Варіант 1. Зашифрувати вхідний символьний потік по наступному алгоритму: замінити кожний символ на такий, код якого більший на значення коду ключового символу. При виконанні завдання:

- Реалізувати метод, який реалізує шифрування
- Реалізувати метод, який реалізує дешифрування.
- Використати для розв'язання задачі потоки типу FilerOutputStream-FilterInputStream/FilterWriter-FilteredReader.

Варіант 2. Реалізувати функцію завантаження вмісту файлу у реляційну таблицю (наприклад, завантаження фото із файлу в БД).

Для роботи з БД використовувати JDBC-драйвер. Для роботи з файлами та при роботі із великими даними із JDBC використовувати Input/OutputStreams.

До вибору БД вимог не пред'являється – це може, наприклад, бути Oracle, MySQL, PostgreSQL.

Нижче дано приклад коду для завантаження фото із БД в файл. Як БД використано Oracle (від вибоу БД залежать типи колонок в таблиці).

---------- table ----------

```
create table streamexample (NAME varchar2 (256), GIFDATA long raw);
```

---------- Java code ----------

```
ResultSet rset = stmt.executeQuery ("select GIFDATA from streamexample where NAME='LESLIE'");

// get first row
if (rset.next())
{
   // Get the GIF data as a stream from Oracle to the client
   InputStream gif_data = rset.getBinaryStream (1);
  try
  {
    FileOutputStream file = null;
    file = new FileOutputStream ("leslie.gif");
    int chunk;
    while ((chunk = gif_data.read()) != -1)
      file.write(chunk);
  }
  catch (IOException e)
  {
    String err = e.toString();
    System.out.println(err);
  }
  finally
  {
```

```
        if file != null()
            file.close();
    }
}
```