**1. Основные термины и понятия криптографии**

**Криптология** (kryptos - тайный, logos - наука) - наука, занимающаяся проблемой защиты информации путем ее преобразования. Разделяется на два направления:

* **Криптография** - занимается поиском и исследованием математических методов преобразования информации
* **Криптоанализ** - исследует возможности расшифровывания информации без знания ключей

**Основные свойства информации:**

* **Конфиденциальность** - субъективно определяемая характеристика информации, составляющей коммерческую или личную тайну
* **Целостность** - свойство информации существовать в неискаженном виде
* **Доступность** - свойство информации, обеспечивающее беспрепятственный доступ к ней для санкционированных операций

**Криптографические понятия:**

* **Алгоритм шифрования** - известный набор действий, необходимый для шифрования/расшифрования данных
* **Ключ** - конкретное секретное состояние параметров алгоритма криптографического преобразования
* **Шифр** - совокупность обратимых преобразований множества открытых данных на множество зашифрованных данных
* **Гамма шифра** - псевдослучайная двоичная последовательность для зашифровывания информации
* **Гаммирование** - процесс наложения гаммы шифра на открытые данные
* **Имитовставка** - отрезок информации фиксированной длины для обеспечения имитозащиты
* **Криптостойкость** - характеристика шифра, определяющая его стойкость к дешифрованию без знания ключа

**2. Тенденции развития криптографии как науки**

**Основные тенденции:**

**Переход от классической к современной криптографии:**

* Развитие от простых методов замены и перестановки к сложным математическим преобразованиям
* Использование теории чисел, алгебры, теории групп и полей

**Появление асимметричной криптографии:**

* Революционный переход от симметричных к асимметричным системам
* Решение проблемы распределения ключей
* Создание систем электронной подписи

**Квантовая криптография:**

* Использование квантовых свойств для обеспечения безопасности
* Квантовое распределение ключей
* Подготовка к эре квантовых компьютеров

**Криптография на эллиптических кривых:**

* Повышение эффективности при меньших размерах ключей
* Лучшее соотношение безопасность/производительность

**Постквантовая криптография:**

* Разработка алгоритмов, устойчивых к квантовым атакам
* Решетчатая криптография, код-ориентированная криптография

**3. Задачи и цели криптографической защиты информации в информационно-телекоммуникационных сетях**

**Основные задачи криптографии:**

1. **Обеспечение конфиденциальности** - защита данных от просмотра
2. **Обеспечение целостности** - защита от изменений
3. **Аутентификация** - подтверждение источника данных
4. **Невозможность отказа** - подтверждение факта передачи/приема информации
5. **Неотслеживаемость** - применяется в системах электронных денег

**Современная криптография включает четыре раздела:**

1. Симметричные криптосистемы
2. Криптосистемы с открытыми ключами
3. Системы электронной подписи
4. Системы управления ключами

**Базовые требования к современным криптографическим системам:**

1. Знание алгоритма шифрования не должно снижать криптостойкости
2. Зашифрованное сообщение должно поддаваться чтению только при наличии правильного ключа
3. Шифр должен быть стойким даже при знании нарушителем большого количества исходных и зашифрованных данных
4. Число операций для расшифрования должно выходить за пределы возможностей современной техники
5. Незначительное изменение ключа или исходного текста должно приводить к существенному изменению шифртекста (лавинный эффект)
6. Не должно быть простых зависимостей между ключами
7. Все ключи должны быть равнозначными

**4. Арифметика больших чисел. Сферы применения. Примеры алгоритмов**

**Определение и представление**

Арифметика больших чисел - это раздел криптографии, изучающий алгоритмы выполнения операций над числами высокой разрядности, которые не помещаются в стандартные типы данных процессора.

**Представление целого числа a в системе счисления с основанием B:**

a = aₙ₋₁Bⁿ⁻¹ + aₙ₋₂Bⁿ⁻² + ... + a₁B + a₀

где 0 ≤ aᵢ < B

Это представление аналогично представлению полинома степени n-1:

a(x) = aₙ₋₁xⁿ⁻¹ + aₙ₋₂xⁿ⁻² + ... + a₁x + a₀

**Основные алгоритмы**

**1. Сложение**

**Алгоритм 1.** Найти сумму w = a + b:

1. Определить c = 0, i = 0
2. Для i = 0, 1, ..., n-1 выполнить:
   * Вычислить t = aᵢ + bᵢ + c
   * Положить c = t mod B
   * wᵢ = c
3. Положить wₙ = t
4. Результат: w = wₙBⁿ + wₙ₋₁Bⁿ⁻¹ + ... + w₁B + w₀

**2. Вычитание**

**Алгоритм 2.** Найти разность w = a - b (где 0 < b ≤ a):

1. Определить c = 0, i = 0
2. Для i = 0, 1, ..., n-1 выполнить:
   * Вычислить t = aᵢ - bᵢ + c
   * Положить c = t mod B
   * wᵢ = c
3. Результат: w = wₙ₋₁Bⁿ⁻¹ + wₙ₋₂Bⁿ⁻² + ... + w₁B + w₀

**3. Умножение**

**Алгоритм 3.** Найти произведение w = a × b:

1. Для i = 0, 1, ..., n+m-1 определить wᵢ = 0
2. Для i = 0, 1, ..., m-1 выполнить:
   * Определить c = 0
   * Для j = 0, 1, ..., n-1 выполнить:
     + Вычислить t = wᵢ₊ⱼ + aⱼ × bᵢ + c
     + Положить wᵢ₊ⱼ = t mod B
     + c = t mod B
   * wᵢ₊ₙ = c
3. Результат: w = wₙ₊ₘ₋₁Bⁿ⁺ᵐ⁻¹ + ... + w₁B + w₀

**4. Деление**

**Алгоритм 4.** Найти частное q и остаток r от деления a на b:

1. Определить такое целое число d > 0, что d × bₙ₋₁ ≥ [B/2]
2. Для i = m+n, m+n-1, ..., 1 выполнить:
   * Определить Q = min((Rᵢ × B + Rᵢ₋₁) / Vₙ₋₁, B-1)
   * Пока Vₙ₋₂ × Q > (RᵢB + Rᵢ₋₁ - QVₙ₋₁)B + Rᵢ₋₂ выполнять Q = Q-1
   * Вычислить w = r - bQ и, если w < 0, положить Q = Q-1
   * Положить w = r - bQ и qᵢ₋ₙ = Q

**Сферы применения**

* Криптосистемы с открытым ключом (RSA, ECC)
* Цифровые подписи
* Генерация больших простых чисел
* Модульная арифметика
* Вычисления в конечных полях

**5. Методы ускоренных арифметических вычислений**

**Модульное умножение**

Для эффективного вычисления произведения по модулю используются специальные алгоритмы, позволяющие избежать вычисления полного произведения.

**Метод Монтгомери**

Это эффективный метод для выполнения модульного умножения, особенно полезный при многократных операциях с одним и тем же модулем.

**Основная идея:** Вместо вычисления ab mod m, вычисляется Montgomery-форма чисел и выполняются операции в этой форме.

**Преимущества:**

* Избегает операции деления
* Эффективен для многократных операций
* Особенно полезен для модульного возведения в степень

**Модульное возведение в степень**

**Алгоритм 8.** Модульное возведение в степень:

* Вход: целое число a, показатель степени n > 2, n = (nₖ, nₖ₋₁, ..., n₀)
* Выход: c = aⁿ (mod m)

**Алгоритм:**

1. Положить c ← a^nₖ
2. Для i = k-1, k-2, ..., 0 выполнить:
   * Вычислить c ← c² (mod m)
   * При nᵢ = 1 вычислить c ← ca (mod m)
3. Результат: c

**Пример:** Для вычисления 3¹¹ mod 7:

* 11 = (1011)₂
* Начинаем с c = 3
* i = 2: c = 3² = 9 ≡ 2 (mod 7), n₂ = 1, c = 2×3 = 6 (mod 7)
* i = 1: c = 6² = 36 ≡ 1 (mod 7), n₁ = 1, c = 1×3 = 3 (mod 7)
* i = 0: c = 3² = 9 ≡ 2 (mod 7), n₀ = 1, c = 2×3 = 6 (mod 7)

**6. Способы поиска обратного значения в конечном поле. Расширенный алгоритм Евклида**

**Расширенный алгоритм Евклида**

Позволяет не только найти НОД(a, b), но и коэффициенты u, v такие, что au + bv = НОД(a, b).

**Алгоритм:**

1. Если b = 0, то НОД(a, b) = a, u = 1, v = 0
2. Иначе:
   * Вычислить q = a div b, r = a mod b
   * Рекурсивно вызвать алгоритм для (b, r)
   * Пересчитать коэффициенты

**Пример:** Найти НОД(1071, 462) и коэффициенты:

* 1071 = 2 × 462 + 147
* 462 = 3 × 147 + 21
* 147 = 7 × 21 + 0

Обратный ход:

* 21 = 462 - 3 × 147
* 21 = 462 - 3 × (1071 - 2 × 462) = 7 × 462 - 3 × 1071

Результат: НОД(1071, 462) = 21, u = -3, v = 7

**Поиск обратного элемента**

Для поиска a⁻¹ mod m используется расширенный алгоритм Евклида:

* Если НОД(a, m) = 1, то существует u такое, что au ≡ 1 (mod m)
* Значение u и есть обратный элемент a⁻¹

**7. Сравнения и классы вычетов. Привести примеры**

**Определение сравнений**

Два целых числа a и b называются **сравнимыми по модулю m**, если они имеют одинаковые остатки от деления на m.

