**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

Факультет программной инженерии и компьютерной техники

Дисциплина «Информационная безопасность»

**Лабораторная работа №2.5**

**«Шифрование открытого текста на основе эллиптических кривых»**

Вариант: 4

**Учебно-методическое пособие:** Криптографические системы с секретным и открытым

ключом: учебное пособие. / А.А. Ожиганов; УНИВЕРСИТЕТ ИТМО. — Санкт-Петербург, 2015

**Автор**: Калинин Даниил Дмитриевич

**Группа**: P34141

**Преподаватель**: Маркина Татьяна Анатольевна

г. Санкт-Петербург

2024

# Содержание

[**Содержание 2**](#_1j29sb1plksc)

[**Цель работы 2**](#_ic6vz9o4aev6)

[**Порядок выполнения работы 2**](#_7c966s7zbuo)

[**Вариант 2**](#_tk58wqserqjy)

[**Выполнение работы 3**](#_i73wvolbwe0b)

[Код 3](#_opx7en3yhfv9)

[Результаты работы программы 5](#_h1i0yqxqtni2)

[**Вывод 6**](#_uft2d2ek0ay3)

# Цель работы

Зашифровать открытый текст, используя алфавит, приведенный в [4], в подразделе «Задачи к лабораторным работам по криптографии на эллиптических кривых (используется кривая E751( -1,1) − и генерирующая точка G = (0, 1))».

# Порядок выполнения работы

* Ознакомьтесь с теорией в учебном пособии «Криптография», а также в учебно-методическом пособии к выполнению лабораторного практикума по дисциплине «Криптография;
* Получите вариант задания у преподавателя;
* Зашифруйте открытый текст;
* Результаты и промежуточные вычисления оформить в виде отчета.

# Вариант

| **№ варианта** | **Открытый текст** | **Открытый ключ *B*** | **Значения случайных чисел *k* для букв открытого текста** |
| --- | --- | --- | --- |
| 4 | симметрия | (179, 275) | 11, 17, 18, 19, 16, 6, 12, 8, 2 |

# Выполнение работы

## Код

import csv

def read\_alphabet():

alphabet = []

with open("../resources/alphabet.csv", encoding='utf-8') as r\_file:

file\_reader = csv.reader(r\_file, delimiter=",")

count = 0

for row in file\_reader:

if count != 0:

alphabet.append([int(row[0]), row[1], int(row[2]), int(row[3])])

count += 1

return alphabet

def find\_symbol\_point\_in\_alphabet(alphabet, symbol):

for row in alphabet:

if row[1] == symbol:

return row[2], row[3]

print(f'Ошибка: символ {symbol} не найден в исходном алфовите')

exit(1)

def sum\_points\_elliptic\_curve(p, E, P, Q):

*# Вычисляем лямбда*

l = 0

if P[0] == Q[0] and P[1] == Q[1]:

l\_top = (3\*P[0]\*\*2 + E[0]) % p

l\_bottom = (2\*P[1]) % p

for res in range(p+1):

if (res == p):

print("Ошибка: невозможно найти модуль от деления в процессе вычисления lambda")

exit(1)

if ((l\_bottom \* res) % p) == (l\_top % p):

l = res

break

else:

l\_top = (Q[1] - P[1]) % p

l\_bottom = (Q[0] - P[0]) % p

for res in range(p + 1):

if (res == p):

print("Ошибка: невозможно найти модуль от деления в процессе вычисления lambda")

exit(1)

if ((l\_bottom \* res) % p) == (l\_top % p):

l = res

break

if (l % 1 == 0):

l = int(l)

else:

print(f'Ошибка: lambda = {l}, не целое цисло!')

exit(1)

*#print(f'lambda = {l}')*

*# Вычисляем координаты*

x = (pow(l, 2, p) - P[0] - Q[0]) % p

y = (l\*(P[0] - x) - P[1]) % p

return [x, y]

def calc\_k\_multiply\_point(p, E, k, Point, point\_name):

iPoint = Point

for i in range(2, k+1):

iPoint = sum\_points\_elliptic\_curve(p, E, iPoint, Point)

*#print(f'{i}{point\_name} = {iPoint}')*

return iPoint

if \_\_name\_\_ == '\_\_main\_\_':

*# Константы*

p = 751

E = [-1, 1]

G = [0, 1]

alphabet = read\_alphabet()

*# Описание варианта*

open\_message = "симметрия"

Pb = [179, 275]

k\_array\_A = [11, 17, 18, 19, 16, 6, 12, 8, 2]

print("-- Исходные данные --")

print(f'Открытый текст = {open\_message}')

print(f'Открытый ключ B = {Pb}')

print(f'Значения случайных чисел k для букв открытого текста = {k\_array\_A}')

print()

