

Федеральное государственное автономное образовательное учреждение высшего образования  
«Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

Направление системного и прикладного программного обеспечения

Отсчет по лабораторной работе № 6  
Дисциплина «Программирование»  
Вариант № 87698

Выполнила:  
Исаева Александра-Ирина Антоновна  
**Группа:** Р3109  
Преподаватели:  
Гаврилов Антон Валерьевич  
Мустафаева Айнур Вугар кызы

## Содержание

Задание .....	3
UML диаграмма команд.....	4
Исходный код .....	4
Выводы.....	4

## Задание

Разделить программу из [лабораторной работы №5](#) 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 умолчанию.
- 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 сетевой канал.
- Для обмена данными на клиенте необходимо использовать потоки ввода-вывода.
- 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 введенной 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 **Log4J2**

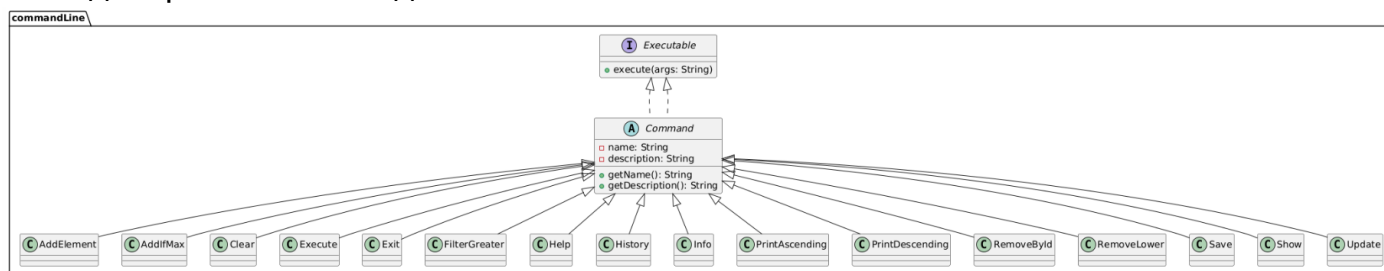
### Отчёт по работе должен содержать:

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.

### UML диаграмма команд



### Исходный код

[https://github.com/isa-alex/Prog\\_lab6](https://github.com/isa-alex/Prog_lab6)

### Выводы

Во время разработки клиент-серверного приложения я познакомилась с протоколом UDP. Разработала клиент-серверную архитектуру на основе пятой лабораторной, поделив её на клиент и на сервер. Научилась сереализации классов. Углубила знания о логировании.