Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление программная инженерия
Образовательная программа системное и прикладное программное обеспечение

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

курса «Программирование»

Вариант № 33

Выполнил студент:

Бых Даниил Максимович

группа: P3109

Преподаватель:

Гаврилов А. В.,
Гиря М. Д.

Санкт-Петербург, 2026 г.

Содержание

Лабораторная работа № 5

1. Задание варианта № 33

Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса Route, описание которого приведено ниже.

Разработанная программа должна удовлетворять следующим требованиям:

- Класс, коллекцией экземпляров которого управляет программа, должен реализовывать сортировку по умолчанию.
- Все требования к полям класса (указанные в виде комментариев) должны быть выполнены.
- Для хранения необходимо использовать коллекцию типа *java.util.ArrayList*
- При запуске приложения коллекция должна автоматически заполняться значениями из файла.
- Имя файла должно передаваться программе с помощью: *переменная окружения.*
- Данные должны храниться в файле в формате *json.*
- Чтение данных из файла необходимо реализовать с помощью класса *java.util.Scanner.*
- Запись данных в файл необходимо реализовать с помощью класса *java.io.PrintWriter*
- Все классы в программе должны быть задокументированы в формате javadoc.
- Программа должна корректно работать с неправильными данными (ошибки пользовательского ввода, отсутсвие прав доступа к файлу и т.п.).

В интерактивном режиме программа должна поддерживать выполнение следующих команд:

- **help** : вывести справку по доступным командам
- **info** : вывести в стандартный поток вывода информацию о коллекции (тип, дата инициализации, количество элементов и т.д.)
- **show** : вывести в стандартный поток вывода все элементы коллекции в строковом представлении
- **add {element}** : добавить новый элемент в коллекцию
- **update id {element}** : обновить значение элемента коллекции, id которого равен заданному
- **remove_by_id id** : удалить элемент из коллекции по его id
- **clear** : очистить коллекцию
- **save** : сохранить коллекцию в файл
- **execute_script file_name** : считать и исполнить скрипт из указанного файла. В скрипте содержатся команды в таком же виде, в котором их вводит пользователь в интерактивном режиме.
- **exit** : завершить программу (без сохранения в файл)
- **remove_lower {element}** : удалить из коллекции все элементы, меньшие, чем заданный
- **sort** : отсортировать коллекцию в естественном порядке
- **history** : вывести последние 13 команд (без их аргументов)
- **filter_starts_with_name name** : вывести элементы, значение поля name которых начинается с заданной подстроки
- **print_unique_distance** : вывести уникальные значения поля distance всех элементов в коллекции
- **print_field_descending_distance** : вывести значения поля distance всех элементов в порядке убывания

Формат ввода команд:

- Все аргументы команды, являющиеся стандартными типами данных (примитивные типы, классы-оболочки, String, классы для хранения дат), должны вводиться в той же строке, что и имя команды.

- Все составные типы данных (объекты классов, хранящиеся в коллекции) должны вводиться по одному полю в строку.
- При вводе составных типов данных пользователю должно показываться приглашение к вводу, содержащее имя поля (например, "Введите дату рождения:")
- Если поле является enum'ом, то вводится имя одной из его констант (при этом список констант должен быть предварительно выведен).
- При некорректном пользовательском вводе (введена строка, не являющаяся именем константы в enum'е; введена строка вместо числа; введённое число не входит в указанные границы и т.п.) должно быть показано сообщение об ошибке и предложено повторить ввод поля.
- Для ввода значений null использовать пустую строку.
- Поля с комментарием "Значение этого поля должно генерироваться автоматически"не должны вводиться пользователем вручную при добавлении.

Описание хранимых в коллекции классов:

```
public class Route {
    private Integer id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, 
    private String name; //Поле не может быть null, Строка не может быть пустой
    private Coordinates coordinates; //Поле не может быть null
    private java.time.LocalDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматическ
    private Location from; //Поле не может быть null
    private Location to; //Поле не может быть null
    private int distance; //Значение поля должно быть больше 1
}
public class Coordinates {
    private Float x; //Поле не может быть null
    private long y; //Максимальное значение поля: 106
}
public class Location {
    private Integer x; //Поле не может быть null
    private double y;
    private String name; //Строка не может быть пустой, Поле может быть null
}
public class Location {
    private double x;
    private Double y; //Поле не может быть null
    private Integer z; //Поле не может быть null
    private String name; //Поле может быть null
}
```

Рис. 1.1: Описание коллекции

2. Выполнение задания.

