

Правительство Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»  
(НИУ ВШЭ)

Московский институт электроники и математики им. А.Н. Тихонова

ОТЧЕТ  
О ПРАКТИЧЕСКОЙ РАБОТЕ № 4  
по дисциплине «Математические основы защиты информации»  
Криптосистема RSA

Студент гр. БИБ201  
Д.А. Морин  
«20» 05.2022 г.

Руководитель  
Заведующий кафедрой  
информационной  
безопасности киберфизических систем  
канд. техн. наук, доцент  
О.О. Евсютин  
«\_\_» \_\_\_\_\_ 2022 г.

Москва 2022

## Содержание

1. Краткие теоретические сведения.....	3
2. “Ручное” шифрование и расшифрование.....	4
3. Программная реализация зашифрования и расшифрования.....	7
4. Криптоанализ шифров.....	10
5. Вывод.....	14
6. Список использованных источников.....	15

## **1. Краткие теоретические сведения**

Шифр - система обратимых преобразований, зависящая от некоторого секретного параметра (ключа) и предназначенная для обеспечения секретности передаваемой информации.

Ключ - это часть информации, обычно представляющая собой строку цифр или букв, хранящуюся в файле, которая при обработке с помощью криптографического алгоритма может кодировать или декодировать криптографические данные.

Шифрование (зашифрование) - обратимое преобразование информации в целях сокрытия от неавторизованных лиц с предоставлением в это же время авторизованным пользователям доступа к ней.

Расшифрование - процесс преобразования зашифрованных данных в открытые данные при помощи шифра.

Дешифрование (дешифровка) - процесс извлечения открытого текста без знания криптографического ключа на основе известного шифрованного.

Открытый текст - в криптографии исходный текст, подлежащий шифрованию, либо получившийся в результате расшифровки. Может быть прочитан без дополнительной обработки (без расшифровки).

Шифротекст, шифртекст, криптограмма - результат операции шифрования.

RSA — криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел.

Открытый (публичный) ключ применяется для шифрования информации и может передаваться по незащищенным каналам. Закрытый (приватный) ключ применяется для расшифровки данных, зашифрованных открытым ключом.

## 2. “Ручное” шифрование и расшифрование в криптосистеме RSA

Пример 1.

Для начала генерируем ключи (открытый и закрытый): возьмём 2 простых числа  $p = 103$ ,  $q = 233$ . Найдём  $n = p * q = 103 * 233 = 23999$ , функцию Эйлера  $\phi_n = (p - 1) * (q - 1) = 23664$ . Выберем число  $e$  от 1 до  $(23664 - 1)$ . Пусть  $e = 101$ . Решим уравнение и найдем секретный ключ  $d$  (с помощью расширенного алгоритма Евклида):

$$101 * d \equiv 1 \pmod{23664}$$

$d = 8669$  (решение уравнения записано в таблице 1)

Таблица 1

q	r	a	b	y2	y1
-	-	23664	101	0	1
234	30	101	30	1	-234
3	11	30	11	-234	703
2	8	11	8	703	-1640
1	3	8	3	-1640	2343
2	2	3	2	2343	-6326
1	1	2	1	-6326	8669
2	0	1	0	8669	

В итоге мы получили  $(101, 23999)$  - открытый ключ и  $8669$  - закрытый ключ.

Зашифруем сообщение, в качестве которого возьмём слово “drop”.  $drop = (3, 17, 14, 15)$ . Зашифруем каждый символ сообщения следующими действиями:

$$3^{101} \pmod{23999} = 22111$$

$$17^{101} \pmod{23999} = 22654$$

$$14^{101} \pmod{23999} = 10278$$

$$15^{101} \pmod{23999} = 17256$$

На выходе получим следующее зашифрованное сообщение:  $(22111, 22654, 10278, 17256)$

Расшифруем сообщение. Для этого каждое число из шифртекста мы преобразуем в число из открытого текста при помощи секретного ключа  $d$ :

$$22111 \wedge 8669 \bmod 23999 = 3$$

$$22654 \wedge 8669 \bmod 23999 = 17$$

$$10278 \wedge 8669 \bmod 23999 = 14$$

$$17256 \wedge 8669 \bmod 23999 = 15$$

Получили открытый текст (3, 17, 14, 15) = drop.