**Обозначение:** a ≡ b (mod m)

**Эквивалентное определение:** a ≡ b (mod m) ⟺ m | (a - b)

**Свойства сравнений**

1. **Рефлексивность:** a ≡ a (mod m)
2. **Транзитивность:** a ≡ b (mod m), b ≡ c (mod m) ⟹ a ≡ c (mod m)
3. **Почленное сложение:** a₁ ≡ b₁ (mod m), a₂ ≡ b₂ (mod m) ⟹ a₁ + a₂ ≡ b₁ + b₂ (mod m)
4. **Почленное умножение:** a₁ ≡ b₁ (mod m), a₂ ≡ b₂ (mod m) ⟹ a₁a₂ ≡ b₁b₂ (mod m)
5. **Прибавление числа:** a ≡ b (mod m) ⟹ a + c ≡ b + c (mod m)
6. **Умножение на число:** a ≡ b (mod m) ⟹ ac ≡ bc (mod m)
7. **Деление на взаимно простое число:** ad ≡ bd (mod m), НОД(m, d) = 1 ⟹ a ≡ b (mod m)
8. **Сокращение:** НОД(a, m) = НОД(b, m) = 1 ⟹ ac ≡ bc (mod m) ⟹ a ≡ b (mod m)

**Классы вычетов**

**Класс вычетов** числа a по модулю m - это множество всех чисел, сравнимых с a по модулю m.

**Полная система вычетов** по модулю m - это совокупность m чисел, попарно несравнимых по модулю m.

**Приведенная система вычетов** по модулю m - это совокупность чисел из полной системы вычетов, взаимно простых с m.

**Примеры**

1. **Пример сравнения:** 17 ≡ 5 (mod 12), так как 17 - 5 = 12 делится на 12
2. **Полная система вычетов по модулю 5:** {0, 1, 2, 3, 4}
3. **Приведенная система вычетов по модулю 6:** {1, 5} (числа, взаимно простые с 6)

**8. Методы решения систем сравнений первой степени**

**Решение одного сравнения**

Рассмотрим сравнение ax ≡ b (mod m).

**Случай 1: НОД(a, m) = 1**

1. Найти число u такое, что au ≡ 1 (mod m) (обратный элемент)
2. Умножить обе части сравнения на u: x ≡ bu (mod m)
3. Сравнение имеет единственное решение по модулю m

**Случай 2: НОД(a, m) = d > 1**

1. Необходимое условие разрешимости: d | b
2. Если условие выполнено, привести к виду: a₁x ≡ b₁ (mod m₁), где a = a₁d, b = b₁d, m = m₁d
3. Получаем одно решение x ≡ x₁ (mod m₁)
4. По модулю m имеем d решений: x₁, x₁ + m₁, x₁ + 2m₁, ..., x₁ + (d-1)m₁

**Теорема 1**

Пусть НОД(a, m) = d. Сравнение ax ≡ b (mod m) разрешимо тогда и только тогда, когда d | b. В этом случае оно имеет d решений.

**Система из двух сравнений**

Рассмотрим систему:

x ≡ b₁ (mod m₁)

x ≡ b₂ (mod m₂)

**Алгоритм решения:**

1. Из первого сравнения: x = b₁ + m₁t
2. Подставить во второе: m₁t ≡ b₂ - b₁ (mod m₂)
3. Критерий разрешимости: НОД(m₁, m₂) | (b₂ - b₁)
4. Если условие выполнено, система имеет единственное решение по модулю НОК(m₁, m₂)

**Китайская теорема об остатках**

Пусть m₁, m₂, ..., mᵣ - попарно взаимно простые модули, m = m₁m₂...mᵣ.

**Теорема:** Система сравнений

x ≡ u₁ (mod m₁)

x ≡ u₂ (mod m₂)

...

x ≡ uᵣ (mod mᵣ)

имеет единственное решение по модулю m:

x ≡ Σ(i=1 to r) cᵢdᵢuᵢ (mod m)

где cᵢ = m/mᵢ, а dᵢ = cᵢ⁻¹ (mod mᵢ).

**Пример:** Решить систему

x ≡ 2 (mod 5)

x ≡ 15 (mod 17)

x ≡ 5 (mod 12)

1. M₀ = 5×17×12 = 1020
2. M₁ = 1020/5 = 204, M₂ = 1020/17 = 60, M₃ = 1020/12 = 85
3. Решить вспомогательные сравнения:
   * 204y₁ ≡ 2 (mod 5) ⟹ y₁ = 3
   * 60y₂ ≡ 15 (mod 17) ⟹ y₂ = 13
   * 85y₃ ≡ 5 (mod 12) ⟹ y₃ = 5
4. x = 204×3 + 60×13 + 85×5 = 1817 ≡ 797 (mod 1020)

**9. Решение сравнений высших порядков. Критерий Эйлера**

**Критерий Эйлера**

Пусть p - простое число и d = НОД(n, p-1).

**Теорема:** Сравнение xⁿ ≡ a (mod p), где a ≢ 0 (mod p), разрешимо тогда и только тогда, когда a^((p-1)/d) ≡ 1 (mod p).

В случае разрешимости оно имеет ровно d различных решений.

**Квадратичные вычеты**

Для квадратичных сравнений x² ≡ a (mod p):

* Если a ≢ 0 (mod p) - квадратичный вычет по простому модулю p > 2, то сравнение имеет два решения: x ≡ ±x₀ (mod p)
* Если p | a, то сравнение имеет единственное решение x ≡ 0 (mod p)

**Символ Лежандра**

Для простого p > 2 и целого a:

(a/p) = { 1, если a является квадратичным вычетом по модулю p и a ≢ 0 (mod p) -1, если a является квадратичным невычетом по модулю p 0, если a ≡ 0 (mod p) }

**Другая формулировка критерия Эйлера**

a^((p-1)/2) ≡ (a/p) (mod p)

Количество решений сравнения x² ≡ a (mod p): Nₚ = 1 + (a/p)

**Символ Якоби**

Для составного нечетного числа n = p₁^α₁ × p₂^α₂ × ... × pₖ^αₖ:

(a/n) = (a/p₁)^α₁ × (a/p₂)^α₂ × ... × (a/pₖ)^αₖ

где каждый множитель - символ Лежандра.

**Свойства символа Якоби:**

1. (a/n) = (b/n), если a ≡ b (mod n)
2. (ab/n) = (a/n)(b/n)
3. (a/mn) = (a/m)(a/n)
4. (-1/n) = (-1)^((n-1)/2)
5. (2/n) = (-1)^((n²-1)/8)

**10. Существование первообразных корней**

**Определение первообразного корня:**

Число g называется первообразным корнем по модулю m, если порядок g по модулю m равен φ(m), где φ - функция Эйлера.

**Теорема о существовании:**

**Первообразные корни существуют только для модулей вида:**

* m = 1, 2, 4
* m = p^k (p - нечетное простое)
* m = 2p^k (p - нечетное простое)

**Количество первообразных корней:**

Если первообразные корни по модулю m существуют, то их количество равно φ(φ(m)).

**Свойства первообразных корней:**

1. **Если g - первообразный корень по модулю p, то g^k - первообразный корень тогда и только тогда, когда gcd(k, p-1) = 1**
2. **Все первообразные корни по модулю p имеют вид g^k, где gcd(k, p-1) = 1**
3. **Если g - первообразный корень по модулю p², то g или g+p является первообразным корнем по модулю p^k для всех k ≥ 1**

**Алгоритм поиска первообразных корней:**

1. **Факторизовать φ(m) = p₁^a₁ × p₂^a₂ × ... × pₖ^aₖ**
2. **Для каждого кандидата g проверить:**
   * g^(φ(m)/pᵢ) ≢ 1 (mod m) для всех простых делителей pᵢ
3. **Если все проверки пройдены, g - первообразный корень**

**Примеры:**

**Пример 1:** Первообразные корни по модулю 7

* φ(7) = 6 = 2 × 3
* Проверяем g = 3:
  + 3^(6/2) = 3³ ≡ 6 ≢ 1 (mod 7) ✓
  + 3^(6/3) = 3² ≡ 2 ≢ 1 (mod 7) ✓
* Значит, 3 - первообразный корень по модулю 7

**Пример 2:** Модуль 15

* φ(15) = φ(3)φ(5) = 2 × 4 = 8
* Но 15 = 3 × 5, и согласно теореме, первообразных корней по модулю 15 не существует

**Применение в криптографии:**

1. **Генерация ключей в системах на основе дискретного логарифма**
2. **Построение циклических групп**
3. **Псевдослучайные генераторы**
4. **Протоколы обмена ключами (Диффи-Хеллман)**

## 11. Понятие группы и подгруппы, циклические группы

**Группа** - алгебраическая структура с одной бинарной операцией, удовлетворяющая следующим аксиомам:

* Замкнутость относительно операции
* Ассоциативность
* Существование нейтрального элемента
* Существование обратного элемента для каждого элемента

**Подгруппа** - подмножество группы, которое само является группой относительно той же операции.

**Циклическая группа** - группа, которая может быть порождена одним элементом (генератором или порождающим элементом). Все элементы циклической группы можно получить как степени генератора.

**Свойства циклических групп:**

* Любая подгруппа циклической группы также циклическая
* Циклическая группа порядка n изоморфна группе вычетов по модулю n
* Мультипликативная группа ненулевых элементов конечного поля является циклической

**12. Гомоморфизмы групп**

**Определение гомоморфизма групп**

Пусть (G, ∗) и (H, ⊕) — две группы. Отображение f: G → H называется **гомоморфизмом групп**, если для всех a, b ∈ G выполняется: f(a ∗ b) = f(a) ⊕ f(b)

**Типы гомоморфизмов**

1. **Эпиморфизм** — сюръективный гомоморфизм (f(G) = H)
2. **Мономорфизм** — инъективный гомоморфизм
3. **Изоморфизм** — биективный гомоморфизм
4. **Эндоморфизм** — гомоморфизм группы в себя
5. **Автоморфизм** — изоморфизм группы в себя

**Ключевые понятия**

**Ядро гомоморфизма (Ker f):** Ker f = {g ∈ G | f(g) = e\_H} где e\_H — единичный элемент группы H

**Образ гомоморфизма (Im f):** Im f = {f(g) | g ∈ G}

**Свойства гомоморфизмов**

1. f(e\_G) = e\_H (образ единицы — единица)
2. f(g^(-1)) = f(g)^(-1) (образ обратного — обратный к образу)
3. Ker f — нормальная подгруппа в G
4. Im f — подгруппа в H

**Нормальные подгруппы**

Подгруппа H группы G называется **нормальной подгруппой** (обозначается H ⊲ G), если для любого элемента g ∈ G выполняется: gH = Hg

**Факторгруппа**

Если H ⊲ G, то множество смежных классов G/H = {gH | g ∈ G} образует группу, называемую **факторгруппой**.

