**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №6 Вариант 21212

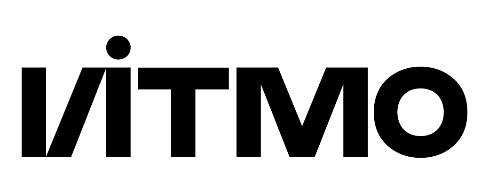
по дисциплине «**Программирование**»

Автор: Трусковский Георгий Александрович

Факультет: ПИИКТ

Группа: Р3114

Преподаватель: Наумова Н.А.



Санкт-Петербург 2024

Оглавление

[Задание 3](file:///C:\ITMO\1%20Курс\Прога\ProgLabs\lab5\Отчёт5Лаба.docx#_Toc166757688)

[Ход работы 5](file:///C:\ITMO\1%20Курс\Прога\ProgLabs\lab5\Отчёт5Лаба.docx#_Toc166757689)

[Исходный код: 5](file:///C:\ITMO\1%20Курс\Прога\ProgLabs\lab5\Отчёт5Лаба.docx#_Toc166757690)

[UML Диаграмма 5](file:///C:\ITMO\1%20Курс\Прога\ProgLabs\lab5\Отчёт5Лаба.docx#_Toc166757691)

[Вывод 7](file:///C:\ITMO\1%20Курс\Прога\ProgLabs\lab5\Отчёт5Лаба.docx#_Toc166757692)

# Задание

Разделить программу из [лабораторной работы №5](https://se.ifmo.ru/courses/programming#lab5) на клиентский и серверный модули. Серверный модуль должен осуществлять выполнение команд по управлению коллекцией. Клиентский модуль должен в интерактивном режиме считывать команды, передавать их для выполнения на сервер и выводить результаты выполнения.

**Необходимо выполнить следующие требования:**

* 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 размеру
* Клиент должен корректно обрабатывать временную недоступность сервера.
* Обмен данными между клиентом и сервером должен осуществляться по протоколу TCP
* Для обмена данными на сервере необходимо использовать **потоки ввода-вывода**
* Для обмена данными на клиенте необходимо использовать **сетевой канал**
* Сетевые каналы должны использоваться в неблокирующем режиме.

**Обязанности серверного приложения:**

* Работа с файлом, хранящим коллекцию.
* Управление коллекцией объектов.
* Назначение автоматически генерируемых полей объектов в коллекции.
* Ожидание подключений и запросов от клиента.
* Обработка полученных запросов (команд).
* Сохранение коллекции в файл при завершении работы приложения.
* Сохранение коллекции в файл при исполнении специальной команды, доступной только серверу (клиент такую команду отправить не может).

**Серверное приложение должно состоять из следующих модулей (реализованных в виде одного или нескольких классов):**

* Модуль приёма подключений.
* Модуль чтения запроса.
* Модуль обработки полученных команд.
* Модуль отправки ответов клиенту.

Сервер должен работать в **однопоточном** режиме.

**Обязанности клиентского приложения:**

* Чтение команд из консоли.
* Валидация вводимых данных.
* Сериализация введённой команды и её аргументов.
* Отправка полученной команды и её аргументов на сервер.
* Обработка ответа от сервера (вывод результата исполнения команды в консоль).
* Команду save из клиентского приложения необходимо убрать.
* Команда exit завершает работу клиентского приложения.

**Важно!**Команды и их аргументы должны представлять из себя объекты классов. Недопустим обмен "простыми" строками. Так, для команды add или её аналога необходимо сформировать объект, содержащий тип команды и объект, который должен храниться в вашей коллекции.

**Дополнительное задание:**  
Реализовать логирование различных этапов работы сервера (начало работы, получение нового подключения, получение нового запроса, отправка ответа и т.п.) с помощью **Java Util Logging**