Пример 2.

Генерируем ключи (открытый и закрытый): возьмём 2 простых числа  $p = 37$ ,  $q = 73$ . Найдём  $n = p * q = 37 * 73 = 2701$ , функцию Эйлера  $f_n = (p-1) * (q-1) = 2592$ . Выберем число  $e$  от 1 до  $(2592 - 1)$ . Пусть  $e = 53$ . Решим уравнение и найдем секретный ключ  $d$  (с помощью расширенного алгоритма Евклида):

$$53 * d \equiv 1 \bmod 2592$$

$d = -1027 \bmod 2592 = 1565 \bmod 2592$  (решение уравнения записано в таблице 2)

Таблица 2

q	r	a	b	y2	y1
-	-	2592	53	0	1
48	48	53	48	1	-48
1	5	48	5	-48	49
9	3	5	3	49	-489
1	2	3	2	-489	538
1	1	2	1	538	-1027
		1	0	-1027	

В итоге мы получили (53, 2701) - открытый ключ и 1565 - закрытый ключ.

Зашифруем сообщение, в качестве которого возьмём слово "NICK". NICK = (13, 8, 2, 10). Зашифруем каждый символ сообщения следующими действиями:

$$13 \wedge 53 \bmod 2701 = 2237$$

$$8 \wedge 53 \bmod 2701 = 356$$

$$2 \wedge 53 \bmod 2701 = 1424$$

$$10^{53} \bmod 2701 = 63$$

На выходе получим следующее зашифрованное сообщение: (2237, 356, 1424, 63)

Расшифруем сообщение. Для этого каждое число из шифртекста мы преобразуем в число из открытого текста при помощи секретного ключа  $d$ :

$$2237^{1565} \bmod 2701 = 13$$

$$356^{1565} \bmod 2701 = 8$$

$$1424^{1565} \bmod 2701 = 2$$

$$63^{1565} \bmod 2701 = 10$$

Получили открытый текст (13, 8, 2, 10) = NICK.

### 3. Программная реализация зашифрования и расшифрования

Пример 1.

```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
1
Введите простое число p = 103
Введите простое число q = 233
Выберите число e от 1 до 23663 такое, чтобы НОД(e, 23664) = 1
e = 101
Открытый ключ: (101, 23999)
Закрытый ключ: 8669
```

Рисунок 1 - генерация ключей для шифрования

```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
2
Введите текст:
drop
Экспонента зашифрования e =
101
Модуль алгоритма n =
23999
Результат зашифрования: [22111, 22654, 10278, 17256]
```

Рисунок 2 - зашифрования открытого текста

```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
3
Введите результат зашифрованного текста через пробел:
22111 22654 10278 17256
Экспонента расшифрования d =
8669
Модуль алгоритма n =
23999
Результат расшифрования: drop
```

Рисунок 3 - расшифрование шифртекста

Пример 2.

```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
1
Введите простое число p = 37
Введите простое число q = 73
Выберите число e от 1 до 2591 такое, чтобы НОД(e, 2592) = 1
e = 53
Открытый ключ: (53, 2701)
Закрытый ключ: 1565
```

Рисунок 4 - генерация ключей для шифрования



```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
2
Введите текст: NICK
Экспонента зашифрования e = 53
Модуль алгоритма n = 2701
Результат зашифрования: [2237, 356, 1424, 63]
```

Рисунок 5 - зашифрования открытого текста

```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
3
Введите результат зашифрованного текста через пробел:
2237 356 1424 63
Экспонента расшифрования d = 1565
Модуль алгоритма n = 2701
Результат расшифрования: nick
```