**Теорема о гомоморфизме (основная теорема)**

Если f: G → H — гомоморфизм групп, то: G/Ker f ≅ Im f

**13. Группы подстановок и действие группы на множестве**

**Группы подстановок:**

**Подстановка** - биективное отображение конечного множества на себя.

**Симметрическая группа Sₙ** - группа всех подстановок множества {1, 2, ..., n}.

**Циклы:**

**Цикл длины k** - подстановка, переставляющая элементы по циклу: (a₁ a₂ ... aₖ): a₁ → a₂ → ... → aₖ → a₁

**Разложение на циклы:**

Любая подстановка может быть разложена на произведение непересекающихся циклов.

**Пример:** (1 2 3)(4 5) в S₅

**Знак подстановки:**

**Четная подстановка** - произведение четного числа транспозиций **Нечетная подстановка** - произведение нечетного числа транспозиций

**Альтернированная группа Aₙ** - группа четных подстановок.

**Действие группы на множестве:**

**Действие** группы G на множестве X - отображение G × X → X, удовлетворяющее:

1. e · x = x для всех x ∈ X
2. (gh) · x = g · (h · x) для всех g, h ∈ G, x

!!!!!!!!!!!!!!!!!

**14. Структура конечных колец**

Конечное кольцо — это множество RRR с двумя бинарными операциями (сложением +++ и умножением ⋅\cdot⋅), удовлетворяющее следующим аксиомам:

1. **(R, +)** — абелева группа:
   * Сложение коммутативно: a+b=b+aa + b = b + aa+b=b+a
   * Существует нейтральный элемент (нуль): 000
   * Для каждого a∈Ra \in Ra∈R существует противоположный элемент −a-a−a, такой что a+(−a)=0a + (-a) = 0a+(−a)=0
2. **(R, ·)** — полугруппа:
   * Умножение ассоциативно: (ab)c=a(bc)(ab)c = a(bc)(ab)c=a(bc)
   * В большинстве рассматриваемых случаев — есть **единица** 1∈R1 \in R1∈R, такая что 1⋅r=r⋅1=r1·r = r·1 = r1⋅r=r⋅1=r
3. **Дистрибутивность**:
   * a(b+c)=ab+aca(b + c) = ab + aca(b+c)=ab+ac
   * (a+b)c=ac+bc(a + b)c = ac + bc(a+b)c=ac+bc

**📌 Примеры конечных колец:**

* Кольцо вычетов по модулю nnn: Zn\mathbb{Z}\_nZn​ с операциями сложения и умножения по модулю nnn
* Множество матриц Mn(Fp)M\_n(\mathbb{F}\_p)Mn​(Fp​)
* Кольца многочленов Fp[x]\mathbb{F}\_p[x]Fp​[x]

**📍 Особенности:**

* В кольце Zn\mathbb{Z}\_nZn​, если nnn **составное**, то присутствуют **делители нуля** (например: ab=0ab = 0ab=0, но a≠0,b≠0a \ne 0, b \ne 0a=0,b=0)
* Если nnn — **простое**, то Zn\mathbb{Z}\_nZn​ — это поле

**🔹 15. Подкольца и свойства колец**

**🔸 Подкольцо — определение:**

Подмножество S⊆RS \subseteq RS⊆R называется **подкольцом** кольца RRR, если:

1. (S,+)(S, +)(S,+) — подгруппа группы (R,+)(R, +)(R,+)
2. Для любых a,b∈Sa, b \in Sa,b∈S, также ab∈Sab \in Sab∈S
3. Если RRR содержит единицу 111, то 1∈S1 \in S1∈S

**📌 Примеры:**

* Z⊂Q⊂R\mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}Z⊂Q⊂R
* Кольцо чётных чисел 2Z2\mathbb{Z}2Z — ассоциативное коммутативное кольцо **без единицы**
* Подкольцо в кольце вычетов Z10\mathbb{Z}\_{10}Z10​: числа, кратные 5, образуют кольцо с единицей, **не совпадающей с единицей всего кольца**

**🔸 Свойства колец:**

Из аксиом кольца вытекают следующие свойства:

* a⋅0=0⋅a=0a·0 = 0·a = 0a⋅0=0⋅a=0
* (−a)b=−(ab)=a(−b)(-a)b = -(ab) = a(-b)(−a)b=−(ab)=a(−b)
* (a−b)c=ac−bc(a - b)c = ac - bc(a−b)c=ac−bc
* a(b−c)=ab−aca(b - c) = ab - aca(b−c)=ab−ac
* (−a)(−b)=ab(-a)(-b) = ab(−a)(−b)=ab

**🔸 Делители нуля:**

Элементы a,b∈Ra, b \in Ra,b∈R, a≠0,b≠0a \ne 0, b \ne 0a=0,b=0, но ab=0ab = 0ab=0 — **делители нуля**. В коммутативных кольцах — нет различия между левыми и правыми.

📌 Пример: в кольце вычетов Zn\mathbb{Z}\_nZn​ при составном nnn, всегда существуют делители нуля.

**🔸 Нильпотентный элемент:**

Элемент x∈Rx \in Rx∈R называется **нильпотентным**, если xn=0x^n = 0xn=0 для некоторого n>0n > 0n>0

**🔸 Идемпотент:**

Элемент x∈Rx \in Rx∈R, для которого x2=xx^2 = xx2=x.  
Если x≠0x \ne 0x=0 и x≠1x \ne 1x=1, то он обязательно является делителем нуля.

**🔸 Множество обратимых элементов:**

Обозначается U(R)U(R)U(R). Это **мультипликативная группа** всех элементов кольца, у которых существует обратный (в смысле умножения).

## 16. Идеалы и гомоморфизмы колец

**Определение кольца**

Множество R с двумя бинарными операциями (сложением + и умножением ·) называется **ассоциативным кольцом с единицей**, если:

1. Относительно сложения (R,+) — абелева группа
2. Умножение — ассоциативная операция с единичным элементом 1
3. Выполняются законы дистрибутивности: (a+b)c = ac+bc, c(a+b) = ca+cb

**Идеалы**

**Определение левого идеала:** Подмножество I ⊂ R называется левым идеалом кольца R, если:

1. I — подгруппа аддитивной группы (R,+)
2. rI ⊆ I для любого r ∈ R

**Определение правого идеала:** Аналогично, но вместо условия 2) имеем: Ir ⊆ I

**Двусторонний идеал:** Подмножество, являющееся одновременно левым и правым идеалом.

**Примеры идеалов**

1. {0} и R — тривиальные идеалы
2. nZ в кольце целых чисел Z
3. Главные идеалы Ra = {ra | r ∈ R} в коммутативном кольце

**Гомоморфизмы колец**

Отображение f: R → R' называется **гомоморфизмом колец**, если:

1. f(a + b) = f(a) + f(b)
2. f(ab) = f(a)f(b)

**Свойства гомоморфизмов колец**

1. f(0) = 0', f(-a) = -f(a)
2. Если R содержит 1, R' содержит 1' и Im f = R', то f(1) = 1'
3. Ker f — двусторонний идеал кольца R
4. f — изоморфизм ⟺ Ker f = {0} и Im f = R'

**Специальные элементы колец**

**Делители нуля:** Элементы a, b ∈ R (a ≠ 0, b ≠ 0) называются делителями нуля, если ab = 0.

**Нильпотентный элемент:** Элемент x ∈ R называется нильпотентным, если x^n = 0 для некоторого n ∈ ℕ.

**Идемпотент:** Элемент x кольца R называется идемпотентом, если x² = x.

**17. Понятие и свойства конечного простого поля**

**Определение поля**

**Полем** <F, +, ⋅> называется множество F с двумя бинарными операциями + и ⋅, такими, что:

* F является аддитивной абелевой группой с единичным элементом 0
* элементы F, отличные от 0, образуют мультипликативную абелеву группу, единичным элементом которой является 1
* операции сложения и умножения связаны законом дистрибутивности
* для операций сложения и умножения существуют обратные операции: вычитание и деление (кроме деления на ноль)

**Примеры полей**

* множество рациональных чисел **Q**
* множество действительных чисел **R**
* множество комплексных чисел **C**

**Конечные поля**

Если поле представляет собой конечное множество, состоящее из *q* элементов, то такое поле называется **конечным полем** или **полем Галуа** и обозначается GF(*q*) или F*q*.

**Важное свойство:** *q* = *p^m*, где *p* — простое число, *m* — натуральное число

**Простые и расширенные поля**

* При *m* = 1 поле называется **простым**
* В противном случае поле называется **расширенным**

**Критическое замечание:** Если *p* — составное число, то алгебраическая система <F, +, ⋅>, где сложение и умножение модулярные операции, полем не является; эта система образует кольцо, где деление даже на ненулевой элемент возможно не всегда.

**Примитивный элемент поля**

Мультипликативная группа ненулевых элементов произвольного конечного поля GF(*q*) является циклической. Образующий элемент α циклической группы называется **примитивным элементом** поля GF(*q*).

Все элементы конечного поля можно представить следующим образом: GF(*q*) = {0, α, α², ..., α^(q-2), α^(q-1) = α⁰ = 1}

где α — примитивный элемент поля.

