МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

Лабораторная работа №5

по дисциплине «Программирование»

Выполнил:

Лабин Макар Андреевич

Проверил:

Обляшевский Севастьян Александрович

г. Санкт-Петербург, 2025

Текст задания

Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса `Ticket`, 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 `xml`
- Чтение данных из файла необходимо реализовать с помощью класса `java.io.BufferedInputStream`
- Запись данных в файл необходимо реализовать с помощью класса `java.io.OutputStreamWriter`
- 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_at index` : удалить элемент, находящийся в заданной позиции коллекции (index)
- `remove_last` : удалить последний элемент из коллекции
- `add_if_max {element}` : добавить новый элемент в коллекцию, если его значение превышает значение наибольшего элемента этой коллекции
- `min_by_price` : вывести любой объект из коллекции, значение поля price которого является минимальным
- `count_greater_than_event event` : вывести количество элементов, значение поля event которых больше заданного
- `print_field_descending_type` : вывести значения поля type 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 Ticket {

    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 int price; //Значение поля должно быть больше 0

    private TicketType type; //Поле не может быть null

    private Event event; //Поле не может быть null

}

public class Coordinates {

    private Float x; //Поле не может быть null

    private Float y; //Поле не может быть null

}

public class Event {

    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 String description; //Длина строки не должна быть больше 1190, Поле
может быть null

    private EventType eventType; //Поле не может быть null

}

public enum TicketType {

    VIP,

    USUAL,

    BUDGETARY,

    CHEAP;

}

public enum EventType {

    CONCERT,

    BASEBALL,

    OPERA;

}
```

Диаграмма классов объектной модели

С диаграммой классов объектной модели можно ознакомиться в репозитории GitHub по ссылке: https://github.com/MrDvD/itmo_labs/blob/dev-prog/programming/5/uml/diagram.png.

Исходный код программы

С исходным кодом программы можно ознакомиться в репозитории GitHub по ссылке: https://github.com/MrDvD/itmo_labs/tree/master/programming/5.

Выводы по работе

w.i.p