

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Санкт-Петербургский национальный исследовательский университет  
информационных технологий, механики и оптики

Факультет программной инженерии и компьютерной техники

# Лабораторная работа №5

по программированию

## Вариант № 33592

Выполнила: Гафурова Фарангиз

Фуркатовна

Группа: Р3120

Принял:

Санкт-Петербург. 2024

## Оглавление

Текст задания.....	3
Диаграмма классов разработанной программы .....	3
Исходный код программы (класс main) .....	4
Результат работы программы.....	6
Вывод.....	7

## Текст задания

Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса `SpaceMarine`, 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.TreeSet`
- 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 `csv`
- 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, отсутствие прав доступа к файлу и т. п.).

**Формат ввода команд:**

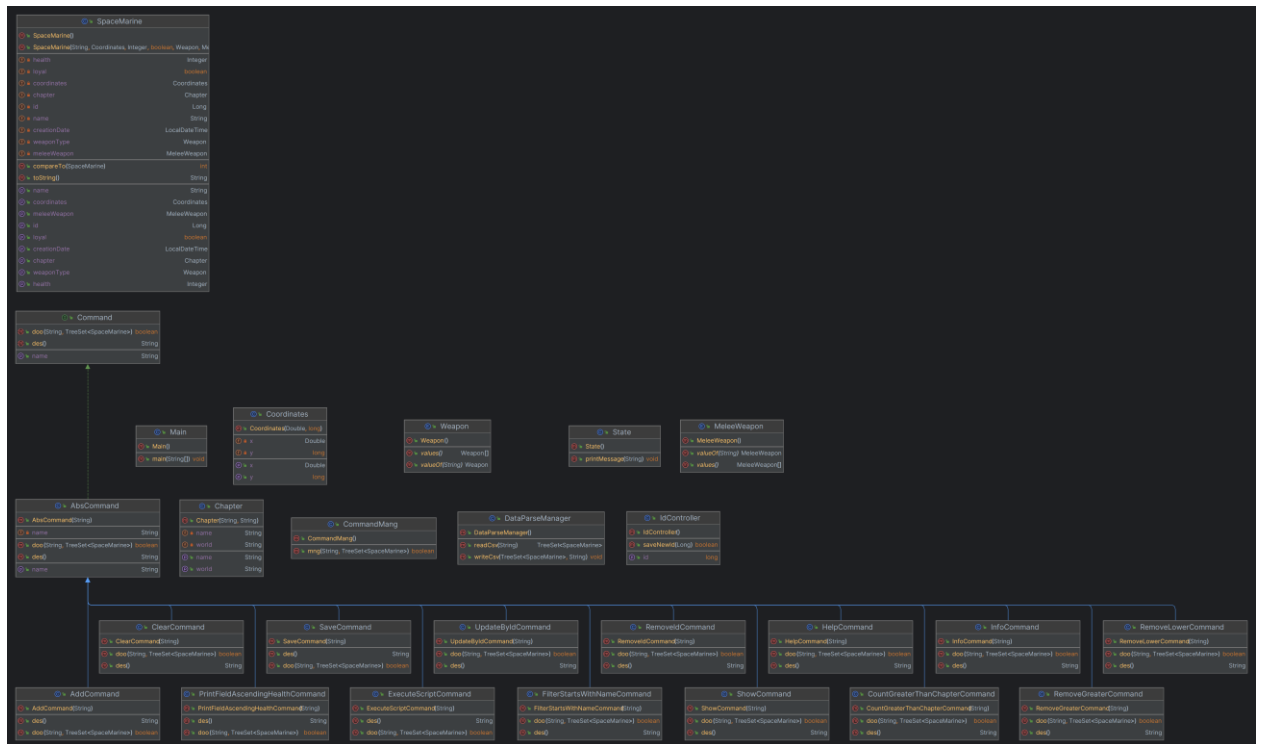
- Все аргументы команды, являющиеся стандартными типами данных (примитивные типы, классы-оболочки, `String`, классы для хранения дат), должны вводиться в той же строке, что и имя команды.
- Все составные типы данных (объекты классов, хранящиеся в коллекции) должны вводиться по одному полю в строку.
- 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 SpaceMarine {`
- `private Long id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически`
- `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 автоматически`
- `private Integer health; //Поле не может быть null, Значение поля должно быть больше 0`
- `private boolean loyal;`
- `private Weapon weaponType; //Поле может быть null`
- `private MeleeWeapon meleeWeapon; //Поле может быть null`
- `private Chapter chapter; //Поле не может быть null`
- `}`
- `public class Coordinates {`
- `private Double x; //Значение поля должно быть больше -267, Поле не может быть null`
- `private long y; //Максимальное значение поля: 803`
- `}`
- `public class Chapter {`
- `private String name; //Поле не может быть null, Строка не может быть пустой`
- `private String world; //Поле не может быть null`
- `}`
- `public enum Weapon {`
- `BOLTGUN,`
- `COMBI_PLASMA_GUN,`
- `FLAMER,`
- `INFERNO_PISTOL;`
- `}`
- `public enum MeleeWeapon {`
- `POWER_SWORD,`
- `MANREAPER,`
- `LIGHTING_CLAW,`
- `POWER_BLADE;`
- `}`

## Диаграмма классов разработанной программы



## Исходный код программы (класс main)

```
1 package org.example;
2
3 import classes.*;
4 import collection.CollectionManager;
5 import command.CommandManager;
6 import statics.ConsoleManager;
7
8 import java.io.IOException;
9 import java.util.TreeSet;
10
11 public class Main {
12     public static void main(String[] args) throws IOException {
13         //String filename = "Data/baza.csv";
14         String filename = System.getenv( "name: \"BD\""); //(BD=Data/baza.csv) environment variable if in IntelliJ*/
15         CollectionManager collectionManager = new CollectionManager();
16         CommandManager cmd = new CommandManager();
17         ConsoleManager console = new ConsoleManager();
18
19         TreeSet<SpaceMarine> mySet = new TreeSet<>();
20         mySet = collectionManager.getAllData(filename);
21
22         console.start(cmd, filename, mySet);
23     }
24 }
```

Весь код доступен тут: <https://github.com/karillisa/Laboratory-work-5>

## Результат работы программы

```
> Task :Main.main()
Коллекция успешно заполнено с файла!
>>> help
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
clear : очистить коллекцию
save : сохранить коллекцию в файл
remove_by_id id : удалить элемент из коллекции по его id
filter_starts_with_name name : вывести элементы, значение поля name которых начинается с заданной подстроки
print_field_ascending_health : вывести значения поля health всех элементов в порядке возрастания
add {element} : добавить новый элемент в коллекцию
update id {element} : обновить значение элемента коллекции, id которого равен заданному
count_greater_than_chapter chapter : вывести количество элементов, значение поля chapter которых больше заданного
remove_greater {element} : удалить из коллекции все элементы, превышающие заданный
remove_lower {element} : удалить из коллекции все элементы, меньшие, чем заданный
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
>>> info
id - identification number
name - название
coordinates - координаты
creationDate - дата создания
health
loyal
weaponType
meleeWeapon
chapter
Количество Объектов: 8
>>> |
```

## Вывод

Во время выполнения данной лабораторной работы я научилась использовать Javadoc, попробовала проектировать архитектуру проекта, работать с потоками, файлами, интерфейсами Comparable и Comparator, научилась сериализовать и десериализовать данные в различные форматы.