**18. Аддитивная и мультипликативная группы конечных полей**

**Структура групп**

* **Аддитивная группа** поля включает в себя все элементы поля
* **Мультипликативная группа** включает все элементы за исключением нуля

**Свойства групп**

* Обе группы являются **коммутативными**
* В аддитивной группе существуют порождающие элементы, поэтому она является **циклической**

**Применение в криптографии**

Коммутативные группы простых полей широко используются в различных криптосистемах, особенно в криптосистемах с открытым ключом:

* схема открытого распределения ключей Диффи-Хеллмана
* ЭЦП Эль-Гамаля
* ЭЦП Шнорра
* стандарты ЭЦП DSA, ECDSA
* ГОСТ Р 34.10-94, ГОСТ Р 34.10-2001

**19. Примеры использования колец и конечных полей в двухключевой криптографии**

**Схема шифрования по Шеннону**

* **Z\_от** — множество открытых текстов
* **Z\_шт** — множество шифртекстов с распределениями P\_от и P\_шт
* **K** — множество ключей с распределением P\_К
* **T: K × Z\_от → Z\_шт** — отображение шифрования
* **T⁻¹: K × Z\_шт → Z\_от** — отображение расшифрования

**Теорема Шеннона о совершенном шифре**

Шифр совершенный, если для всех m∈Z\_от и x∈Z\_шт выполняется: P{μ = m|ξ = x} = P{μ = m}

**Теорема:**

* а) Если шифр совершенный, то |K|≥|Z\_от|
* б) При |K| = |Z\_от| существует совершенный шифр

**Шифр Вернама (One-time pad)**

Система симметричного шифрования, изобретённая в 1917 году сотрудниками AT&T Мейджором Джозефом Моборном и Гильбертом Вернамом.

**Принцип:** Для получения шифротекста открытый текст объединяется операцией «исключающее ИЛИ» с ключом.

**Требования к ключу:**

1. быть истинно случайным
2. совпадать по размеру с заданным открытым текстом
3. применяться только один раз

**Схема открытого распределения ключей Диффи-Хеллмана**

Каждый абонент выбирает случайный **секретный ключ x** и вырабатывает **открытый ключ y** в соответствии с формулой: y = α^x (mod p)

**Процедура:**

1. Все абоненты размещают свои открытые ключи в общедоступном справочнике
2. Абонент **A** берет из справочника открытый ключ абонента **B** и вычисляет общий секретный ключ: Z\_AB = (y\_B)^x\_A = (α^x\_B)^x\_A
3. Абонент **B** аналогично вычисляет: Z\_AB = (y\_A)^x\_B = (α^x\_A)^x\_B = α^(x\_B·x\_A) (mod p)

**Схема электронной цифровой подписи (ЭЦП) Шнорра**

Параметр α - число, относящееся по модулю p к простому показателю γ (160-256 бит).

**Проверочное уравнение:** R = α^S · y^E mod p

**Вычисление подписи к сообщению M:**

1. Генерируется случайное число k, 1 < k < γ
2. Вычисляется значение R = α^k mod p
3. К сообщению M присоединяется число R и вычисляется хэш-функция H от значения M||R: E = H(M||R)
4. Вычисляется значение S: S = k − xE mod q

**Подписью является пара чисел (E, S)**

**Процедура проверки подлинности подписи:**

1. Вычисляется значение R′: R′ = α^S · y^E mod p, где y — открытый ключ
2. К сообщению M присоединяется число R′ и вычисляется значение хэш-функции E′ = H(M||R′)
3. Выполняется сравнение значений E и E′. Если E = E′, то подпись считается подлинной

**Модель КГС RSA**

**Параметры:**

* p и q — два больших простых числа
* N = p·q
* e ∈ Z такое, что НОД(e, χ\_π(N)) = 1, где χ\_π(n) = (p-1)·(q-1) — функция Эйлера
* k = <e, N> — открытый ключ
* d = e⁻¹ mod (χ\_π(N)) — закрытый ключ

**Шифрование:** E\_k(m) = m^e mod N **Расшифрование:** D\_k(c) = c^d mod N

Поскольку ed ≡ 1 mod(χ\_π(N)), то по теореме Эйлера m^ed ≡ m mod N ⇒ m^ed mod N = m

**Российский стандарт ЭЦП ГОСТ Р 34.10-94**

**Параметры:**

* p — простое число, причем: 510 ≤ |p| ≤ 512 либо 1022 ≤ |p| ≤ 1024
* q — простой делитель числа p - 1, 2^255 ≤ q ≤ 2^256 либо 2^511 ≤ q ≤ 2^512
* α — число, относящееся к показателю q по модулю p

**Процедура вычисления подписи:**

1. Генерируется случайное число k, 1 < k < q
2. Вычисляется значение R = (α^k mod p) mod q
3. Вычисляется хэш-функция H от подписываемого сообщения
4. Вычисляется вторая часть подписи: S = kH + xR mod q (Если S = 0, процедура генерации подписи повторяется)

**Проверка подлинности подписи:**

1. Проверяется выполнение условий R < q и S < q
2. Вычисляется значение хэш-функции H от сообщения
3. Вычисляется значение R′ = (α^(S/H) · y^(-R/H) mod p) mod q
4. Сравниваются значения R и R′. Если R = R′, то подпись признается действительной

**20. Основные свойства эллиптических кривых**

**Определение эллиптической кривой:** Эллиптическая кривая E(GF(q)) определяется уравнением, вид которого зависит от характеристики поля:

**Виды уравнений ЭК в зависимости от характеристики поля:**

1. **Для поля характеристики 2:**
   * y² + ay = x³ + bx + c (инвариант j = 0)
   * y² + xy = x³ + ax² + b (инвариант j = 1/b)
2. **Для поля характеристики 3:**
   * y² = x³ + ax + b (инвариант j = 0)
   * y² = x³ + ax² + b (инвариант j ≠ 0)
3. **Для поля характеристики p ≠ 2, p ≠ 3:**
   * y² = x³ + ax + b (инвариант j = 1728(4a)³/Δ)

**Основные свойства:**

1. **Дискриминант:** Δ = -16(4a³ + 27b²)
   * Если Δ ≠ 0, кривая называется **неособой**
   * Неособость означает отсутствие особых точек (точек самопересечения)
2. **Инвариант j:** j = 1728(4a³)/Δ
   * Две кривые с одинаковым инвариантом j являются **изоморфными**
   * Коэффициенты a, b по известному инварианту определяются как:
     + a ≡ 3k (mod p)
     + b ≡ 2k (mod p)
     + где k = j/(1728 - j) (mod p), при j ≠ 0, j ≠ 1728
3. **Условия изоморфизма:** Две ЭК изоморфны, если существует элемент u ∈ GF(q), u ≠ 0, такой что:
   * y₁ = u³y₂
   * x₁ = u²x₂
   * a₁ = u⁴a₂
   * b₁ = u⁶b₂
4. **Условие неособости:** Для полей характеристики p ≠ 2, p ≠ 3 кривая неособа, если многочлен x³ + ax + b не имеет кратных корней.

**21. Групповой закон сложения точек эллиптических кривых**

**Формальное представление группового закона:**

**1. Операция обращения точки:** -(x, y) = (x, -y)

**2. Операция сложения различных точек P₃ = P₁ + P₂:**

* x₃ ≡ λ² - x₁ - x₂ (mod p)
* y₃ ≡ λ(x₁ - x₃) - y₁ (mod p)
* где λ ≡ (y₂ - y₁)/(x₂ - x₁) (mod p)

**3. Операция удвоения точки 2P₁ = (x₃, y₃):**

* x₃ ≡ λ² - 2x₁ (mod p)
* y₃ ≡ λ(x₁ - x₃) - y₁ (mod p)
* где λ ≡ (3x₁² + a)/(2y₁) (mod p)

**Проективные координаты:** Для повышения эффективности вычислений используются проективные координаты, где кривая записывается как: E(GF(q)): Y²Z = X³ + aXZ² + bZ³

**Удвоение в проективных координатах:**

* X₃ = 2Y₁Z₁(3X₁² + aZ₁²)² - 8X₁Y₁²Z₁ (mod p)
* Y₃ = 4Y₁²Z₁(3X₁²(3X₁² + aZ₁²) - 2) - 8Y₁³Z₁³ (mod p)
* Z₃ = 8Y₁³Z₁³ (mod p)

**22. Использование эллиптических кривых для построения криптографических алгоритмов**

**Протокол распределения ключей (аналог Диффи-Хеллмана):**

1. **Инициализация:** Выбираются общие параметры:
   * Эллиптическая кривая E над полем GF(q)
   * Базовая точка G порядка n
2. **Генерация ключей:**
   * Абонент A: генерирует секретный ключ nₐ, вычисляет Pₐ = nₐG
   * Абонент B: генерирует секретный ключ nᵦ, вычисляет Pᵦ = nᵦG
3. **Обмен открытыми ключами:**
   * A передает B свой открытый ключ Pₐ
   * B передает A свой открытый ключ Pᵦ
4. **Вычисление сеансового ключа:**
   * A вычисляет: Kₐ = nₐPᵦ = nₐnᵦG
   * B вычисляет: Kᵦ = nᵦPₐ = nᵦnₐG
   * Получается: Kₐ = Kᵦ = Kc (общий сеансовый ключ)

**Протокол электронной цифровой подписи (ЭЦП):**

**Параметры системы:**

* Простое число p > 2²⁵⁵ (модуль ЭК)
* Эллиптическая кривая E с коэффициентами a, b ∈ GFₚ
* Порядок группы точек m = #E(GFₚ)
* Простое число q (порядок циклической подгруппы): 2²⁵⁴ < q < 2²⁵⁶
* Базовая точка P ∈ E(GFₚ) порядка q
* Хэш-функция h(·): V∞ → V₂₅₆

