Обчислити добуток *n = p* ∙ *q.* Увага! ***n >* 90 та має бути унікальним** в стовпчику *D* в [Таблиця RSA](https://docs.google.com/spreadsheets/d/1IeqT1byTw2PgTOQXvXaozfia_Dh4pSM5RaWOLqZCtEM/edit?usp=sharing) | **N = 3551** |
| 3 | Обчислити функцію Ейлера *φ*(*n*) *=* (*p – 1*) ∙ (*q – 1*) | **3432** |
| 4 | Обрати відкриту експоненту *e* : *1 < e < φ(n),  e –* взаємно просте з *φ(n)* | **155** |
| 5 | Обчислити секретну експоненту *d* таку, що *(e∙d) mod φ(n) = 1* | **155** |
| 6 | Зберегти закритий ключ {*d*, *n*} | **{155, 3551}** |
| 7 | Опублікувати відкритий ключ {*е*, *n*} | **{155, 3551}** |
| 8 | Одержати від відправника / викладача зашифроване повідомлення *C*. *Дії відправника:*  1) обрати текст для шифрування *M;* 2) символи тексту замінити цілими числами *mi* згідно з таблицею ASCII; 3) виконати шифрування за формулою ***ci = (mi)e mod n*** *.* | **[1844, 275, 1844, 2224, 2025, 1227, 275, 659]** |
| 9 | Розшифрувати задане повідомлення *C:*  1) обчислити ***mi = (ci)d mod n***; 2) поставити у відповідність знайденим цілим числам *mi* літери англійського алфавіту, записати одержане слово | **[77, 69, 77, 79, 82, 73, 69, 83]**  **MEMORIES** |

*В процесі шифрування використовується наступне перетворення літер англійської абетки в коди ASCII: ord(‘A’) = 65, chr(65) = ‘A’* .

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | D | … | Z |
| 🡫 | 🡫 | 🡫 | 🡫 |  |  |
| 65 | 66 | 67 | 68 | … | 90 |

# Контрольні питання

1. В чому полягає принципова відмінність асиметричних криптосистем від симетричних?

Симетричне шифрування передбачає використання однакових ключів для шифрування і розшифрування. Алгоритми асиметричного шифрування використовують різні ключі для шифрування і розшифрування.

1. Що таке одностороння (однонаправлена) функція з секретом? Наведіть приклади односторонніх функцій.
2. F(x) є відомою відкритою функцією
3. Для будь-якого x легко обчислити f(x)
4. Для будь-якого y = f(x) складно обчислити x
5. F(x) легко обчислити, якщо відомий секрет t

Приклад: f(x) = (x^4) mod n, n – деяке задане число

1. Складність якої математичної задачі полягає в основі алгоритму RSA?

В основі алгоритму RSA полягає складність задачі

факторизації великих чисел.

1. Як визначається і обчислюється функція Ейлера?

Функцією Ейлера φ(n) називається число додатних цілих чисел, менших вiд n і взаємно-простих з n.

φ(n) = (p – 1) ∙ (q – 1)

1. Як пов’язані між собою відкритий і закритий ключі в алгоритмі RSA?

Відкритий і закритий ключі пов’язані між собою

алгоритмічно.

|  |
| --- |
| Обрати два довільних простих числа *p* і *q (p ≠ q)* |
| Обчислити добуток *n = p* ∙ *q* |
| Обчислити функцію Ейлера *φ*(*n*) *=* (*p – 1*) ∙ (*q – 1*) |
| Обрати відкриту експоненту *e: 1 < e < φ(n), e -* взаємно просте з *φ(n)* |
| Обчислити секретну експоненту *d:* ***(e∙d) mod* φ*(n) = 1*** |
| Зберегти закритий ключ {*d*, *n*} |
| Опублікувати відкритий ключ {*е*, *n*} |

# Список літератури

1. Тарнавський Ю.А. Технології захисту інформації [Електронний ресурс] / Ю. А. Тарнавський. – Київ: КПІ ім. Ігоря Сікорського, 2018. – 162 с.
2. Шнайер Б. Прикладная криптография: Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М.: Диалектика, 2003. – 610 с.
3. Гулак Г*.*М*.* Основи криптографічного захисту інформації: підручник / Г*.*М*.* Гулак*,* В*.*А*.* Мухачов*,* В.О. Хорошко, Ю.Є. Яремчук / – Вінниця: ВНТУ, 2011. – 199 с.
4. Столлингс В. Криптография и защита сетей: принципы и практика, 2-е изд.: Пер. с англ. – М.: «Вильямс», 2001. – 672 с.
5. Саймон Сингх. Книга шифров. Пер. с англ. А. Галыгина. — М.: АСТ: Астрель, 2007. — 448 с.

# Додаток 1

Текст програми, що реалізує шифрування/розшифрування  
 за алгоритмом RSA

import math

import re

def bothsimpl(fn):

    x = 0

    while x != 1:

        y = int(input("Введіть значення експоненти: "))

        if math.gcd(y, fn) == 1:

            x = 1

        else:

            print("Число не взаємно просте з F(n)")

    return y

def generate\_keys(p, q):

    n = p \* q

    fn = (p-1)\*(q-1)

    e = bothsimpl(fn)

    for i in range(0, 1000):

        if ((e\*i) % fn) == 1:

            print("d = ", i)

            d = i

            break

    return [fn, (d, n), (e, n)]

def encrypt(public\_key):

    e, n = public\_key

    encrypt\_values = []

    print("Введіть текст, що потрібно зашифрувати:")

    text\_message = input()

    ascii\_values = [ord(character) for character in text\_message]

    print(ascii\_values)

    encrypt\_ascii\_values = [(ascii\_values[i]\*\*e) %

                            n for i in range(0, len(ascii\_values))]

    print(encrypt\_ascii\_values)

    encrypt\_values = [chr(encrypt\_ascii\_values[i])

                      for i in range(0, len(encrypt\_ascii\_values))]

    print("".join(encrypt\_values))

    return encrypt\_ascii\_values

def decrypt(private\_key):

    d, n = private\_key

    decrypt\_values = []

    print("Введіть текст, що потрібно розшифрувати:")

    encrypt\_message = input()

    ascii\_values = list(

        map(lambda x: int(x), re.findall("\d{1,10}", encrypt\_message)))

    decrypt\_ascii\_values = [(ascii\_values[i]\*\*d) %

                            n for i in range(0, len(ascii\_values))]

    print(decrypt\_ascii\_values)

    decrypt\_values = [chr(decrypt\_ascii\_values[i])

                      for i in range(0, len(decrypt\_ascii\_values))]

    print("".join(decrypt\_values))

    return decrypt\_ascii\_values

def main():

    p = int(input("Введіть просте число p: "))

    q = int(input("Введіть просте число q: "))

    fn, private\_key, public\_key = generate\_keys(p, q)

    print("----------------------------------------------------")

    print("p = ", p)

    print("q = ", q)

    print("n = ", private\_key[1])

    print("f(n) = ", fn)

    print("e = ", public\_key[0])

    print("d = ", private\_key[0])

    print("Приватний ключ: ", private\_key)

    print("Публічний ключ: ", public\_key)

    print("----------------------------------------------------")

    print("Що потрібно зробити?")

    print("1. Зашифрувати")

    print("2. Розшифрувати")

    menu\_value = input()

    if menu\_value == '1':

        encrypt(public\_key)

    elif menu\_value == '2':

        decrypt(private\_key)

main()

# Додаток 2

Скріншоти виконання кроків 1-9 *Таблиці 1.*

Зображення, що містить текст

Автоматично згенерований опис