Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

КАФЕДРА ИНФОРМАТИКИ

Отчет по лабораторной работе №3

“Асимметричная криптография. Криптосистема Рабина”

|  |  |
| --- | --- |
|  | Выполнил:  студент гр. 053504  Горожанкин В.О. |
|  | Проверил:  ассистент каф. информатики  Лещенко Евгений Александрович |

Минск 2023

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc149270766)

[1 Демонстрация работы программы 4](#_Toc149270767)

[2 Описание блок-схемы алгоритма 5](#_Toc149270768)

[Заключение 7](#_Toc149270769)

[Приложение А (обязательное) Листинг программного кода 8](#_Toc149270770)

# ВВЕДЕНИЕ

Криптография является неотъемлемой частью информационной безопасности в современном цифровом мире. Одним из ключевых аспектов криптографии является защита информации с помощью шифрования, которое позволяет передавать данные так, чтобы они были недоступны несанкционированным лицам. Асимметричная криптография представляет собой одну из наиболее важных и широко используемых техник шифрования, которая обеспечивает высокий уровень безопасности в обмене информацией.

Одним из классических примеров асимметричной криптосистемы является криптосистема Рабина, разработанная Рональдом Л. Рабином в 1979 году. Криптосистема Рабина отличается от симметричных алгоритмов шифрования тем, что в ней используются два различных ключа: открытый и закрытый. Открытый ключ используется для шифрования информации, в то время как закрытый ключ используется для расшифровки.

# 1 ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ

Входные данные записываются в файл input.txt (рисунок 1).

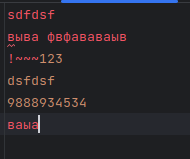


Рисунок 1 – Запись исходного текста в файл input.txt

В результате выполнения шифрованный текст сохраняется в файл encrypted.txt. Содержимое файла:

51841020111664116641001232110094091322510000940913225100001001153476114918412078011207801115347611491841024114918412078011153476114918410241024250024012601102415876518410201116641166410012321100940913225100009409132251000010011534761149184120780112078011153476114918410241149184120780111534761149184102410242500240126011024158761002025202520252025202520252025202520251024518410201116641166410012321100940913225100009409132251000010011534761149184120780112078011153476114918410241149184120780111534761149184102410242500240126011024158761002025202520252025202520252025202520251024518410201116641166410012321100940913225100009409132251000010011534761149184120780112078011153476114918410241149184120780111534761149184102410242500240126011024158761002025202520252025202520252025202520251024518410201116641166410012321100940913225100009409132251000010011534761149184120780112078011153476114918410241149184120780111534761149184102410242500240126011024158761002025202520252025202520252025202520251024518410201116641166410012321100940913225100009409132251000010011534761149184120780112078011153476114918410241149184120780111534761149184102410242500240126011024158761002025202520252025202520252025202520251322510404132251000010404100102410404100001322510404100001322510010242025202520252025202520252025202520252025202520251001024184918491849132251000010404100001322510404100115347612078011153476114918410241192464115347611924641149184115347611491841153476114918412078011153476100108915876158761587624012500260110010000132251040410000132251040410032493136313631363249260127042809260127041001153476114918412078011149184132251000010404100001322510404100115347612078011153476114918410241192464115347611924641149184115347611491841153476114918412078011153476100108915876158761587624012500260110010000132251040410000132251040410032493136313631363249260127042809260127041001153476114918412078011149184

В результате выполнения расшифрованный текст сохраняется в файл decrypted.txt (рисунок 2).

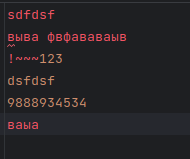


Рисунок 2 – Запись расшифрованного текста в файл decpypted.txt

# 2 ОПИСАНИЕ БЛОК-СХЕМЫ АЛГОРИТМА

Блок-схема алгоритма представлена на рисунке 3.

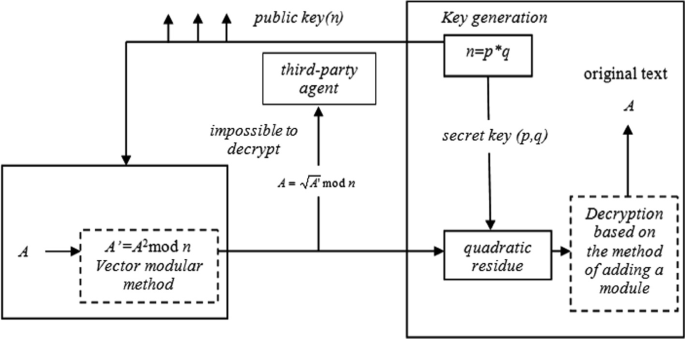


Рисунок 3 – Блок-схема работы алгоритма

.