*# Проверяем корректность введенных данных*

if len(open\_message) != len(k\_array\_A):

print("Ошибка: длина массива k и открытого текста должна быть равной")

exit(1)

*# Шифруем сообщение*

print("-- Шифрование --")

close\_message = []

for i in range(len(open\_message)):

symbol = open\_message[i]

k = k\_array\_A[i]

print(f'Шифруем символ \'{symbol}\', k = {k}')

*# Код символа из алфавита*

Pm = find\_symbol\_point\_in\_alphabet(alphabet, symbol)

print(f'Pm = {Pm}')

*# Открытый ключ A*

kG = calc\_k\_multiply\_point(p, E, k, G, "G")

print(f'{k}G = {kG}')

*# Шифрование символа*

kPb = calc\_k\_multiply\_point(p, E, k, Pb, "Pb")

print(f'{k}Pb = {kPb}')

Cm = sum\_points\_elliptic\_curve(p, E, Pm, kPb)

print(f'Cm = Pm + kPb = {Pm} + {kPb} = {Cm}')

print()

close\_message.append((kG, Cm))

print("-- Результат --")

for i in range(len(open\_message)):

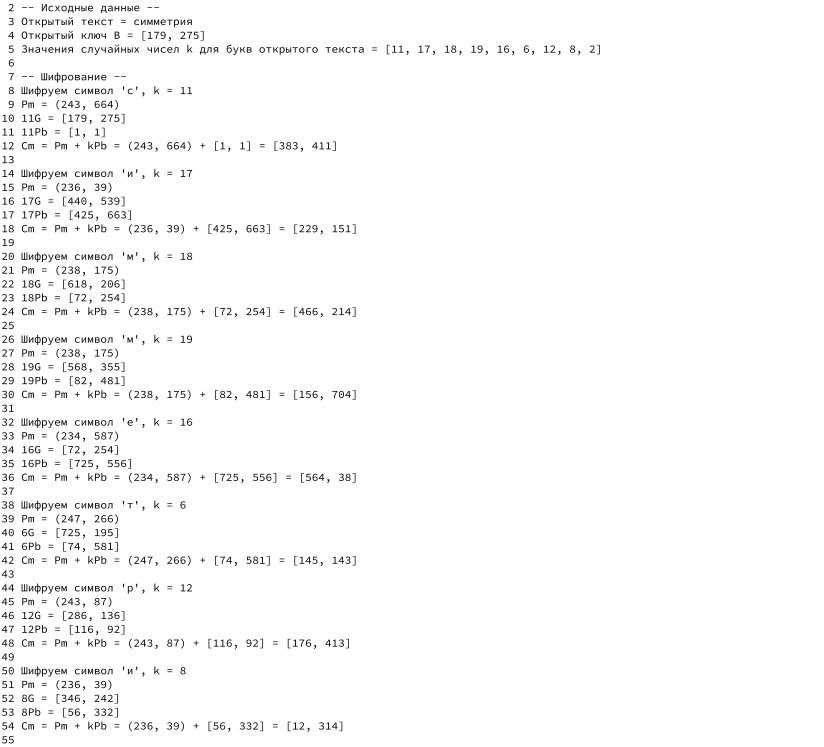
symbol = open\_message[i]

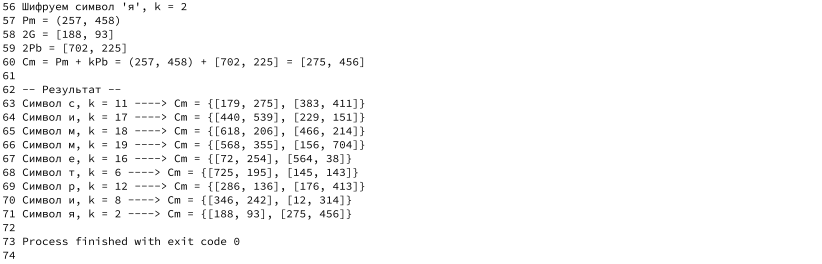
k = k\_array\_A[i]

tmp = "{" + str(close\_message[i][0]) + ", " + str(close\_message[i][1]) + "}"

print(f'Символ {symbol}, k = {k} ----> Cm = {tmp}')

## Результаты работы программы





# Вывод

В ходе лабораторной работы был зашифрован открытый текст, согласно алгоритму, приведенному в [4], в подразделе «Задачи к лабораторным работам по криптографии на эллиптических кривых (используется кривая E751( -1,1) − и генерирующая точка G = (0, 1))». Также в процессе работы был изучен метод шифрования текста на основе эллиптических кривых.