**Алгоритм Ель-Гамаля** лаб1 крипт

Алгоритм Ель-Гамаля є криптографічним алгоритмом з відкритим ключем, який використовується для шифрування та цифрового підпису повідомлень.

Основною ідеєю алгоритму є використання дискретних логарифмів для забезпечення безпеки. Алгоритм використовує циклічні групи підгруп простого поля, що дозволяє надзвичайно складні обчислення дискретних логарифмів.

Данний алгоритм відомий своєю стійкістю до атак на базі кількості, але вимагає великих обчислювальних витрат. Також він вразливий до атак на базі часу, якщо реалізований неоптимально. Однак при правильній імплементації та використанні алгоритм надає надійний рівень безпеки.

Основні кроки алгоритму Ель-Гамаля:

* **Генерація ключів**: Користувач генерує випадкове просте число *p* та вибирає примітивний корінь *g* по модулю *p*. Далі вибирається випадкове ціле число *x* як приватний ключ, та обчислюється ℎ=mod  *h*=*g:x* mod*p*, що стає публічним ключем.
* **Шифрування**: Для кожного повідомлення *m* користувач обирає випадкове ціле число *y*, яке називається випадковим секретом. Потім обчислюється *c*1​=*gy*mod*p* та *c*2​=*m*⋅*hy*mod*p*, що утворює шифртекст (*c*1​,*c*2​).
* **Дешифрування**: Для отримання початкового повідомлення, отримувач використовує свій приватний ключ *x* для обчислення *hy*mod*p*, після чого множить другий компонент шифртексту на обернене значення цього числа за модулем *p*.

**Опис коду:**

**modinv(a, m)**: Ця функція обчислює обернене модулярне значення *a* за модулем *m* за допомогою розширеного алгоритму Евкліда.

**generate\_large\_prime(bit\_length)**: Ця функція генерує випадкове просте число заданої бітової довжини. Вона використовує функцію **is\_prime(n)** для перевірки, чи є число простим.

**is\_prime(n, k=5)**: Ця функція перевіряє, чи є число простим, за допомогою тесту Міллера-Рабіна.

**generate\_keypair(bit\_length)**: Ця функція генерує пару ключів для алгоритму Ель-Гамаля. Вона генерує випадкове просте число *p*, випадкове число *g* в межах від 2 до *p*−1 та випадкове число *x*, яке стає приватним ключем. Публічний ключ обчислюється як *h*=*g^x*mod*p*.

**encrypt(plaintext, public\_key)**: Ця функція шифрує вхідне повідомлення за допомогою публічного ключа. Вона генерує випадкове число *y*, обчислює *c*1​=*g^y*mod*p*, *s*=*h^y*mod*p* та *c*2​=(*plaintext*⋅*s*)mod*p*, що утворює шифртекст.

**decrypt(ciphertext, public\_key, private\_key)**: Ця функція дешифрує шифртекст за допомогою приватного ключа та публічного ключа. Вона обчислює *s*=*c*1*x*​mod*p* та використовує обернене модулярне значення для відновлення початкового повідомлення.

**Основний блок коду**: В цьому блоку генеруються ключі, шифрується вхідне повідомлення, дешифрується шифртекст та виводиться результат.