**Алгоритм формирования ЭЦП:**

1. Вычислить h = h(M)
2. Определить e ≡ α (mod q), где α - целое число по значению h
3. Сгенерировать случайное k: 0 < k < q
4. Вычислить C = kP и r ≡ xc (mod q)
5. Вычислить s ≡ (rd + ke) (mod q)
6. Подпись: ζ = (r || s)

**Алгоритм проверки ЭЦП:**

1. Проверить: 0 < r < q, 0 < s < q
2. Вычислить h = h(M)
3. Определить e ≡ α (mod q)
4. Вычислить v = e⁻¹ (mod q)
5. Вычислить z₁ = sv (mod q), z₂ = -rv (mod q)
6. Вычислить C = z₁P + z₂Q и R = xc (mod q)
7. Проверить: R = r

**23. Понятие эллиптической кривой. Основные свойства эллиптических кривых**

**Определение эллиптической кривой**

Эллиптическая кривая E над конечным полем GF(q) определяется уравнением вида: **y² = x³ + ax + b** где a, b ∈ GF(q) и дискриминант Δ = -16(4a³ + 27b²) ≠ 0

**Виды уравнений ЭК в зависимости от характеристики поля**

| **Характеристика p** | **Уравнение кривой E(GF(q))** | **Инвариант j** |
| --- | --- | --- |
| p = 2 | y² + ay = x³ + bx + c | j = 0 |
| p = 2 | y² + xy = x³ + ax² + b | j = 1/b |
| p = 3 | y² = x³ + ax + b | j = 0 |
| p = 3 | y² = x³ + ax² + b | j ≠ 0 |
| p ≠ 2, p ≠ 3 | y² = x³ + ax + b | j = 1728(4a)³/Δ |

**Основные параметры ЭК**

**Дискриминант:** Δ = -16(4a³ + 27b²)

**Инвариант:** j = 1728 · 4a³/Δ

**Коэффициенты по известному инварианту j:**

* a ≡ 3k (mod p)
* b ≡ 2k (mod p) где k = j/(1728 - j) (mod p), j ≠ 0, j ≠ 1728

**Дополнительные свойства ЭК**

1. **Неособая кривая:** Если дискриминант Δ ≠ 0, кривая называется неособой
2. **Изоморфные кривые:** Две кривые с одинаковым инвариантом j являются изоморфными
3. **Условие изоморфизма:** Две ЭК (y² = x³ + ax + b)₁ и (y² = x³ + ax + b)₂ изоморфны, если:
   * y₁ = u³y₂
   * x₁ = u²x₂
   * a₁ = u⁴a₂
   * b₁ = u⁶b₂ где u ∈ GF(q), u ≠ 0
4. **Критерий неособости:** Для характеристики p ≠ 2 и p ≠ 3, если многочлен x³ + ax + b не имеет кратных корней, то кривая неособая

**Порядок группы точек ЭК**

**Теорема Хассе:** Порядок группы точек ЭК ограничен: q + 1 - 2√q ≤ #E(GF(q)) ≤ q + 1 + 2√q

**Специальные случаи:**

* Для p ≡ 2 (mod 3), p ≡ 5 (mod 6) и a = 0: #E(GF(p)) = p + 1
* Для кривой y² ≡ x³ + b (mod p), где p ≡ 1 (mod 6): #E(GF(p)) = p + 1 ± s

**24. Групповой закон сложения точек эллиптических кривых и его геометрическая интерпретация**

**Групповая структура**

Множество точек эллиптической кривой вместе с точкой в бесконечности O образует абелеву группу относительно операции сложения.

**Операция обращения точки**

Для точки P = (x, y) на кривой: **-P = (x, -y)**

**Операция сложения точек P₃ = P₁ + P₂**

**Для различных точек P₁ = (x₁, y₁) и P₂ = (x₂, y₂):**

x₃ ≡ λ² - x₁ - x₂ (mod p) y₃ ≡ λ(x₁ - x₃) - y₁ (mod p)

где λ ≡ (y₂ - y₁)/(x₂ - x₁) (mod p)

**Операция удвоения точки 2P₁ = (x₃, y₃)**

x₃ ≡ λ² - 2x₁ (mod p) y₃ ≡ λ(x₁ - x₃) - y₁ (mod p)

где λ ≡ (3x₁² + a)/(2y₁) (mod p)

**Проективные координаты**

Для эффективных вычислений используются проективные координаты: **E(GF(q)): Y²Z = X³ + aXZ² + bZ³**

**Удвоение точки в проективных координатах:**

* X₃ = 2Y₁Z₁(3X₁² + aZ₁²)² - 8X₁Y₁²Z₁ (mod p)
* Y₃ = 4Y₁²Z₁(3X₁²(3X₁² + aZ₁²) - 2Y₁²Z₁²) (mod p)
* Z₃ = 8Y₁³Z₁³ (mod p)

**Геометрическая интерпретация**

1. **Сложение различных точек:**
   * Проводим прямую через точки P₁ и P₂
   * Находим третью точку пересечения с кривой
   * Отражаем её относительно оси x — получаем P₃ = P₁ + P₂
2. **Удвоение точки:**
   * Проводим касательную к кривой в точке P₁
   * Находим вторую точку пересечения касательной с кривой
   * Отражаем её относительно оси x — получаем 2P₁
3. **Точка в бесконечности:**
   * Служит нейтральным элементом группы
   * P + O = P для любой точки P

**Свойства группового закона**

1. **Коммутативность:** P₁ + P₂ = P₂ + P₁
2. **Ассоциативность:** (P₁ + P₂) + P₃ = P₁ + (P₂ + P₃)
3. **Нейтральный элемент:** P + O = P
4. **Обратный элемент:** P + (-P) = O

**25. Алгоритмы выбора эллиптических кривых с хорошими криптографическими свойствами**

**Параметры, влияющие на стойкость:**

* Вид конечного поля
* Характеристика поля и степень расширения
* Уравнение ЭК
* Порядок циклической подгруппы точек ЭК
* Генератор подгруппы точек ЭК

**Порядок группы точек ЭК:** По теореме Хассе: q + 1 - 2√q ≤ #E(GF(q)) ≤ q + 1 + 2√q

**Специальные случаи:**

1. При p ≡ 2 (mod 3), p ≡ 5 (mod 6) и a = 0: #E(GF(p)) = p + 1
2. Для кривой y² ≡ x³ + b (mod p) при p ≡ 1 (mod 6): #E(GF(p)) = p + 1 ± s

**Методика выбора ЭК:**

1. Определить характеристику p и степень расширения k
2. Выбрать уравнение ЭК согласно характеристике
3. Сгенерировать коэффициенты a, b ∈ GF(p)
4. Определить порядок группы #E(GF(q))
5. Проверить условие нестойкости: #E(GF(q)) не должно делиться на (qⁱ - 1) для i = 1, 2, ..., 30
6. При невыполнении условия - вернуться к шагу 3

**Сложность методов анализа:**

* Метод "встречи посередине": O(√q)
* Метод Полларда: O(√q)
* Метод Сильвера-Полига-Хеллмана: O(log ζ)

## 26. Свойства простых чисел, используемые в криптографии

**Определение простого числа:** Простое число - это натуральное число, не равное 1, которое делится без остатка только на 1 и само себя.

**Основные свойства простых чисел в криптографии:**

1. **Свойство делимости**: Если p - простое число и p делит произведение ab, то p делит a или b.
2. **Поле вычетов**: Кольцо вычетов Zₚ является полем тогда и только тогда, когда p - простое число.
3. **Характеристика поля**: Характеристика каждого поля - это ноль или простое число.
4. **Малая теорема Ферма**: Если p - простое число, а a - натуральное число, то aᵖ⁻¹ ≡ 1 (mod p).
5. **Свойство конечных групп**: Если G - конечная группа с pⁿ элементов, то G содержит элемент порядка p.
6. **Теорема Вильсона**: Натуральное число p > 1 является простым тогда и только тогда, когда (p-1)! + 1 делится на p.
7. **Постулат Бертрана**: Если n > 1 - натуральное число, то существует простое p такое, что n < p < 2n.
8. **Расходимость ряда**: Ряд чисел, обратных к простым числам, расходится.

**Специальные классы простых чисел:**

**Простые числа Мерсенна:** Числа вида Mₚ = 2ᵖ - 1, где p - простое число. Примеры: M₂ = 3, M₃ = 7, M₅ = 31, M₇ = 127.

**Простые числа Ферма:** Числа вида F(n) = 2^(2^n) + 1. Первые четыре числа Ферма (5, 17, 257, 65537) - простые.

**Простые числа Софи Жермен:** Простые числа p такие, что 2p + 1 также простое. Примеры: 2, 3, 5, 11, 23, 29.

**Простые-близнецы:** Пары простых чисел, разность которых равна 2. Примеры: (3,5), (11,13), (17,19).

**27. Алгоритмы построения простых чисел высокой размерности**

**Основные подходы к генерации больших простых чисел:**

1. **Случайная генерация**: Формируются случайные числа заданного размера и проверяются на простоту с помощью вероятностных тестов.
2. **Генерация псевдопростых чисел**: Генерируются псевдопростые числа, которые затем проверяются детерминированными тестами.
3. **Комбинированная генерация**: Сначала формируются псевдопростые числа промежуточного размера, на основе которых затем генерируются псевдопростые числа большего размера.