Рисунок 6 - расшифрование шифртекста

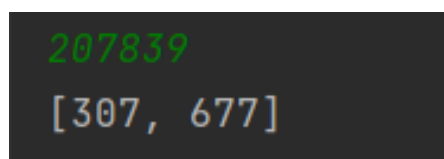
#### 4. Криптоанализ шифров

Пусть мы имеем такое зашифрованное сообщение: (13496, 22453, 56479, 137554, 104738, 71192, 137554, 137554, 0, 148667, 71192, 137554, 22453, 118226, 7708, 67188, 156885, 1, 71192, 156885, 71192, 67051, 82907, 115667, 92015, 13496, 71192, 156885)

Открытый ключ мы знаем - (997, 207839)

1) Атака с помощью математического анализа.

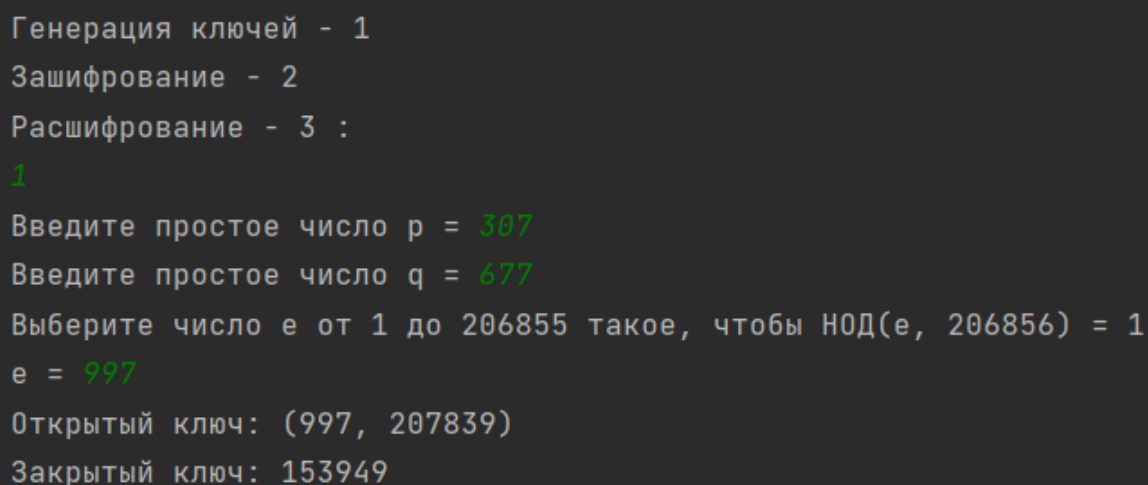
Разложим  $n$  на два простых множителя (рисунок 7)



```
207839
[307, 677]
```

Рисунок 7 - результат разложения числа на простые множители с помощью программы simple.py

Теперь мы можем расшифровать сообщение, найдя секретный ключ (рисунок 8, 9)



```
Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
1
Введите простое число p = 307
Введите простое число q = 677
Выберите число e от 1 до 206855 такое, чтобы НОД(e, 206856) = 1
e = 997
Открытый ключ: (997, 207839)
Закрытый ключ: 153949
```

Рисунок 8 - находим значение секретного ключа  $d$

```

Генерация ключей - 1
Зашифрование - 2
Расшифрование - 3 :
3
Введите результат зашифрованного текста через пробел:
13496 22453 56479 137554 104738 71192 137554 137554 0 148667 71192 137554 224
Экспонента расшифрования d = 153949
Модуль алгоритма n = 207839
Результат расшифрования: thismessageshouldbedecrypted

