Задание 1

Создайте мини приложение, в котором я отправляю сообщение серверу.

Мой класс сообщения:

MessageData.java

- String userName;
- String messageText;
- Date sentDate; // java.util.Date

Сделайте два конструктора :)

Получается у меня есть два приложения: 1-сервер, 2-клиент

Сервер при запуске создает **ServerSocket** и начинает ожидать клиента. При подключении клиента, сервер создает поток вывода и ввода (**ObjectOutputStream**, **ObjectInputStream**), и принимает объект класса **MessageData** которую нам должен отправить клиент.
Клиент при запуске начинает запрашивать имя пользователя. Затем он подключается к серверу и выводит данное меню:

PRESS [1] TO SEND MESSAGE
Insert message text:
PRESS [2] TO EXIT

После ввода сообщения, наш клиент создает объект класса Message, настроив значение userName именем пользователя, значение messsageText текстом сообщения и sentDate текущим временем. Текущее время можно получить при создании нового объекта java.util.Date. Получается при создании нового объекта мы по умолчанию получим текущее время. Date date = new Date();
Метод toString() класса Date возвращает текстовое значение времени.

После отправки сообщения, на сервере мы должны отобразить данные сообщения клиента.

Например:
"Hello" from Olzhas at 04.04.2024 15:34

Задание 2

Создайте мини приложение клиент-сервера, где ваш клиент будет загружать данные списка класса **Book** с сервера.
Класс **Book**:
- int id;
- String name;
- String author;

Конструктор
Геттеры и сеттеры

Создайте специальный класс **PackageData** который будет как пакет передачи данных через сеть.
Параметры класса **PackageData**:
- String operationType;
- ArrayList<**Book**> books;
- Book book;

Получается, мы на серверной стороне имеем список книг, которые храним в сериализированном файле.
Наш клиент подключается к серверу и делает запрос на список книг, и сервер отправляет клиенту список через специальный класс **PackageData**.
Панель клиента выглядит таким образом:

PRESS 1 TO LIST BOOKS
PRESS 2 TO ADD BOOKS
PRESS 0 TO DISCONNECT FROM SERVER

Сделайте так, чтобы несколько клиентов могли одновременно подключаться к серверу. (Используйте потоки).