Лабораторная работа 7

Дискретное логарифмирование в конечном поле

Греков Максим Сергеевич

Содержание

# Цель работы

* Ознакомиться с задачей дискретного логарифмирования в конечном поле
* Рассмотреть теоретические основы представленного алгоритма
* Реализовать р-Метод Полларда для задач дискретного логарифмирования

# Описание задачи

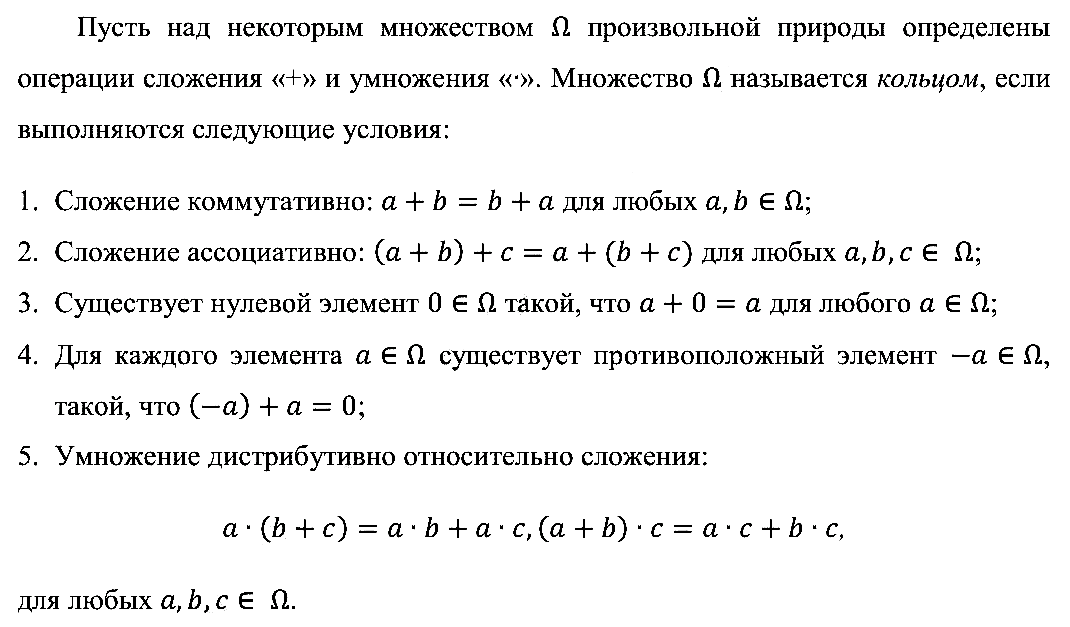
## Введение

Задача дискретного логарифмирования, как и задача разложения на множители, применяется во многих алгоритмах криптографии с открытым ключом.

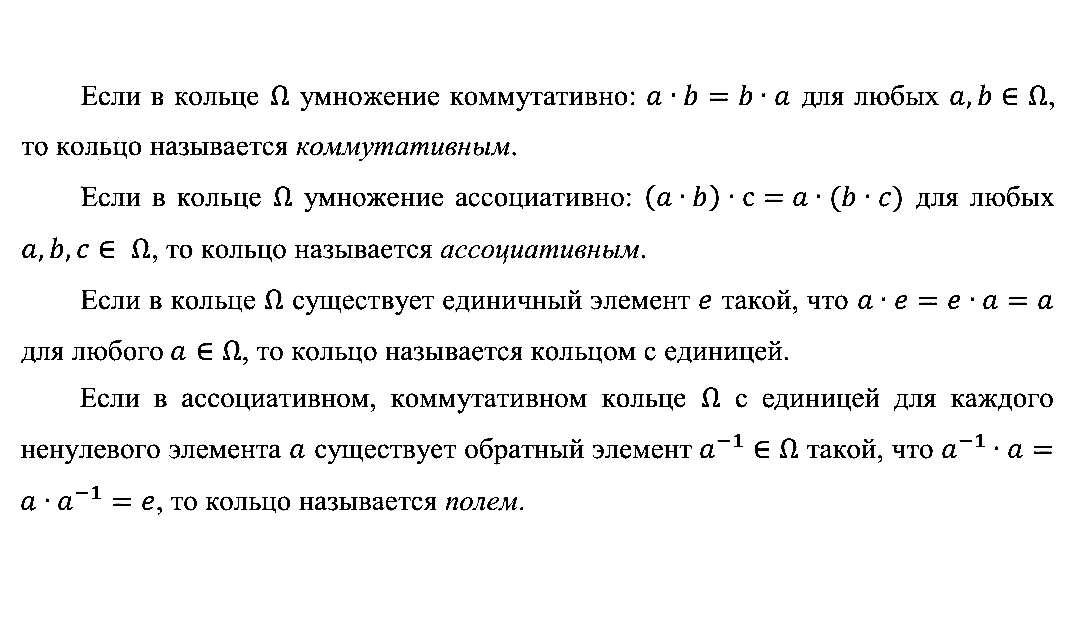
Предложена в 1976 году У. Диффи и М. Хеллманом для установления сеансового ключа.