```

Рисунок 9 - результат дешифрования сообщения

## 2) Атака ширококвещательной передачи

Предположим, что пользователь хочет передать сообщение тремя различными способами с тем же самым общедоступным ключом  $e = 3$  и модулями  $n_1 = 13 * 17$ ,  $n_2 = 19 * 23$ ,  $n_3 = 29 * 31$ .

$$C_1 = p^3 \bmod n_1$$

$$C_2 = p^3 \bmod n_2$$

$$C_3 = p^3 \bmod n_3$$

Применяя китайскую теорему об остатках к этим трём уравнениям, можно найти уравнение формы  $C' = p^3 \bmod (n_1 * n_2 * n_3)$ . Так как  $p^3 < n_1 * n_2 * n_3$ ,  $C' = p^3$ , а  $p = C'^{(1/3)}$ .

Пусть в качестве открытого текста передадим слово hello = (7, 4, 11, 11, 14). На примере первого символа:

$$C_1 = 7^3 \bmod 221 = 122$$

$$C_2 = 7^3 \bmod 437 = 343$$

$$C_3 = 7^3 \bmod 899 = 343$$

$$N = 221 * 437 * 899$$

$$N_1 = 437 * 899 = 392863$$

$$N_2 = 221 * 899 = 198679$$

$$N_3 = 221 * 437 = 96577$$

$$C' = (122 * (-56) * 392863 + 343 * 14 * 198679 + 343 * 199 * 96577) \bmod N = 343$$

$$343^{(1/3)} = 7 = h$$

Аналогично с другими буквами. В итоге получим открытый текст.

