**ImageCipher.java**

**1. Класс ImageCipher и метод bytesToHex:**

* **Назначение:** Преобразует массив байтов в шестнадцатеричную строку.
* **Шаги:**
  1. **Итерация по байтам:** Для каждого байта в массиве:
     + Применяется маска b & 0xFF, чтобы обработать байт как беззнаковый.
     + Конвертация в HEX: Integer.toHexString().
     + Если HEX-строка короче 2 символов, добавляется ведущий ноль.
  2. **Результат:** Строка, где каждый байт представлен двумя HEX-символами (например, 1A, FF).

**2. Метод encrypt:**

* **Цель:** Встраивание текста в изображение.
* **Шаги:**
  1. **Добавление флага:** К тексту добавляется префикс $enc, чтобы отметить начало сообщения.
  2. **Конвертация в байты:**
     + Текст с флагом преобразуется в байты (UTF-8).
     + Изображение читается в массив байтов.
  3. **Объединение байтов:**
     + Создается новый массив combinedBytes, содержащий байты изображения и сообщения.
     + Данные копируются через System.arraycopy().
  4. **Сохранение:** Объединенный массив записывается обратно в файл изображения.

**Пример:**  
Исходный файл (100 байт) + сообщение "Hello" (5 байт) = новый файл (105 байт).

**3. Метод decrypt:**

* **Цель:** Извлечение скрытого текста из изображения.
* **Шаги:**
  1. **Чтение файла:** Весь файл конвертируется в HEX-строку.
  2. **Поиск флага:**
     + Флаг $enc преобразуется в HEX.
     + HEX-строка файла разбивается на части по HEX-флагу.
  3. **Обработка частей:**
     + Игнорируется часть до первого флага.
     + Каждая последующая часть конвертируется из HEX в байты.
     + Байты декодируются в строку (UTF-8) и обрезаются (trim()).
  4. **Результат:** Массив строк, найденных после каждого вхождения флага.

**Пример:**  
Если файл содержит два флага $enc, вернется массив из двух сообщений.

Этот код выполняет **шифрование сообщения с помощью транспозиционного шифра и скрывает его в аудиофайле** методом LSB (Least Significant Bit). Вот пошаговое объяснение:

**AudioCipherEncrypt.java**

**1. Метод encrypt(String inputWav, String textMessage)**

* **Цель:** Координирует процесс шифрования и скрытия сообщения.
* **Шаги:**
  1. Вызывает transpositionEncrypt(textMessage) для шифрования текста перестановкой.
  2. Передает зашифрованный текст в hideMessageInWav(inputWav, ciphertext) для встраивания в аудиофайл.

**2. Транспозиционное шифрование (transpositionEncrypt)**

* **Цель:** Переставить символы сообщения по таблице 3×N.
* **Шаги:**
  1. **Создание таблицы:**
     + Число строк lines = длина сообщения / 3 (с округлением вверх).
     + Таблица table[lines][3] заполняется символами сообщения построчно. Недостающие элементы заполняются пробелами.
     + Пример для "HelloWorld":

Copy

H e l

l o W

o r l

d (пробелы)

* 1. **Чтение по столбцам:**
     + Символы считываются колонка за колонкой: H l o d, e o r , l W l .
     + Результат: Hlodeo rlWl .

**3. Сокрытие сообщения в аудио (hideMessageInWav)**

* **Цель:** Спрятать зашифрованный текст в WAV-файле, заменяя младшие биты аудиоданных.
* **Шаги:**
  1. **Чтение аудиофайла:**
     + Загружаются аудиоданные в массив frames (байты).
  2. **Подготовка сообщения:**
     + Добавляется маркер конца "=====".
     + Сообщение конвертируется в байты UTF-8.
     + Проверка: длина сообщения \* 8 ≤ длина frames (1 бит сообщения на 1 байт аудио).
  3. **Встраивание битов:**
     + Каждый бит сообщения записывается в **младший бит** аудиобайта:

java

Copy

frames[i] = (frames[i] & 0xFE) | bit; // 0xFE = 11111110 (обнуление младшего бита)

* 1. **Сохранение измененного аудио:**
     + Перезаписывается исходный файл новыми данными.