**Отчёт по работе должен содержать:**

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

**Вопросы к защите лабораторной работы:**

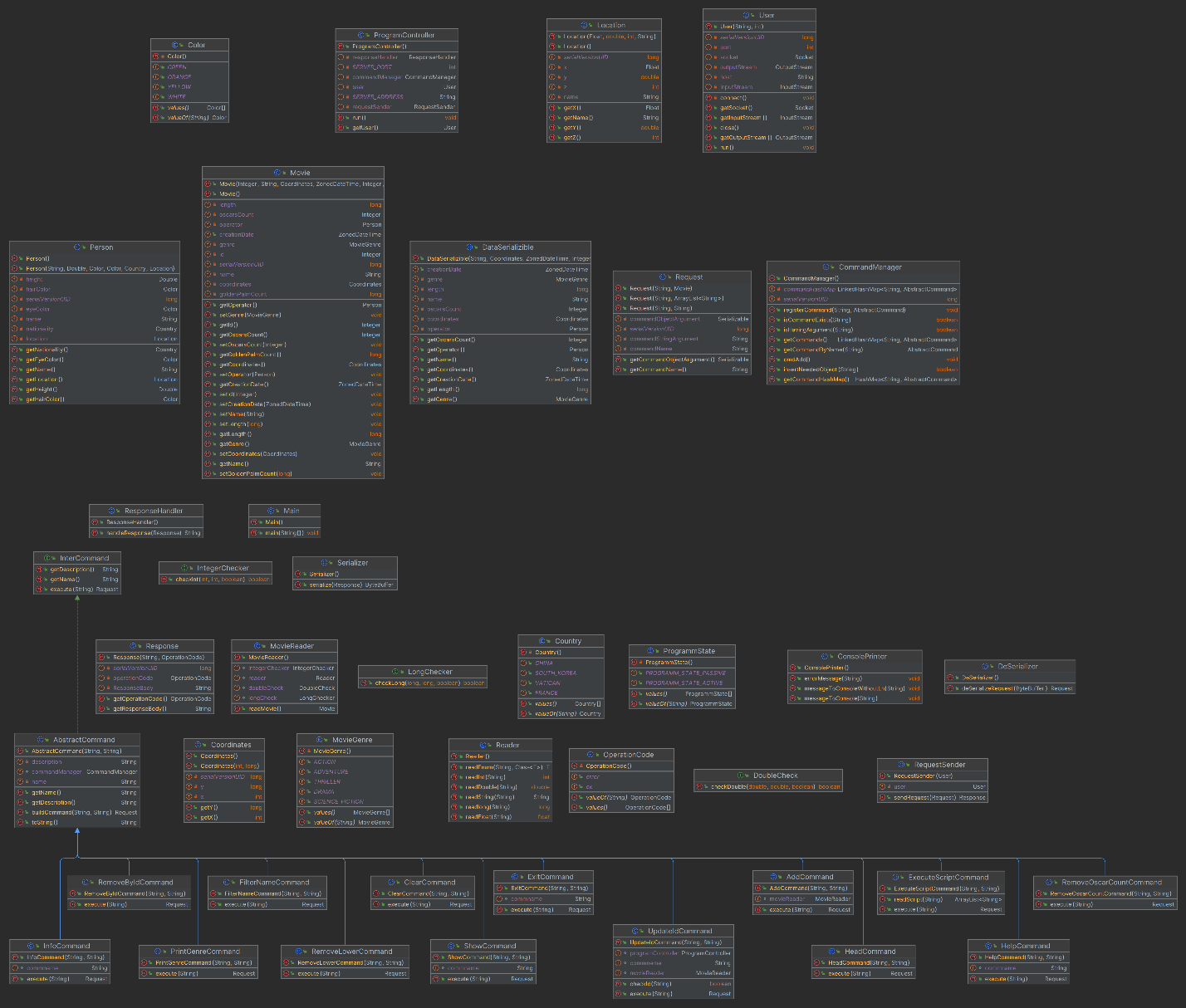
1. Сетевое взаимодействие - клиент-серверная архитектура, основные протоколы, их сходства и отличия.
2. Протокол TCP. Классы Socket и ServerSocket.
3. Протокол UDP. Классы DatagramSocket и DatagramPacket.
4. Отличия блокирующего и неблокирующего ввода-вывода, их преимущества и недостатки. Работа с сетевыми каналами.
5. Классы SocketChannel и DatagramChannel.
6. Передача данных по сети. Сериализация объектов.
7. Интерфейс Serializable. Объектный граф, сериализация и десериализация полей и методов.
8. Java Stream API. Создание конвейеров. Промежуточные и терминальные операции.
9. Шаблоны проектирования: Decorator, Iterator, Factory method, Command, Flyweight, Interpreter, Singleton, Strategy, Adapter, Facade, Proxy.

# Ход работы

## Исходный код:

<https://github.com/Sinchi1/ProgLabs/tree/main/Lab6>

## UML Диаграмма





# Вывод:

В ходе выполнения лабораторной работы, научился использовать несколько паттернов программирования. Также научился создавать клиент серверные приложения, работать с транспортными протоколами TCP/UDP, сериализовывать объекты.