Эта задача послужила основой для создания протоколов шифрования и цифровой подписи, доказательств с нулевым разглашением и других криптографических протоколов.

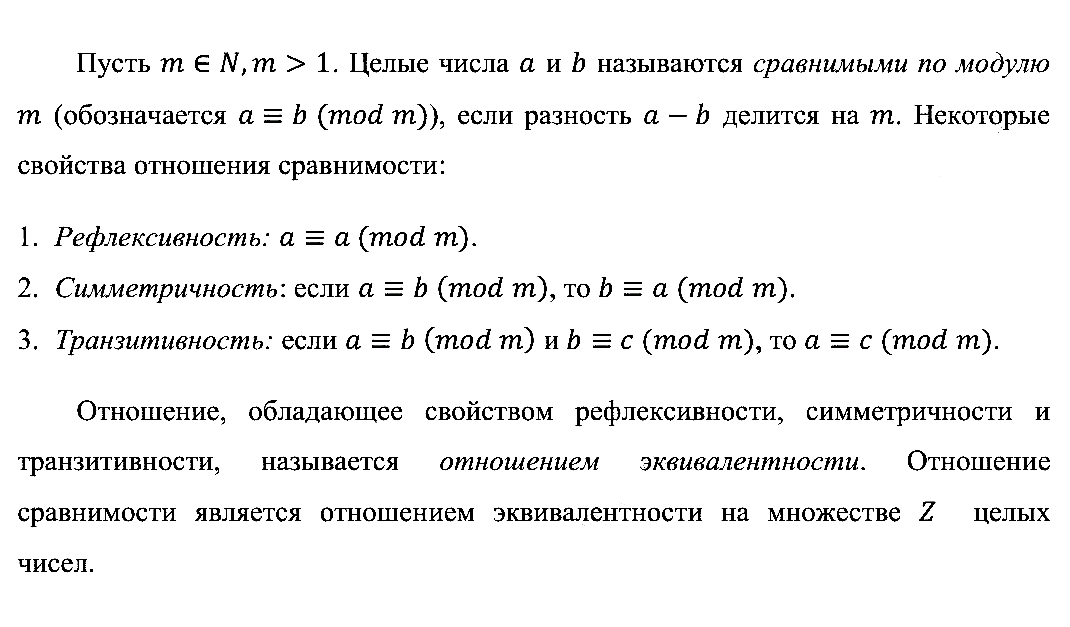
