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

Факультет программной инженерии и компьютерной техники

Дисциплина «Информационная безопасность»

**Лабораторная работа №2.6**

**«Расшифрование криптограммы 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

Дан шифртекст, 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))» и зная секретный ключ nb, найти открытый текст

# 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;
* Найдите открытый текст;
* Результаты и промежуточные вычисления оформите в виде отчета

# Вариант

|  |  |  |
| --- | --- | --- |
| **№ варианта** | **Секретный ключ** | **Шифртекст** |
| 4 | 34 | {(618, 206), (426, 662)}; {(72, 254), (67, 667)};  {(286, 136), (739, 574)}; {(16, 416), (143, 602)};  {(618, 206), (313, 203)}; {(618, 206), (114, 607)};  {(618, 206), (438, 711)}; {(188, 93), (573, 168)} |

# 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\_by\_point\_in\_alphabet(alphabet, point):

for row in alphabet:

if int(row[2]) == point[0] and int(row[3]) == point[1]:

return row[1]

print(f'Ошибка: точка {point} не найдена 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*

close\_message\_text = """

{(618, 206), (426, 662)}; {(72, 254), (67, 667)};

{(286, 136), (739, 574)}; {(16, 416), (143, 602)};

{(618, 206), (313, 203)}; {(618, 206), (114, 607)};

{(618, 206), (438, 711)}; {(188, 93), (573, 168)}

"""

nb = 34

*# Обработка шифртекста*

close\_message = []

for part in close\_message\_text.split(";"):

part = part.strip().replace("{", "").replace("}","").replace("(","").replace(")","")

part\_split = part.split(", ")

close\_message.append(([int(part\_split[0]), int(part\_split[1])], [int(part\_split[2]), int(part\_split[3])]))

print("-- Исходные данные --")

print(f'Шифртекст = {close\_message}')

print(f'Закрытый ключ B = {nb}')

print()

*#Дешифрование сообщения*

print("-- Дешифрование --")

open\_message = ""

Pm\_array = []

for kG, Cm in close\_message:

print(f'Дешифруем часть шифртекста {Cm}, kG = {kG}')

nbkG = calc\_k\_multiply\_point(p, E, nb, kG, "\*kG")

print(f'{nb}\*kG = {nbkG}')

nbkG[1] = p - nbkG[1]

print(f'-{nb}\*kG = {nbkG}')

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

print(f'Pm = Cm - nb(kG) = {Cm} + {nbkG} = {Pm}')

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

print(f'Символ {Pm} = \'{symbol}\'')

open\_message += symbol

Pm\_array.append(Pm)

print()

print("-- Результат --")

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

kG = close\_message[i][0]

Cm = close\_message[i][1]

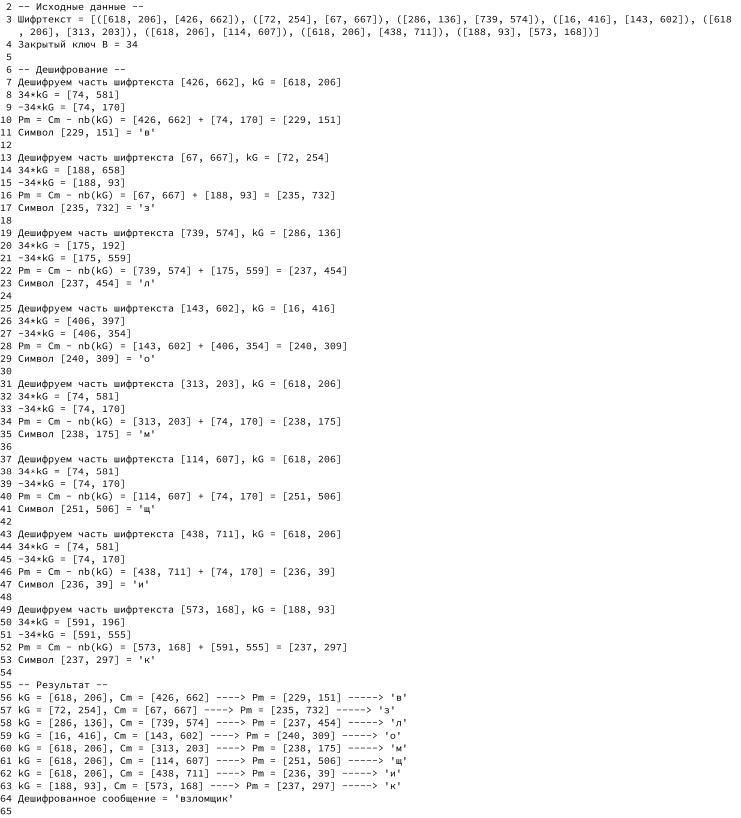
Pm = Pm\_array[i]

symbol = open\_message[i]

print(f'kG = {kG}, Cm = {Cm} ----> Pm = {Pm} -----> \'{symbol}\'')

print(f"Дешифрованное сообщение = \'{open\_message}\'")

## Результаты работы программы



# Вывод

В ходе лабораторной работы был дешифрован открытый текст по известному шифртексту и секретному ключу nb согласно алфавиту, 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 дешифрования открытого текста из шифртекста и закрытого ключа на основе эллиптических кривых.