**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии   
и прикладная математика»   
Кафедра: 806 «Вычислительная математика   
и программирование»**

Лабораторная работа № 5   
по курсу «Криптография»

Группа: М8О-310Б-21

Студент: А. С. Личковаха

Преподаватель: А. В. Борисов

Оценка:

Дата: 06.06.2024

Москва, 2024

**ОГЛАВЛЕНИЕ**

[1](#_gjdgxs) Тема 3

[2](#_30j0zll) Задание 3

[3](#_1fob9te) Теория 4

[4](#_3znysh7) Ход лабораторной работы 5

[5](#_2et92p0) Выводы 7

# **Тема**

Эллиптические кривые

# **Задание**

Подобрать такую эллиптическую кривую, порядок точки которой полным перебором находится за 10 минут на ПК. Упомянуть в отчёте результаты замеров работы программы, характеристики вычислителя. Также указать какие алгоритмы и/или теоремы существуют для облегчения и ускорения решения задачи полного перебора.

Рассмотреть для случая конечного простого поля Z\_p.

# **Теория**

Эллиптические кривые — это набор точек, удовлетворяющих уравнению вида y^2=x^3+ax+b, где a и b - константы. Эти кривые получили свое название из-за своего отношения к эллипсам. Они имеют ряд уникальных свойств, которые делают их полезными в различных областях математики, включая криптографию.

Эллиптические кривые используются в криптографии для создания систем с публичным ключом. Они обеспечивают высокий уровень безопасности при использовании меньших ключей по сравнению с традиционными методами, такими как RSA. Это делает их более эффективными с точки зрения производительности и потребления памяти.

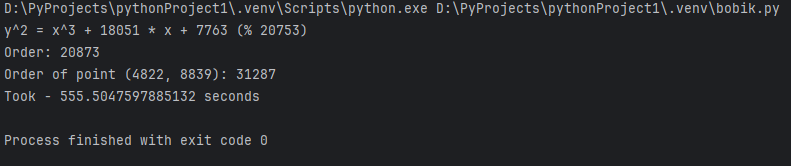
Одной из основных операций, используемых в эллиптической криптографии, является операция сложения точек на кривой. Эта операция, вместе с свойством дискретного логарифма, обеспечивает основу для создания криптографических систем на основе эллиптических кривых.

# **Ход лабораторной работы**

Для выполнения лабораторной работы я написал программу на python. Коэффициенты A, B были выбраны случайно, а параметр p (простое число) – был подобран экспериментально.

Получившиеся значения:

A = 83756655655161651589498758697  
 B = 20930197854613279396529204242  
 p = 20753

При таких условиях, решение отрабатывает на моей системе ( i7-6700HQ) за 555 секунд:

В ходе работы программа полным перебором находит все точки кривой, рассматривается порядок случайной точки.

Задачу полного перебора, очевидно, можно ускорить. Например, для этого широко используется алгоритм Шуфа. Данный алгоритм работает гораздо быстрее (сравнение сложностей – O(p^2) у полного перебора против O(log^8(n) (n число элементов поля) у алгоритма Шуфа).

Код программы:

import random  
import time  
  
  
def is\_elliptic\_curve(x, y, p):  
 return (y \*\* 2) % p == (x \*\* 3 + (A % p) \* x + (B % p)) % p  
  
def extended\_euclidean\_algorithm(a, b):  
 s, old\_s = 0, 1  
 t, old\_t = 1, 0  
 r, old\_r = b, a  
  
 while r != 0:  
 quotient = old\_r // r  
 old\_r, r = r, old\_r - quotient \* r  
 old\_s, s = s, old\_s - quotient \* s  
 old\_t, t = t, old\_t - quotient \* t  
  
 return old\_r, old\_s, old\_t  
  
  
def inverse\_of(n, p):  
 gcd, x, y = extended\_euclidean\_algorithm(n, p)  
 assert (n \* x + p \* y) % p == gcd  
 return x % p  
  
def order\_point(point, p):  
 i = 1  
 check = add\_points(point, point, p)  
 while check != (0, 0):  
 check = add\_points(check, point, p)  
 i += 1  
 return i  
  
def add\_points(p1, p2, p):  
 if p1 == (0, 0):  
 return p2  
 elif p2 == (0, 0):  
 return p1  
 elif p1[0] == p2[0] and p1[1] != p2[1]:  
 return (0, 0)  
  
 if p1 == p2:  
 s = ((3 \* p1[0] \*\* 2 + (A % p)) \* inverse\_of(2 \* p1[1], p)) % p  
 else:  
 s = ((p1[1] - p2[1]) \* inverse\_of(p1[0] - p2[0], p)) % p  
 x = (s \*\* 2 - 2 \* p1[0]) % p  
 y = (p1[1] + s \* (x - p1[0])) % p  
 return (x, -y % p)  
  
  
  
  
  
A = 85154655655161651535498798898  
B = 10431197854613279813146972163  
p = 26849

print("y^2 = x^3 + {0} \* x + {1} (% {2})".format(A % p, B % p, p))  
points = []  
start = time.time()  
for x in range(0, p):  
 for y in range(0, p):  
 if is\_elliptic\_curve(x, y, p):  
 points.append((x, y))  
   
end = time.time()  
print("Order: {0}".format(len(points)))  
point = random.choice(points)  
print("Order of point {0}: {1}".format(point, order\_point(point, p)))  
print("Took - {} seconds".format(end - start))

# **Выводы**

В ходе данной работы я познакомился с понятием эллиптических кривых. Мною была написана программа, которая за 10 минут путем полного перебора находит порядок точки некой эллиптической кривой. Были проанализированы альтернативные алгоритмы, которые помогают значительно ускорить данный процесс.

# **Список используемой литературы**

1. <https://habr.com/ru/articles/335906/>
2. <https://en.wikipedia.org/wiki/Elliptic-curve_cryptography>
3. https://crypto-kantiana.com/semyon.novoselov/talks/2023-ecc-mephi.pdf