## Условия кольца



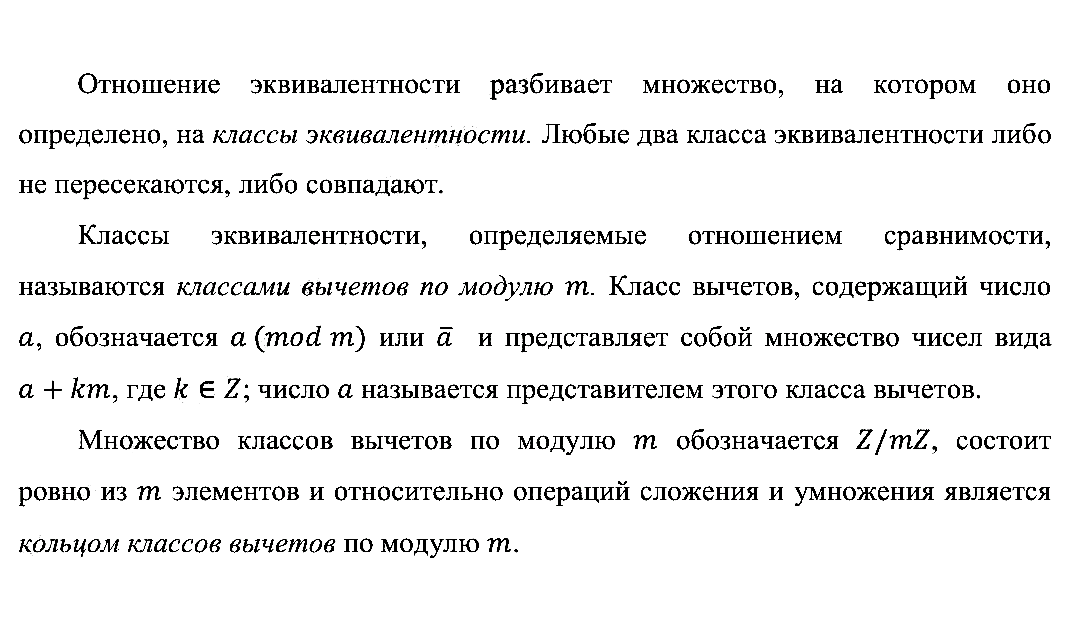
## Свойства кольца



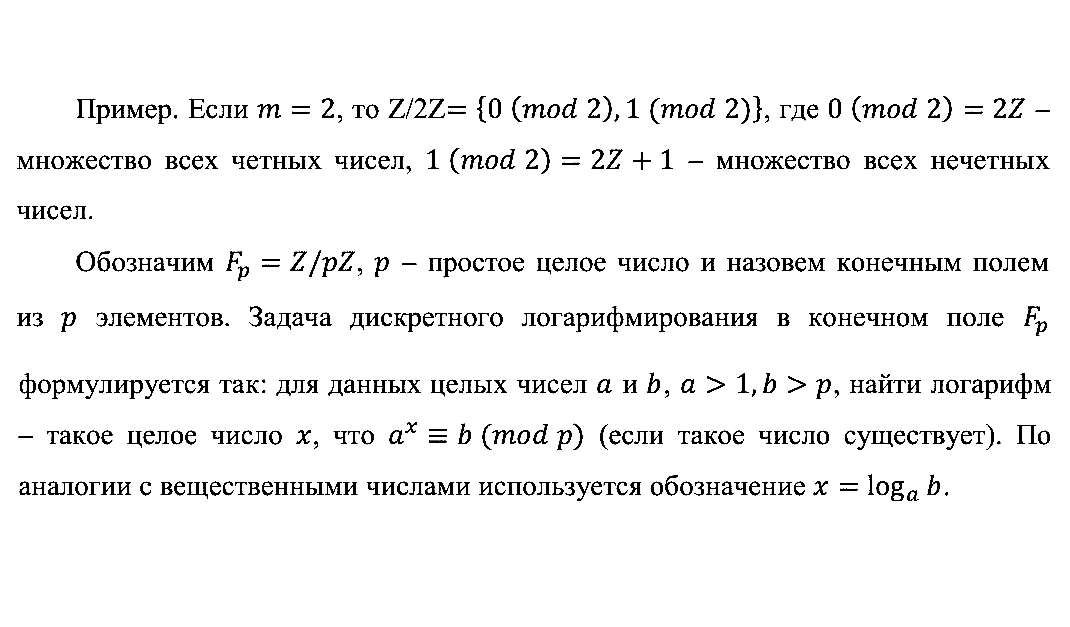
## Свойства отношения сравнимости



## Классы эквивалентности