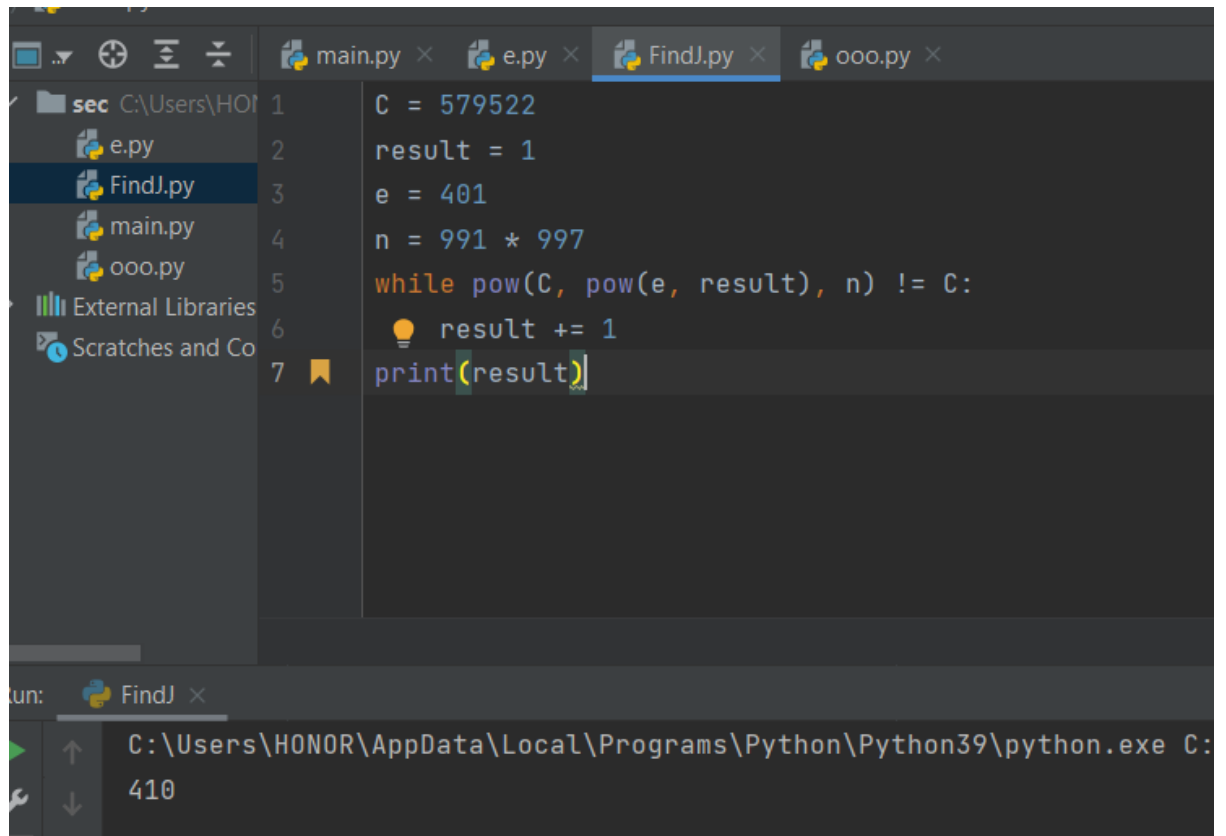
## 3) Метод бесключевого чтения RSA

Нам известен открытый ключ  $(e, n)$  и шифртекст  $C$ . Чтобы найти открытый текст, мы должны найти такое  $j$ , что  $C^{(e^j)} \bmod n = C$ . Тогда

какое-то число  $A = C^{(e^j-1)} \bmod n$  мы возводим в степень  $e$  и получаем шифртекст, поэтому  $A$  ничто иное, как открытый текст.

Например,  $(e, n) = (401, 991 * 997)$ ,  $M = 12345$ . Зашифруем сообщение:  $C = 12345^{401} \bmod (991 * 997) = 579522$ .

С помощью программы FindJ.py находим число  $j$  (рисунок 10)



The screenshot shows a Python IDE with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure with files e.py, FindJ.py, main.py, and ooo.py. The code editor shows the following code in FindJ.py:

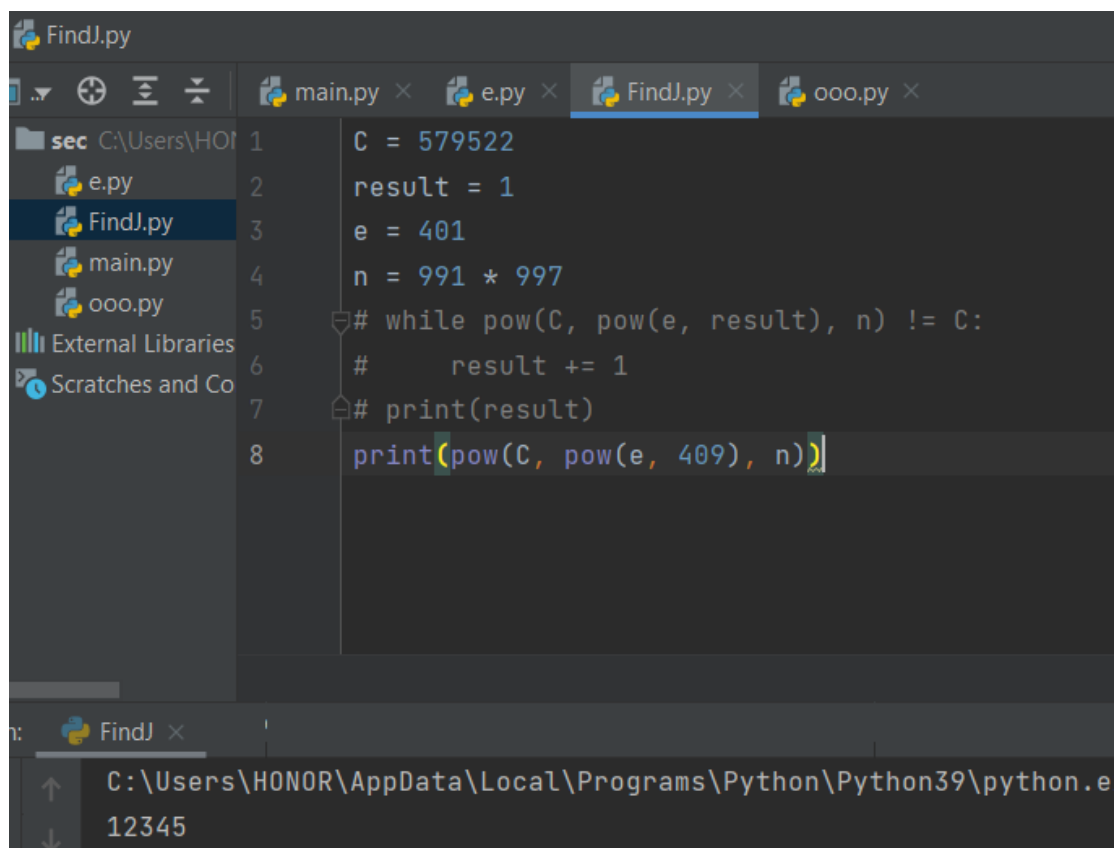
```
1 C = 579522
2 result = 1
3 e = 401
4 n = 991 * 997
5 while pow(C, pow(e, result), n) != C:
6     result += 1
7 print(result)
```