**Детерминистическая генерация с подбором разложения функции Эйлера:**

1. Формируется набор k простых чисел {q₁, q₂, ..., qₖ} сравнительно малой длины
2. Случайным образом выбираются h простых чисел m₁, m₂, ..., mₕ
3. Вычисляется число p₁ = 1 + 2∏mᵢ
4. Проверяются условия простоты для полученного числа

**Алгоритм по ГОСТ Р 34.10-94:**

**Теорема**: Пусть p = qN + 1, где q - нечетное простое число, N - четное, и p < (2q + 1)². Число p является простым, если выполняются условия:

1. 2^(qN) ≡ 1 (mod p)
2. 2^N ≢ 1 (mod p)

**Шаги алгоритма:**

1. Строится убывающий набор натуральных чисел t₀, t₁, ..., tₛ
2. Последовательно вырабатываются простые числа pₛ, pₛ₋₁, ..., p₀
3. Исходное простое значение pₛ формируется случайным выбором числа размером менее 17 бит
4. Генерация простого числа pᵢ₋₁ по простому числу pᵢ осуществляется с использованием формулы pᵢ₋₁ = pᵢN + 1

**28. Проверка чисел на простоту**

**Классификация алгоритмов проверки:**

* **Детерминированные алгоритмы**: дают точный ответ, но могут быть медленными
* **Вероятностные алгоритмы**: дают быстрый ответ с определенной вероятностью ошибки
* **Полиномиальные алгоритмы**: время работы полиномиально зависит от размера числа

**Основные методы проверки:**

**Решето Эратосфена:** Классический метод для нахождения всех простых чисел до заданного предела путем последовательного исключения составных чисел.

**Решето Сундарама:** Простые числа (кроме 2) получаются по формуле X = 2L + 1, где L - число, не присутствующее в определенных арифметических прогрессиях.

**Тест Ферма:** Проверка соотношения b^(p-1) ≡ 1 (mod p) для различных значений b. Числа, проходящие тест Ферма, но не являющиеся простыми, называются числами Кармайкла.

**Примеры чисел Кармайкла:**

* 561 = 3·11·17
* 1105 = 5·13·17
* 1729 = 7·13·19
* 2465 = 5·17·29

**Тест Соловея-Штрассена:** Проверка равенства (b/p) ≡ b^((p-1)/2) (mod p), где (b/p) - символ Лежандра.

**AKS-тест:** Полиномиальный детерминированный алгоритм Агравала, Кайяла и Саксены - первый известный полиномиальный детерминированный алгоритм проверки простоты.

**29. Вероятностные алгоритмы проверки чисел на простоту**

**Вероятностные алгоритмы** — это алгоритмы, которые с высокой вероятностью определяют, простое ли число, но могут ошибаться (чаще — принимают составное число за простое).

🔸 **Основные алгоритмы**:

1. **Ферма (простая проверка)**  
   Если ppp — простое, то для любого a∈[1,p−1]a \in [1, p - 1]a∈[1,p−1]:

ap−1≡1mod  pa^{p-1} \equiv 1 \mod pap−1≡1modp

Проверка: если an−1≢1mod  na^{n-1} \not\equiv 1 \mod nan−1≡1modn, то nnn — **составное**.  
▪️ Недостаток — **псевдопростые числа**.

1. **Соловая–Штрассена (Solovay–Strassen)**  
   Проверка на основе символа Якоби и сравнения:

a(n−1)/2≢(an)mod  n⇒n составноеa^{(n-1)/2} \not\equiv \left( \frac{a}{n} \right) \mod n \Rightarrow n \text{ составное}a(n−1)/2≡(na​)modn⇒n составное

▪️ Ошибка ≤ 50% на каждом прогоне (при неправильном выборе aaa).  
▪️ Быстрее, но всё ещё вероятностный.

1. **Миллера–Рабина**  
   Один из самых эффективных вероятностных алгоритмов:
   * Представляем n−1=2s⋅dn - 1 = 2^s \cdot dn−1=2s⋅d
   * Проверяем: ad≢1mod  na^d \not\equiv 1 \mod nad≡1modn и a2rd≢−1a^{2^r d} \not\equiv -1a2rd≡−1 для всех 0≤r<s0 \le r < s0≤r<s

Если оба условия выполняются — **n составное**.  
▪️ Каждый прогон снижает вероятность ошибки в 4 раза (ошибка ≤ 1/4).  
▪️ Часто используется в криптографии и при генерации ключей RSA, DH и др.

1. **Baillie–PSW тест (комбинированный)**  
   Комбинация Миллера–Рабина и теста Люка.  
   ▪️ Неизвестны составные числа, которые его проходят.  
   ▪️ Используется в библиотеке GMP и системах вроде OpenSSL.

**📌 Применение:**

* Генерация криптографических ключей.
* Быстрая фильтрация больших чисел.
* Подходят для очень больших чисел (> 1000 бит).

**🔹 30. Детерминированные алгоритмы проверки чисел на простоту**

**Детерминированные алгоритмы** — дают точный ответ: **либо число простое, либо составное**.

🔸 **Основные алгоритмы**:

1. **Решето Эратосфена**  
   ▪️ Эффективный способ найти все простые ≤ N.  
   ▪️ Не подходит для проверки одного большого числа.
2. **Полный перебор делителей до n\sqrt{n}n​**  
   ▪️ Медленный: O(n)O(\sqrt{n})O(n​)  
   ▪️ Используется в обучающих целях и при проверке малых чисел.
3. **AKS алгоритм (Agrawal–Kayal–Saxena, 2002)**  
   ▪️ Первый **детерминированный полиномиальный** алгоритм простоты.  
   ▪️ Проверяет:

(x−a)n≡xn−amod  n,x(x - a)^n \equiv x^n - a \mod n, x(x−a)n≡xn−amodn,x

для всех a∈Za \in \mathbb{Z}a∈Z  
▪️ Сложность: O(log⁡cn)O(\log^c n)O(logcn), c∼6c \sim 6c∼6  
▪️ Не используется на практике из-за высокой константы.

1. **Pocklington's test, Lucas test**  
   ▪️ Теоретически детерминированы, но требуют знания факторизации n−1n-1n−1 или n+1n+1n+1.

**31. Общие сведения о двухключевых криптосистемах**

**Асимметричная криптографическая система (КГС)** - это система шифрования, в которой асимметричным образом используются ключи двух видов:

1. **Открытые ключи (ОК)** - общедоступны, задают процесс зашифрования направляемых в адрес участника протокола сообщений
2. **Закрытые ключи (ЗК)** - хранятся в тайне участником протокола, задают процесс расшифрования направляемых в его адрес сообщений

**Основа асимметричных систем** - односторонние функции с секретом:

**Односторонняя функция** f, осуществляющая отображение X→Y, удовлетворяет условиям:

* для x ∈ X легко вычислить y = f(x), y ∈ Y
* почти для любого y ∈ Y найти f⁻¹(y) вычислительно невозможно

**Односторонняя функция с секретом** fₖ удовлетворяет условиям:

* для x ∈ X легко вычислить y = fₖ(x), y ∈ Y
* при известном k для любого y ∈ Y легко вычислить x = fₖ⁻¹(y)
* почти для всех k и почти для всех y нахождение x = fₖ⁻¹(y) вычислительно невозможно без знания параметра k

**32. Криптографические системы с открытым ключом, основанные на сложности разложения целых чисел**

**Система RSA** (1977 год, Ривест, Шамир, Адлеман):

**Принцип работы:**

* Получатель выбирает два больших простых числа p и q
* Вычисляет N = p·q (открытый ключ)
* Выбирает e ∈ Z: НОД(e, φ(N)) = 1
* Вычисляет d = e⁻¹ mod φ(N) (закрытый ключ)
* Открытый ключ: (N, e)
* Закрытый ключ: (p, q, d)

**Шифрование:** c = mᵉ mod N **Расшифрование:** m = cᵈ mod N

**Система Рабина:**

* Выбираются два больших простых числа p ≡ q ≡ 3 (mod 4)
* n = p·q (открытый ключ)
* p, q - секретный ключ
* Шифрование: c = m² mod n
* Расшифрование: извлечение квадратного корня из c

**Вероятностное шифрование:**

* Для каждого бита mᵢ = 0 выбирается случайное четное число xᵢ < N
* Для каждого бита mᵢ = 1 выбирается случайное нечетное число xᵢ < N
* Вычисляется cᵢ = (xᵢᵉ) mod N

**33. Криптографические системы с открытым ключом, основанные на сложности дискретного логарифмирования в конечных полях**

**Система Эль-Гамаля:**

* Выбирается простое число p и целое число g
* У пользователя A есть секретный ключ a и открытый ключ y = gᵃ (mod p)
* Пользователь B выбирает случайное число k < p
* Вычисляет: y₁ = gᵏ mod p, y₂ = m ⊕ (yᵏ mod p)
* Отправляет пару (y₁, y₂)
* Расшифрование: m = (y₁ᵃ mod p) ⊕ y₂

**Система Диффи-Хеллмана:**

* Выбирается большое простое число p и первообразный корень α < p
* Каждый абонент выбирает секретный ключ x
* Вычисляет открытый ключ y = αˣ (mod p)
* Общий секретный ключ: Z\_AB = (y\_B)^x\_A = (y\_A)^x\_B = α^(x\_A·x\_B) (mod p)

**34. Криптографические системы с открытым ключом, основанные на сложности дискретного логарифмирования в группе точек эллиптической кривой**

**Общие параметры системы**

**Доменные параметры:**

* Простое число p > 2²⁵⁵ — модуль ЭК
* Эллиптическая кривая E с коэффициентами a, b ∈ GF(p)
* Целое число m = #E(GF(p)) — порядок группы точек ЭК
* Простое число q — порядок циклической подгруппы: m = nq, n ≥ 1, 2²⁵⁴ < q < 2²⁵⁶
* Точка P ∈ E(GF(p)) с координатами (x\_p, y\_p): P ≠ O, qP = O
* Хэш-функция h(·): V∞ → V₂₅₆ согласно ГОСТ Р 34.11

**Протокол распределения ключей (аналог Диффи-Хеллмана)**

| **Этап** | **Действия** |
| --- | --- |
| 1 | A генерирует секретный ключ n\_A и вычисляет P\_A = n\_A · G |
| 2 | B генерирует секретный ключ n\_B и вычисляет P\_B = n\_B · G |
| 3 | A передает B свой открытый ключ P\_A |
| 4 | B передает A свой открытый ключ P\_B |
| 5 | A вычисляет ключ K\_A = n\_A · P\_B = n\_A · n\_B · G |
| 6 | B вычисляет ключ K\_B = n\_B · P\_A = n\_B · n\_A · G |
| 7 | K\_A = K\_B = K\_C — сеансовый ключ сгенерирован |