Задание было выполнено в редакторе кода Visual Studio Code, собрано в `jar` файл `lab3.jar` и загружено в Git репозиторий на GitHub [1].

2. 1. Исходный код программы

https://github.com/DaniilRen/ITMO-labs/tree/main/Java/labs/Lab5

2. 2. UML-диаграмма классов

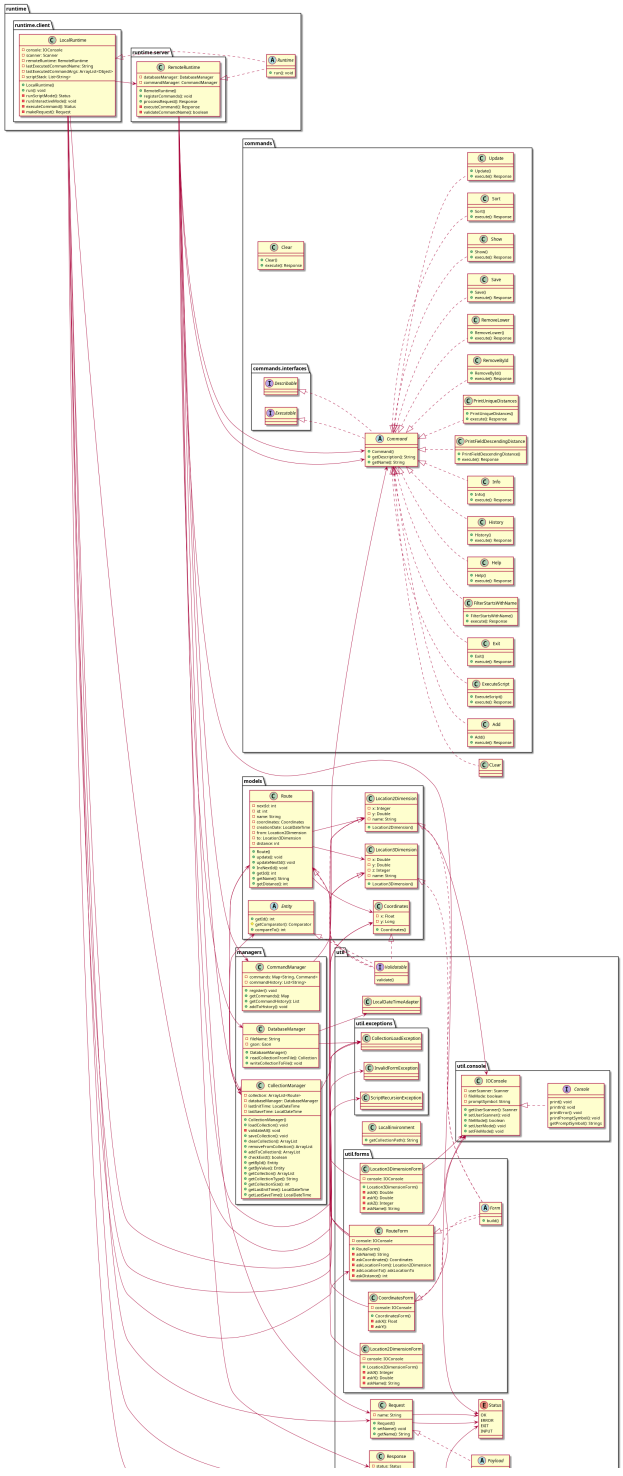

Рис. 1.2: UML диаграмма

Ссылка на диаграмму: https://github.com/DaniilRen/ITMO-labs/blob/main/Java/labs/Lab5/docs/assets/Lab5.png

