# Лабораторная работа 6

## Разложение чисел на множители

Пологов Владислав Александрович

2022 Москва

RUDN University, Moscow, Russian Federation

# Цель работы

# Цель работы

Реализовать алгоритм, реализующий p-метод Полларда

# Описание реализации

# Описание реализации

Для реализации алгоритмов использовались средства языка Python.

# Реализация

# Алгоритм, реализующий p-метод Полларда

Итак, мы хотим факторизовать число n. Предположим, что n = pq и p ≈ q. Понятно, что труднее случая, наверное, нет. Алгоритм итеративно ищет наименьший делитель и таким образом сводит задачу к как минимум в два раза меньшей. Алгоиртм, реализующий p-метод Полларда приведён на рисунке 1. (рис. -fig. **??**)

# Алгоритм, реализующий p-метод Полларда

1. Положить $a \leftarrow c, b \leftarrow c$.
2. Вычислить $a \leftarrow f(a)(mod\ n), b \leftarrow f(b)\ (mod\ n)$
3. Найти $d \leftarrow$ НОД$(a - b, n)$.
4. Если $1 < d < n$, то положить $p \leftarrow d$ и результат: $p$. При $d = n$ результат: «Делитель не найден»; при $d = 1$ вернуться на шаг 2.

**Figure 1:** Алгоритм, реализующий p-метод Полларда

# Алгоритм, реализующий p-метод Полларда

Возьмём произвольную «достаточно случайную» с точки зрения теории чисел функцию. Например f(x) = (x + 1) ^2 mod n.

Граф, в котором из каждой вершины есть единственное ребро , называется функциональным. Если в нём нарисовать «траекторию» произвольного элемента — какой-то путь, превращающийся в цикл — то получится что-то похожее на букву (ро). Алгоритм из-за этого так и назван. Траектория произвольного элемента представлена на рисунке 2. (рис. -fig. **??**)

# Траектория произвольного элемента

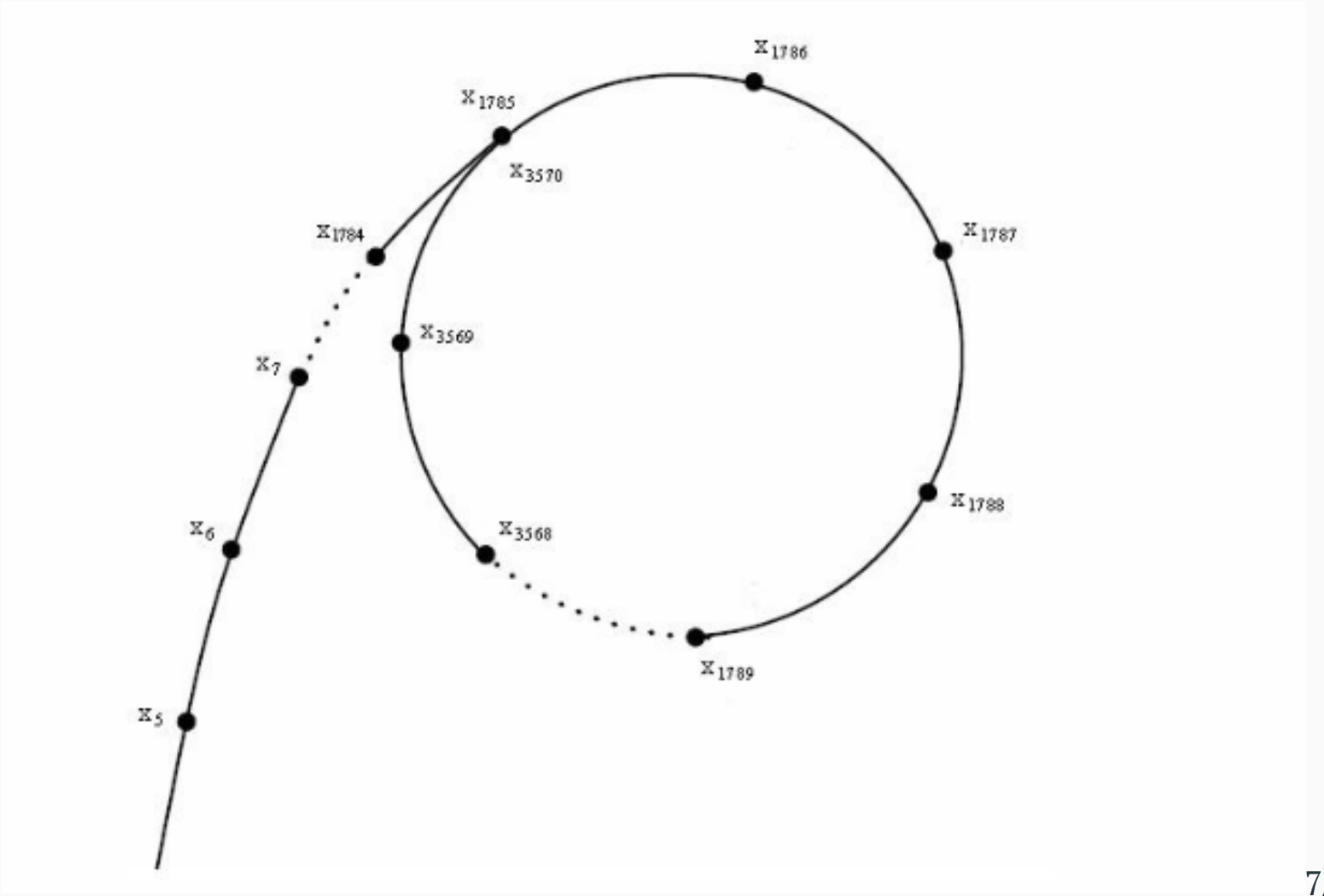




**Figure 2:** Траектория произвольного элемента

# Код, реализующий алгоритм

Использовались библиотеки math для вычисления НОД и randint для получения целого рандомного числа. Код, реализующий p-метод Полларда представлен на рисунке 3. (рис. -fig. **??**)

# Код, реализующий алгоритм

```python
import math
from random import randint

def pollard(n: int) -> int:
    f = lambda x: (x**2 + 1) % n
    c = randint(0, n - 1)
    a = c
    b = c

    while True:
        a = f(a)
        b = f(f(b))
        d = math.gcd(a - b, n)
        if 1 < d < n:
            return d
        elif d == n:
            return None


for i in range(2000, 3000):
    p = pollard(i)
    if p: print(i, p, i // p)
```

# Вывод

# Вывод

- Реализован программно p-метод Полларда. Проведена проверка методом квадратов.