**Алгоритм формирования цифровой подписи**

**Исходные данные:** сообщение M ∈ V∞, ключ подписи d

**Алгоритм:**

1. Вычислить значение хэш-функции: h = h(M)
2. По двоичному значению h вычислить целое число α и определить e ≡ α (mod q). Если e = 0, то определить e = 1
3. Сгенерировать случайное число k: 0 < k < q
4. Вычислить точку C = kP и определить r ≡ x\_C (mod q). Если r = 0, вернуться к шагу 3
5. Вычислить s ≡ (rd + ke) (mod q). Если s = 0, вернуться к шагу 3
6. Вычислить двоичные векторы r̄ и s̄

**Выход:** ЭЦП ζ = (r̄ || s̄)

**Алгоритм проверки цифровой подписи**

**Исходные данные:** сообщение M, подпись ζ, открытый ключ Q

**Алгоритм:**

1. По полученной подписи ζ = (r̄ || s̄) вычислить r и s. Проверить: 0 < r < q, 0 < s < q
2. Вычислить h = h(M)
3. Вычислить e ≡ α (mod q). Если e = 0, то e = 1
4. Вычислить v = e⁻¹ (mod q)
5. Вычислить z₁ = sv (mod q), z₂ = -rv (mod q)
6. Вычислить точку C = z₁P + z₂Q и определить R = x\_C (mod q)
7. Если R = r, то подпись верна, иначе неверна

**Математическая основа стойкости**

Стойкость системы основана на сложности решения задачи дискретного логарифмирования в группе точек эллиптической кривой (ECDLP): по известным точкам P и Q найти число k такое, что Q = kP.

**Сложность методов анализа**

* **Метод Полларда:** O(√q)
* **Метод Сильвера-Полига-Хеллмана:** O(log ζ)
* **Метод "встречи посередине":** O(√q)

**35. Классификация алгоритмов и методов факторизации чисел**

**По вычислительной сложности:**

1. **Алгоритмы с экспоненциальной сложностью:**
   * Метод Ферма
   * Алгоритм Полларда-Штрассена (ρ-метод)
   * P+1 метод Уильямса
   * ρ-метод Полларда
   * Метод Шэнкса
   * Метод Шермана-Лемана
2. **Алгоритмы с субэкспоненциальной сложностью:**
   * Алгоритм Диксона
   * Алгоритм Бриллхарта-Моррисона
   * Алгоритм квадратичного решета
   * Алгоритм решета числового поля
3. **Алгоритмы с полиномиальной сложностью:**
   * Алгоритм Ленстры
   * Алгоритм Шнорра-Ленстры
   * Алгоритм Ленстры-Померанса

**36. Требования, предъявляемые при выборе криптографических параметров асимметричных систем шифрования**

**Методика выбора ЭК над полем GF(q)**

**Шаг 1:** Определить характеристику p конечного поля GF(p^k) и степень расширения k в соответствии с исходными данными.

**Шаг 2:** Выбрать уравнение ЭК в соответствии с характеристикой p:

* Для p = 2: y² + ay = x³ + bx + c или y² + xy = x³ + ax² + b
* Для p = 3: y² = x³ + ax + b или y² = x³ + ax² + b
* Для p ≠ 2, p ≠ 3: y² = x³ + ax + b

**Шаг 3:** Сгенерировать коэффициенты уравнения ЭК a, b ∈ GF(p).

**Шаг 4:** Определить порядок группы #E(GF(q)).

* Если коэффициенты a (при p ≡ 2 (mod 6)) или b (при p ≡ 3 (mod 4)) равны 0, то #E(GF(q)) = p + 1

**Шаг 5:** Проверить невыполнимость условия делимости: #E(GF(q)) не должно делиться на (q^i - 1) для i = 1, 2, ..., 30 Если условие делимости выполняется, вернуться к шагу 3.

**Шаг 6:** Получены параметры: характеристика поля p, уравнение ЭК с порядком группы, удовлетворяющим показателям стойкости.

**Критерии выбора порядка группы точек ЭК**

**Для кривой y² ≡ x³ + b (mod p), где p ≡ 1 (mod 6):**

1. Если b — кубический и квадратический вычет одновременно: #E(GF(p)) = 6m
2. Если b — только кубический вычет: #E(GF(p)) = 3m
3. Если b — только квадратический вычет: #E(GF(p)) = 2m
4. Если b — кубический и квадратический невычет: #E(GF(p)) = 6m + 1

**Для кривой y² = x³ + ax (mod p):**

1. Если (p-a) при p ≡ 1 (mod 4) — квадратический невычет: #E(GF(p)) = 2m
2. Если (p-a) — квадратический вычет: #E(GF(p)) = 4m

где m — простое число.

**Параметры, влияющие на стойкость**

**Основные факторы:**

* Вид конечного поля
* Характеристика поля и его расширения
* Уравнение ЭК
* Порядок циклической подгруппы точек ЭК
* Генератор подгруппы точек ЭК

**Требования к размерам:**

* Простое число p > 2²⁵⁵
* Порядок подгруппы q: 2²⁵⁴ < q < 2²⁵⁶
* Избегать аномальных и суперсингулярных кривых
* Проверять MOV-редукцию и других известных атак

**Рекомендации по безопасности**

1. **Избегать слабых кривых:**
   * Аномальные кривые (#E(GF(p)) = p)
   * Суперсингулярные кривые
   * Кривые с малым порядком вложения
2. **Проверять стойкость к атакам:**
   * MOV-атака
   * Атака Фрея-Рука
   * Атака на аномальные кривые
3. **Использовать стандартизированные кривые** или тщательно верифицированные параметры согласно криптографическим стандартам.

**37. Алгоритмы быстрого разложения чисел на множители**

**Алгоритм пробных делений:**

* Последовательно проверяет делимость на малые простые числа
* Эффективен для чисел малой разрядности
* Для 18-разрядного числа проверяет простые числа меньше 10⁹

**Алгоритм Ферма:**

* Представляет N в виде разности квадратов: N = x² - y²
* Ищет x и y такие, что x² - N = y²
* Если найдены, то N = (x-y)(x+y)

**Алгоритм Ленстры:**

* Основан на теореме о существовании не более 11 делителей
* Сложность: O(n^(1/3) log n)
* Использует последовательность {(aᵢ, bᵢ, cᵢ)} для поиска делителей

**38. Алгоритм разложения чисел субэкспоненциальной сложности**

**Метод квадратичного решета:**

**Основные понятия:**

* **Факторная база B** = {p₁, p₂, ..., pₖ} - множество небольших простых чисел
* **B-число** - число b, для которого b² mod N разлагается на простые из B

**Алгоритм:**

1. Формируется факторная база простых чисел ≤ M
2. Ищутся числа xᵢ такие, что yᵢ = xᵢ² - N является B-числом
3. Для каждого B-числа создается вектор показателей степеней
4. Ищутся линейные зависимости между векторами по модулю 2
5. Формируется произведение x и y
6. Если x² ≡ y² (mod N) и x ≢ ±y (mod N), то НОД(x±y, N) дает делитель

**Метод решета числового поля:**

1. **Выбор многочленов:** выбираются f(X) и g(X) с общим корнем m mod N
2. **Просеивание:** поиск пар (a,b) с гладкими значениями
3. **Линейная алгебра:** поиск квадратов идеалов
4. **Вычисление корней:** получение X и Y с X² ≡ Y² (mod N)

**Сравнительная таблица размеров ключей**

| **Симметричный алгоритм** | **Схемы на ЭК** | **RSA/DSA** |
| --- | --- | --- |
| 56 бит | 112 бит | 512 бит |
| 80 бит | 160 бит | 1024 бит |
| 112 бит | 224 бит | 2048 бит |
| 128 бит | 256 бит | 3072 бит |
| 192 бит | 384 бит | 7680 бит |
| 256 бит | 512 бит | 15360 бит |

**Области применения RSA:**

* Стандарты: ISO 9796, ANSI X 9.31, ITU-T X.509
* Протоколы: S/MIME, IPSec, TLS, HTTPS, SSL
* Программное обеспечение: криптопровайдеры ОС, PGP, браузеры
* Аппаратура: смарт-карты, криптографическое оборудование

**39. Общие сведения, классификация алгоритмов и методов дискретного логарифмирования**

**Формализация задачи дискретного логарифмирования**

**Дискретное логарифмирование (DLOG)** — задача обращения функции g^x в некоторой конечной мультипликативной группе G.

**Классификация алгоритмов дискретного логарифмирования**

**По сложности:**

1. **Алгоритмы с экспоненциальной сложностью**
2. **Алгоритмы с субэкспоненциальной сложностью**

**По специфике применения:**

1. **Универсальные алгоритмы** (работают в произвольных группах)
2. **Специализированные алгоритмы** (для конкретных типов групп)

**По принципу работы:**

1. **Алгоритмы полного перебора**
2. **Алгоритмы больших и малых шагов**
3. **Алгоритмы на основе факторизации**
4. **Алгоритмы решета**

**40. Методы дискретного логарифмирования с экспоненциальной сложностью**

**Метод полного перебора**

**Пример:** Решить 3^x ≡ 13 mod 17

Выписываем таблицу всех степеней числа 3:

* 3¹ ≡ 3
* 3² ≡ 9
* 3³ ≡ 10
* 3⁴ ≡ 13 ← **Ответ: x = 4**

**Алгоритмы решения в произвольной мультипликативной группе**

В алгоритме используется таблица, состоящая из O(√n) пар элементов и выполняется O(√n) умножений.

**Алгоритмы в кольце вычетов по простому модулю**

Для уравнения a^x ≡ b mod p, если a является образующим элементом группы Z/pZ, то уравнение имеет решение при любых b.