2. 3. Упрощенная архитектурная схема

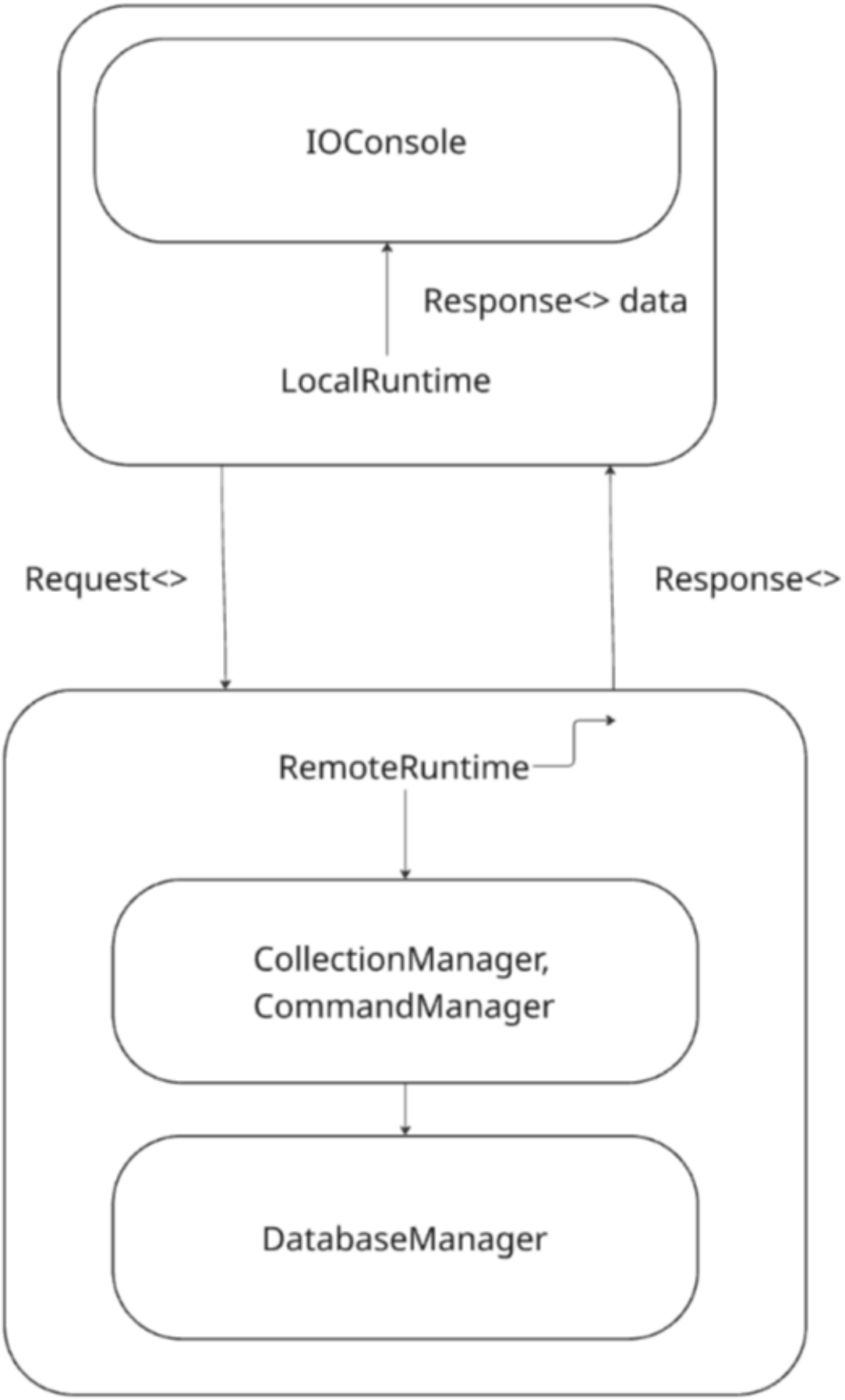

Рис. 1.3: архитектурная схема

3. Вывод

Во время выполнения лабораторной работы я изучил различные структуры данных в Java и файлы, методы работы с ними, углубил свои знания в ООП и SOLID, изучил параметризованные типы, wildcard-параметры и утилиту javadoc.

Литература

[1] Ссылка на личный репозиторий GitHub: https://github.com/DaniilRen/ITMO-labs/tree/main/Java/Lab3

[2] Ссылка на сайт с информацией о покемонах: https://pokemondb.net

[3] Ссылка на документацию по jar библиотеке с покемонами: https://se.ifmo.ru/~tony/doc/