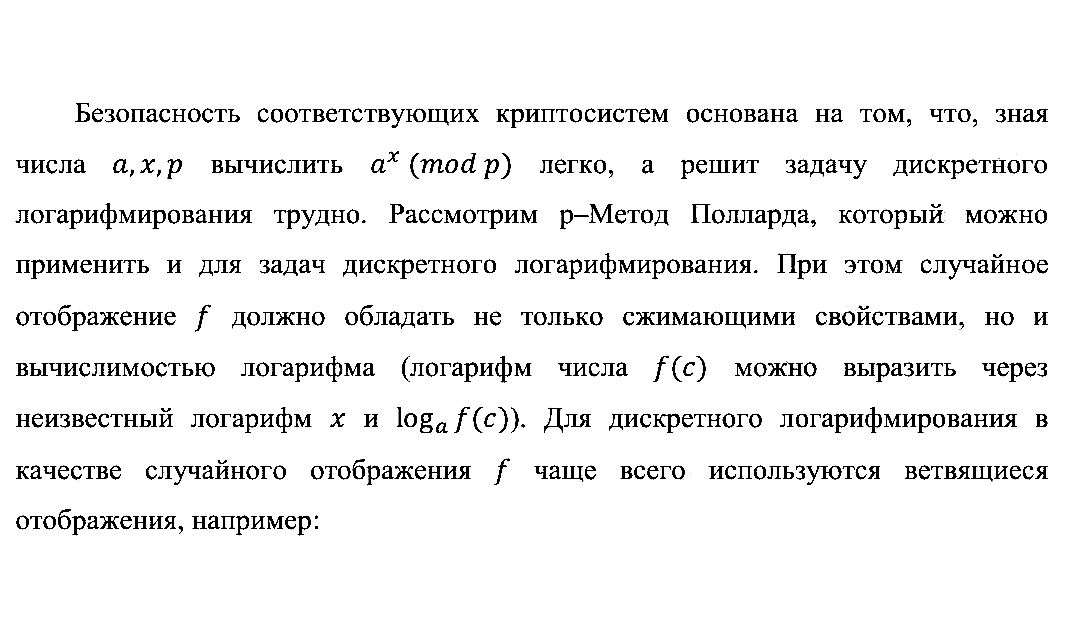


## Постановка задачи

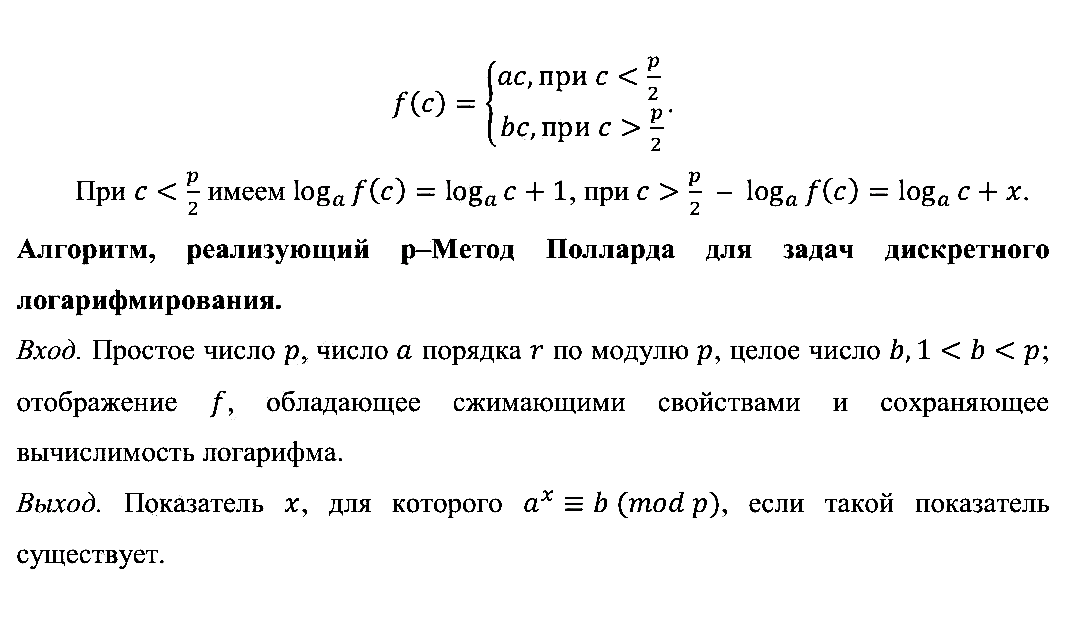


# р-Метод Полларда

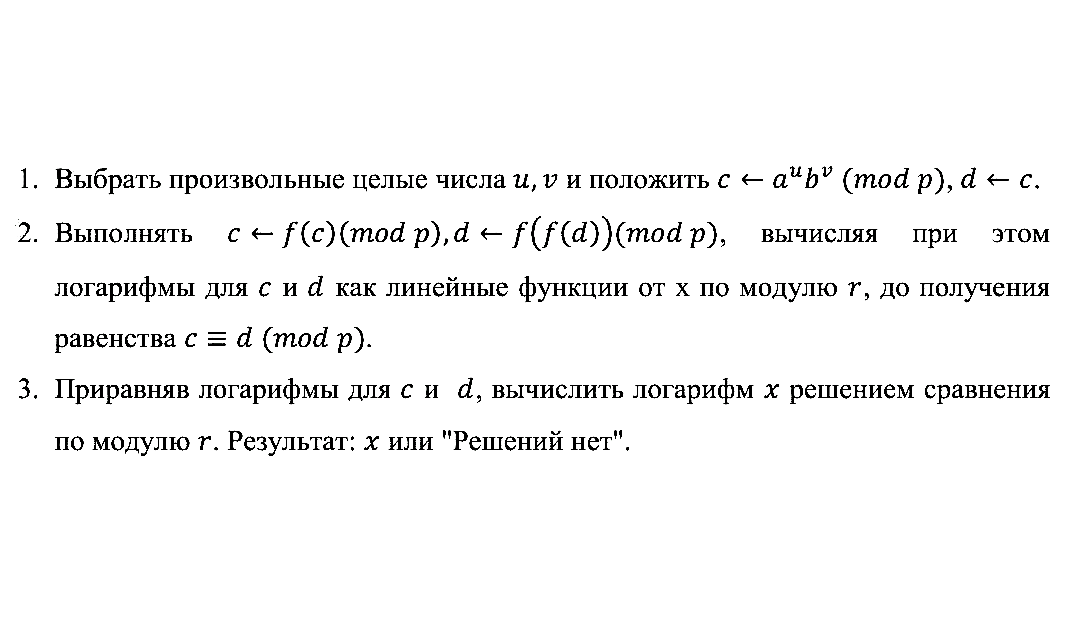
## Отображение f()