**4. Пример работы**

* **Исходное сообщение:** "Secret"
* **После транспозиции:** "S ce r et " (пример)
* **В аудиофайле:**
  + Каждый бит "S ce r et=====" встраивается в младшие биты аудиоданных.

**5. Особенности и ограничения**

* **Формат аудио:** Должен быть несжатым (например, WAV с PCM).
* **Перезапись файла:** Исходный файл изменяется (нет резервной копии).
* **Маркер конца:** "=====" помогает при извлечении сообщения.
* **Безопасность:** LSB-стеганография не является криптографически стойкой.

**6. Как извлечь сообщение?**

Для декодирования нужно:

1. Прочитать младшие биты аудиоданных.
2. Преобразовать биты в байты UTF-8.
3. Найти маркер "=====" для определения конца сообщения.
4. Применить обратную транспозицию к полученному тексту.

**AudioCipherDecrypt.java**

**1. Общая цель**

Код предназначен для извлечения и расшифровки сообщения, скрытого в аудиофайле (WAV) с использованием двух методов:

* **Стеганография LSB**: Сообщение спрятано в младших битах аудиоданных.
* **Транспозиционный шифр**: Перестановка символов по определенному алгоритму.

**2. Метод transpositionDecrypt(String ciphertext)**

**Цель**: Расшифровать текст, зашифрованный транспозиционным шифром.

**Шаги**:

1. **Инициализация**:
   * Создается список converted для хранения расшифрованных строк.
   * Вычисляется количество строк таблицы: lines = длина\_текста / 3 (с округлением вверх).
   * Исходный текст разбивается на массив символов textL.
2. **Создание таблицы**:
   * Создается двумерный массив table[lines][3].
   * Таблица заполняется **по столбцам** (а не по строкам):

java

Copy

for (столбец i от 0 до 2) {

for (строка j от 0 до lines) {

table[j][i] = textL[index];

index++;

}

}

1. Это обратная операция к шифрованию, где текст записывался по строкам, а считывался по столбцам.
2. **Сборка результата**:
   * Каждая строка таблицы объединяется в строку.
   * Все строки собираются в одну строку decryptedText.

**Пример**:

* Зашифрованный текст: "ABC123XYZ" (длина 9).
* Таблица 3x3:

Copy

A 1 X

B 2 Y

C 3 Z

* Расшифрованный текст: "A1XB2YC3Z" (читается по строкам).

**3. Метод extractMessageFromWav(String inputWavPath)**

**Цель**: Извлечь скрытое сообщение из аудиофайла.

**Шаги**:

1. **Чтение аудиофайла**:
   * Файл открывается через AudioInputStream.
   * Все байты аудиоданных сохраняются в frames.
2. **Извлечение LSB-битов**:
   * Для каждого байта аудио извлекается младший бит: b & 1.
   * Пример: байт 01100101 → бит 1.
3. **Формирование байтов сообщения**:
   * Каждые 8 бит объединяются в байт:

java

Copy

for (i от 0; шаг 8) {

currentByte = (бит[i] << 7) | (бит[i+1] << 6) | ... | бит[i+7];

extractedBytes.add(currentByte);

}

1. **Преобразование в строку**:
   * Список байтов extractedBytes конвертируется в строку decodedMessage1.
2. **Поиск маркера конца**:
   * Ищется подстрока "=====". Если найдена, все после нее отбрасывается.
3. **Расшифровка транспозицией**:
   * Результат передается в transpositionDecrypt.

**4. Метод decrypt(String outputWav)**

**Цель**: Публичный метод для запуска процесса расшифровки.

* Просто вызывает extractMessageFromWav(outputWav).

**5. Вспомогательный метод toByteArray**

* Конвертирует список Integer в массив byte.

**Пример полного цикла**

1. **Шифрование (предполагается)**:
   * Исходный текст: "HELLO".
   * Транспозиция: Записывается в таблицу по строкам, считывается по столбцам → "HLOEL".
   * Стеганография: Каждый бит "HLOEL" встраивается в LSB аудиофайла.
2. **Расшифровка**:
   * Из аудио извлекаются биты → "HLOEL=====".
   * Удаляется маркер → "HLOEL".
   * Транспозиция: Заполнение таблицы по столбцам → исходное "HELLO".