Решение можно находить по формуле: x = Σ(i=1 to p-1) (1 - a^i)⁻¹ · b^i mod p

**Алгоритм больших и малых шагов**

**Временная асимптотическая оценка сложности:** O(√M log M)

**Идея:** Положим x = AB - Q, где A — заранее выбранная константа, зависящая от p.

Тогда из a^(AB-Q) = b (mod p) получаем a^(AB) = b·a^Q (mod p)

**Алгоритм Полига-Хеллмана**

**Исходные данные:** a^x ≡ b mod p и известно разложение p - 1 на простые множители: p - 1 = Π(i=1 to s) q\_i^α\_i

**Идея алгоритма:**

1. Достаточно найти x по модулям q\_i^α\_i для всех i
2. Для нахождения x по каждому модулю нужно решить сравнение: a^((p-1)/q\_i^α\_i)·x ≡ b^((p-1)/q\_i^α\_i) mod p

**ρ-метод Полларда для дискретного логарифмирования**

**Идея алгоритма:** Рассматриваются последовательности {u\_i}, {υ\_i}, {z\_i}, i ∈ N, определённые специальным образом с разбиением на три области.

**Сложность алгоритма:** O(√p)

Ищется i, для которого z\_i = z\_2i, что даёт уравнение: b^(u\_2i - u\_i) ≡ a^(υ\_i - υ\_2i) mod p

Если при этом НОД(u\_2i - u\_i, p-1) = 1, то: x ≡ log\_a b ≡ (u\_2i - u\_i)⁻¹(υ\_i - υ\_2i) mod p-1

**41. Сложность алгоритмов дискретного логарифмирования**

**Алгоритм Адлемена**

**Сложность:** O(exp(c(log p log log p)^d)) арифметических операций, где c и 0≤d<1 — некоторые константы.

**Этапы алгоритма:**

1. **Этап 1:** Сформировать факторную базу из всех простых чисел q: q ≤ B = e^(const)
2. **Этап 2:** Найти натуральные числа r\_i такие, что a^r\_i mod p раскладывается по факторной базе
3. **Этап 3:** Решить систему линейных уравнений относительно дискретных логарифмов элементов факторной базы
4. **Этап 4:** Найти значение r для специального разложения
5. **Этап 5:** Найти дискретные логарифмы промежуточных простых чисел
6. **Этап 6:** Определить искомый дискретный логарифм

**Оценка сложности:** O(exp(c(log p log log p)^(1/2))) арифметических операций

**Алгоритм COS**

**Этапы:**

1. Формируем множество H + c, где H = [1/2 + c·√(L^(1/2))], c < L^(1/2)
2. Ищем пары c₁, c₂ такие, что (H + c₁)(H + c₂) раскладывается по факторной базе
3. Решаем систему линейных уравнений
4. Находим значение w для специального разложения
5. Находим логарифмы больших простых чисел
6. Получаем ответ

**Сложность:** O(exp((log p log log p)^(1/2))) арифметических операций. В этом алгоритме константа c = 1, d = 1/2.

**Решето числового поля**

Алгоритм, предложенный Д. Гордоном в 1993 году, имел эвристическую сложность: O(exp(c(log p log log p)^(1/3)))

**Наилучшие параметры:**

* c = (9√2 + 2√6)^(1/3)/3 ≈ 1.902
* d = 1/3

**Для чисел специального вида:**

* c ≈ 1.00475
* d = 2/5

Проанализировав прикрепленные лекции по криптографии, я могу подготовить развернутые ответы на следующие вопросы, для которых нашел информацию в материалах:

**42. Классификация методов криптоанализа асимметричных систем шифрования**

Согласно лекционному материалу, методы криптоанализа асимметричных систем шифрования можно классифицировать по нескольким критериям:

**По математической основе:**

* Методы, основанные на факторизации больших чисел (для RSA и подобных систем)
* Методы, основанные на решении задачи дискретного логарифмирования (для систем типа ElGamal, DSA)
* Методы, основанные на решении задачи дискретного логарифмирования в группе точек эллиптической кривой

**По вычислительной сложности:**

* Алгоритмы с экспоненциальной сложностью
* Алгоритмы с субэкспоненциальной сложностью
* Алгоритмы с полиномиальной сложностью

**По типу атаки:**

* Атаки на основе математических свойств используемых функций
* Атаки на основе слабостей в реализации
* Атаки на основе побочных каналов

**43. Алгоритмы разложения чисел на множители**

Из лекции 9 следует подробная классификация алгоритмов факторизации:

**АЛГОРИТМЫ С ЭКСПОНЕНЦИАЛЬНОЙ СЛОЖНОСТЬЮ:**

1. **Алгоритм "пробных делений"**
   * Принцип: последовательная проверка деления на простые числа
   * Пример: N = 9438 = 2·3·11·11·13
   * Эффективен только для чисел малой разрядности
   * Для 18-разрядного числа (60 бит) нужно проверить простые числа меньше 10^9
2. **Алгоритм Ферма**
   * Основан на представлении N = x² - y²
   * Ищет такие x и y, что N = (x-y)(x+y)
   * Пример: N = 5959, √N ≈ 77.194
   * Эффективен для чисел, имеющих близкие по величине множители
3. **Алгоритм Ленстры**
   * Использует алгоритм Евклида для поиска делителей
   * Сложность: O(n^(1/3) log n)
   * Работает с последовательностями (aᵢ, bᵢ, cᵢ) и поиском линейных зависимостей
4. **Алгоритм Полларда-Штрассена (ρ-метод)**
   * Основан на парадоксе дней рождения
   * Ищет коллизии в последовательности xᵢ₊₁ = f(xᵢ) mod N
5. **P+1 метод Уильямса**
   * Модификация ρ-метода Полларда
   * Использует свойства последовательности Лукаса
6. **Метод Шэнкса**
   * Метод "больших и малых шагов"
   * Временная сложность: O(√M log M)
7. **Метод Шермана-Лемана**
   * Комбинирует различные подходы к факторизации

**АЛГОРИТМЫ С СУБЭКСПОНЕНЦИАЛЬНОЙ СЛОЖНОСТЬЮ:**

1. **Алгоритм Диксона**
   * Первый алгоритм субэкспоненциальной сложности
   * Использует факторную базу и поиск квадратичных вычетов
2. **Алгоритм Бриллхарта-Моррисона**
   * Развитие метода Диксона
   * Более эффективный отбор кандидатов
3. **Метод квадратичного решета**
   * Использует факторную базу B = {p₁, p₂, ..., pₖ}
   * B-число: число b такое, что b² mod N раскладывается по факторной базе
   * Поиск линейных зависимостей между векторами показателей степеней
   * Формирование x² ≡ y² mod N для нахождения НОД(N, x±y)
4. **Алгоритм Ленстры-Померанса**
   * Оптимизированная версия квадратичного решета
5. **Алгоритм квадратичного решета числового поля**
   * Использует алгебраические числовые поля
   * Более эффективен для больших чисел
6. **Решето числового поля (NFS)**
   * Наиболее эффективный известный алгоритм факторизации
   * Этапы:
     + Выбор многочленов f(X) и g(X)
     + Просеивание для поиска гладких пар (a,b)
     + Методы линейной алгебры
     + Вычисление квадратных корней
   * Использует гомоморфизмы φₐ и φβ для получения X² ≡ Y² mod N

**АЛГОРИТМЫ С ПОЛИНОМИАЛЬНОЙ СЛОЖНОСТЬЮ:**

1. **Алгоритм Шнорра-Ленстры**
   * Теоретически полиномиальный, но практически неэффективен
   * Использует свойства эллиптических кривых

**44. Сравнительная сложность основных методов криптоанализа криптосистем с открытым ключом**

**ФАКТОРИЗАЦИЯ ЧИСЕЛ:**

*Экспоненциальная сложность:*

* Алгоритм пробных делений: O(√N)
* Алгоритм Ферма: O(√N)
* Алгоритм Ленстры: O(N^(1/3) log N)
* ρ-метод Полларда: O(N^(1/4))

*Субэкспоненциальная сложность:*

* Квадратичное решето: O(exp(c√(ln N ln ln N)))
* Решето числового поля: O(exp(c(ln N)^(1/3)(ln ln N)^(2/3)))

**ДИСКРЕТНОЕ ЛОГАРИФМИРОВАНИЕ:**

Из лекции 10 следует классификация алгоритмов дискретного логарифмирования:

*Экспоненциальная сложность:*

* Метод полного перебора: O(p-1)
* Алгоритм больших и малых шагов: O(√M log M)
* Алгоритм Полига-Хеллмана: зависит от факторизации p-1
* ρ-метод Полларда: O(√p)

*Субэкспоненциальная сложность:*

* Алгоритм Адлемена: O(exp(c(log p log log p)^(1/2)))
* Алгоритм COS: O(exp((log p log log p)^(1/2)))
* Решето числового поля: O(exp(c(log p)^(1/3)(log log p)^(2/3)))

**СРАВНИТЕЛЬНЫЙ АНАЛИЗ:**

1. **Для чисел общего вида:**
   * Факторизация: NFS наиболее эффективен
   * Дискретное логарифмирование: решето числового поля
2. **Для чисел специального вида:**
   * Возможны более эффективные алгоритмы
   * Параметры сложности: c ≈ 1.00475, d = 2/5
3. **Практические соображения:**
   * Константы в оценках сложности критически важны
   * Требования к памяти могут быть ограничивающим фактором
   * Возможность параллелизации влияет на практическую эффективность

**РЕКОМЕНДАЦИИ ДЛЯ КРИПТОГРАФИЧЕСКИХ СИСТЕМ:**

1. Размер ключей должен выбираться с учетом лучших известных алгоритмов атак
2. Необходимо учитывать развитие вычислительных технологий
3. Для RSA рекомендуется использовать ключи длиной не менее 2048 бит
4. Для систем на основе дискретного логарифмирования требуются сопоставимые по сложности параметры