Разделить программу из лабораторной работы №5 на клиентский и серверный модули. Серверный модуль должен реализовывать все функции управления коллекцией в интерактивном режиме, кроме отображения текста в соответствии с сюжетом предметной области. Клиентский модуль должен запрашивать у сервера текущее состояние коллекции, генерировать сюжет, выводить его на консоль и завершать работу.

Хранящиеся в коллекции объекты должны иметь следующие характеристики:

* имя, название или аналогичный текстовый идентификатор;
* размер или аналогичный числовой параметр;
* характеристику, определяющую местоположение объекта на плоскости/в пространстве;
* время/дату рождения/создания объекта.

Если аналогичные характеристики уже есть, добавлять их не нужно.

**Необходимо выполнить следующие требования:**

* Коллекцию из ЛР №5 заменить на ее потокобезопасный аналог.
* Операции обработки объектов коллекции должны быть реализованы с помощью Stream API с использованием лямбда-выражений.
* Объекты между клиентом и сервером должны передаваться в сериализованном виде.
* Объекты в коллекции, передаваемой клиенту, должны быть отсортированы по умолчанию.
* Получив запрос, сервер должен создавать отдельный поток, который должен формировать и отправлять ответ клиенту.
* Клиент должен корректно обрабатывать временную недоступность сервера.
* Обмен данными между клиентом и сервером должен осуществляться по протоколу UDP.
* На стороне сервера должен использоваться датаграммы а на стороне клиента - сетевой канал.

Отчёт по работе должен содержать:

* Текст задания.
* Диаграмма классов разработанной программы (как клиентского, так и серверного приложения).
* Исходный код программы.
* Выводы по работе.

Вопросы к защите лабораторной работы:

* Сетевое взаимодействие - клиент-серверная архитектура, основные протоколы, их сходства и отличия.
* Протокол TCP. Классы Socket и ServerSocket.
* Протокол UDP. Классы DatagramSocket и DatagramPacket.
* Передача данных по сети. Сериализация объектов.
* Интерфейс Serializable. Объектный граф, сериализация и десериализация полей и методов.
* Многопоточные программы. Концепции. Класс Thread и интерфейс Runnable.
* Состояние потока. Синхронизация потока.
* Пакет java.util.concurrent. Интерфейс Lock и его реализации.
* Атомарные операции.
* Java Stream APi. Создание конвейеров. Промежуточные и терминальные операции.