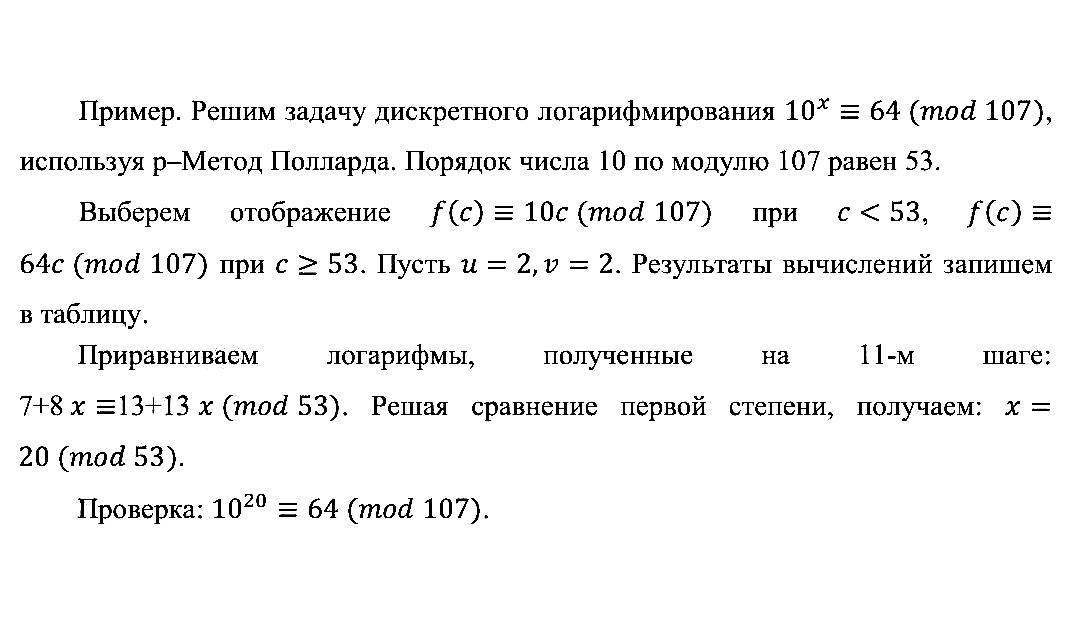
## Описание алгоритма



## Последовательность вычислений



## Пример



# Реализация алгоритма

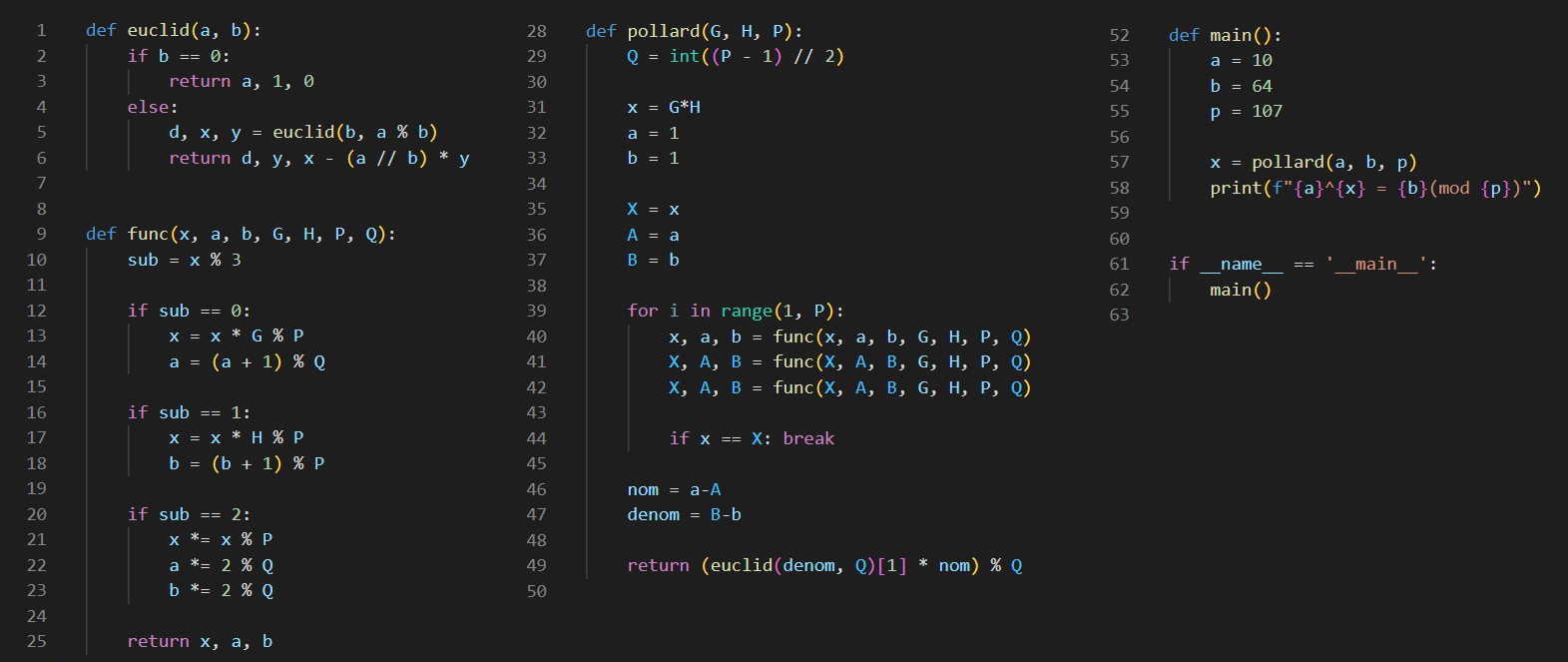


Figure 1: Код на языке Python

## Результат работы

Результат работы данного алгоритма:

## Программный код

def euclid(a, b):  
 if b == 0:  
 return a, 1, 0  
 else:  
 d, x, y = euclid(b, a % b)  
 return d, y, x - (a // b) \* y  
  
  
def func(x, a, b, G, H, P, Q):  
 sub = x % 3  
  
 if sub == 0:  
 x = x \* G % P  
 a = (a + 1) % Q  
  
 if sub == 1:  
 x = x \* H % P  
 b = (b + 1) % P  
  
 if sub == 2:  
 x \*= x % P  
 a \*= 2 % Q  
 b \*= 2 % Q  
  
 return x, a, b  
  
  
def pollard(G, H, P):  
 Q = int((P - 1) // 2)  
  
 x = G\*H  
 a = 1  
 b = 1  
  
 X = x  
 A = a  
 B = b  
  
 for i in range(1, P):  
 x, a, b = func(x, a, b, G, H, P, Q)  
 X, A, B = func(X, A, B, G, H, P, Q)  
 X, A, B = func(X, A, B, G, H, P, Q)  
  
 if x == X: break  
  
 nom = a-A  
 denom = B-b  
  
 return (euclid(denom, Q)[1] \* nom) % Q  
  
  
def main():  
 a = 10  
 b = 64  
 p = 107  
  
 x = pollard(a, b, p)  
 print(f"{a}^{x} = {b}(mod {p})")  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

# Выводы

* Ознакомились с задачей дискретного логарифмирования в конечном поле
* Рассмотрели теоретические основы представленного алгоритма
* Реализовали р-Метод Полларда для задач дискретного логарифмирования