Система Рабина, как и любая аси[мметричная криптосистема](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%BC_%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC), использует открытый и закрытый ключи. Открытый ключ используется для шифрования сообщений и может быть опубликован для всеобщего обозрения. Закрытый ключ необходим для расшифровки и должен быть известен только получателям зашифрованных сообщений.

Процесс генерации ключей следующий:

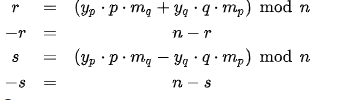
* выбираются два случайных числа *p* и *q* с учётом следующих требований:
  + числа должны быть большими;
  + числа должны быть простыми;
  + должно выполняться условие: *p* ≡ *q* ≡ 3 mod 4.

Выполнение этих требований сильно ускоряет процедуру извлечения корней по модулю *р* и *q*;

* вычисляется число *n* = *p* · *q*;
* число *n* — открытый ключ; числа *p* и *q* — закрытый.

Исходное сообщение *m* (текст) шифруется с помощью открытого ключа — числа *n* по следующей формуле: *c* = *m*² mod *n*.

Для расшифровки сообщения необходим закрытый ключ — числа *p* и *q*. Процесс расшифровки выглядит следующим образом: сначала, используя алгоритм Евклида, из уравнения . Далее, используя китайскую теорему об остатках, вычисляют четыре числа:



Одно из этих чисел является истинным открытым текстом *m*.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной лабораторной работы было реализовано программное средство шифрования и дешифрования текстовых файлов с использованием Криптосистемы Рабина. Этот процесс включал в себя несколько важных шагов, включая генерацию ключей, шифрование и последующую дешифрацию данных.

В итоге, выполнение данной лабораторной работы позволило нам приобрести практические навыки в области асимметричной криптографии и ознакомиться с принципами работы Криптосистемы Рабина. Не смотря на свои плюсы, криптосистема обладает большим количеством минусов, в связи с чем данная криптосистема на данный момент широко не применяется, в частности стоит отметить что данная криптосистема часто применяется в академических целях. Эти знания могут быть полезными при решении задач по защите конфиденциальности данных в современном информационном мире.

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Листинг программного кода**

import random

def modulo(a, b):

if a >= 0:

return a % b

else:

return (b - abs(a % b)) % b

def generate\_key(bits):

def is\_prime(n):

if n <= 1:

return False

for i in range(2, int(n \*\* 0.5) + 1):

if n % i == 0:

return False

return True

def generate\_prime(bits):

while True:

num = random.getrandbits(bits)

if num % 4 == 3 and is\_prime(num):

return num

p = generate\_prime(bits)

q = generate\_prime(bits)

open\_key = p \* q

close\_key = (p, q)

return open\_key, close\_key

def number\_to\_text(numbers):

result = []

for item in numbers:

for i in range(0, len(item), 4):

number = item[i:i + 4]

while number[0] == '0':

number = number[1:]

result.append(chr(int(number)))

return result

def extended\_gcd(a, b):

if a == 0:

return (0, 1)

else:

x, y = extended\_gcd(b % a, a)

return (y - (b // a) \* x, x)

def find\_Yp\_Yq(p, q):

x, y = extended\_gcd(p, q)

if x < 0:

x += q

Yp = x

Yq = (1 - Yp \* p) // q

return Yp, Yq

def encrypted(text, open\_key):

number = ord(text)

c = (number \*\* 2) % open\_key

return c

def mod(k, b, m):

i = 0

a = 1

v = []

while k > 0:

v.append(k % 2)

k = (k - v[i]) // 2

i += 1

for j in range(i):

if v[j] == 1:

a = (a \* b) % m

b = (b \* b) % m

else:

b = (b \* b) % m

return a

def decrypted(c, open\_key, close\_key):

p = close\_key[0]

q = close\_key[1]

# алгоритм Евклида

x, y = find\_Yp\_Yq(\*close\_key)

while x \* p + y \* q != 1:

x, y = find\_Yp\_Yq(\*close\_key)

# китайская теорема об остатках

r = mod((p+1)/4, c, p)

s = mod((q+1)/4, c, q)

r1 = (x\*p\*s + y\*q\*r) % open\_key

r2 = (open\_key - r1)

r3 = (x \* p \* s - y \* q \* r) % open\_key

r4 = (open\_key - r3)

# print(r1, r2, r3, r4)

for item in (r1, r2, r3, r4):

if item <= 1200:

return chr(item)

with open("input.txt", "r", encoding='utf-8') as f:

text = f.read()

open\_key, close\_key = generate\_key(42)

while close\_key[0] == close\_key[1]:

open\_key, close\_key = generate\_key(42)

for item in text:

encrypted\_text = encrypted(item, open\_key)

with open("encrypted.txt", "a", encoding='utf-8') as f:

f.write(str(encrypted\_text))

decrypted\_text = decrypted(encrypted\_text, open\_key, close\_key)

with open("decrypted.txt", "a", encoding='utf-8') as f:

f.write(decrypted\_text)