Below the code editor, the output of the program is displayed in a terminal window. It shows the command prompt path and the output value 410.

```
run: FindJ x
C:\Users\HONOR\AppData\Local\Programs\Python\Python39\python.exe C:
410
```

Рисунок 10 - результат работы программы по нахождению  $j$

Теперь находим число  $A$  (рисунок 11)



```
FindJ.py
main.py x e.py x FindJ.py x ooo.py x
sec C:\Users\HONOR\AppData\Local\Programs\Python\Python39\python.exe
e.py
FindJ.py
main.py
ooo.py
External Libraries
Scratches and Console
1 C = 579522
2 result = 1
3 e = 401
4 n = 991 * 997
5 # while pow(C, pow(e, result), n) != C:
6 #     result += 1
7 # print(result)
8 print(pow(C, pow(e, 409), n))

C:\Users\HONOR\AppData\Local\Programs\Python\Python39\python.exe
12345
```

Рисунок 11 - результат работы программы по нахождению открытого текста

Таким образом, мы получили открытый текст 12345 без секретного ключа.

## **5. Вывод**

В данной работе познакомился с основами генерацией ключей, зашифрованием и расшифрованием в RSA, проанализировал, каким образом можно взламывать такой шифр.

## **6. Список использованных источников**

1. <https://intuit.ru/studies/courses/552/408/lecture/9371?page=4>
2. [https://yandex.ru/images/search?pos=0&img\\_url=https%3A%2F%2Fstatic.rosuchebnik.ru%2Fupload%2Fiblock%2F655%2F6556eb7dfd381193b26c2b06dfbab1fb.jpg&text=Простое%20число&lr=10743&rpt=simage&source=qa](https://yandex.ru/images/search?pos=0&img_url=https%3A%2F%2Fstatic.rosuchebnik.ru%2Fupload%2Fiblock%2F655%2F6556eb7dfd381193b26c2b06dfbab1fb.jpg&text=Простое%20число&lr=10743&rpt=simage&source=qa)
3. <https://habr.com/ru/company/neobit/blog/303264/>
4. <https://algolist.manual.ru/defence/attack/rsa.php>
5. [https://yandex.ru/images/search?pos=0&img\\_url=https%3A%2F%2Fpbs.twimg.com%2Fmedia%2FFJLcp0NXsAEBeXA.jpg&text=номера%20а%20нгл%20букв&lr=10743&rpt=simage&source=serp](https://yandex.ru/images/search?pos=0&img_url=https%3A%2F%2Fpbs.twimg.com%2Fmedia%2FFJLcp0NXsAEBeXA.jpg&text=номера%20а%20нгл%20букв&lr=10743&rpt=simage&source=serp)