МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 5

по дисциплине

‘ПРОГРАММИРОВАНИЕ’

Вариант №3131006

Выполнил:
Студент группы P3131
Дворкин Борис
Александрович

Преподаватель:
Гаврилов Антон
Валерьевич

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2023

Задание:

Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса `City`, 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.io.InputStreamReader`
- Запись данных в файл необходимо реализовать с помощью класса `java.io.BufferedWriter`
- 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)
- `add_if_min {element}` : добавить новый элемент в коллекцию, если его значение меньше, чем у наименьшего элемента этой коллекции
- `remove_greater {element}` : удалить из коллекции все элементы, превышающие заданный
- `history` : вывести последние 14 команд (без их аргументов)
- `sum_of_meters_above_sea_level` : вывести сумму значений поля metersAboveSeaLevel для всех элементов коллекции
- `print_descending` : вывести элементы коллекции в порядке убывания
- `print_field_descending_meters_above_sea_level` : вывести значения поля metersAboveSeaLevel 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 City {
    private long id; //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.LocalDate creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически
    private Integer area; //Значение поля должно быть больше 0, Поле не может быть null
    private int population; //Значение поля должно быть больше 0
    private long metersAboveSeaLevel;
    private Climate climate; //Поле может быть null
    private Government government; //Поле не может быть null
    private StandardOfLiving standardOfLiving; //Поле не может быть null
    private Human governor; //Поле может быть null
}
public class Coordinates {
    private Integer x; //Максимальное значение поля: 499, Поле не может быть null
    private Double y; //Значение поля должно быть больше -274, Поле не может быть null
}
public class Human {
    private String name; //Поле не может быть null, Строка не может быть пустой
}
public enum Climate {
    TROPICAL_SAVANNA,
    OCEANIC,
    STEPPE;
}
public enum Government {
    ANARCHY,
    GERONTOCRACY,
    DEMARCHY,
    COMMUNISM,
    NOOCRACY;
}
public enum StandardOfLiving {
    ULTRA_HIGH,
    VERY_HIGH,
    HIGH,
    ULTRA_LOW,
    NIGHTMARE;
}
```

Исходный код доступен по ссылке или QR-коду:

https://github.com/worthant/Java_labs/tree/main/lab5

Вывод:

Во время выполнения данной лабораторной работы я научился использовать Javadoc, попробовал проектировать архитектуру проекта, работать с потоками, файлами, интерфейсами Comparable и Comparator, научился сереализовать и десереализовать данные в различные форматы.