Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Вариант №367081

Лабораторная работа №5

По дисциплине

Программирование

Выполнил студент группы P3115:

Кохан Всеволод

Преподаватель:

Сорокин Роман Борисович

Письмак Алексей Евгеньевич

Санкт-Петербург 2024 г.

1. **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.TreeSet * При запуске приложения коллекция должна автоматически заполняться значениями из файла. * Имя файла должно передаваться программе с помощью: **аргумент командной строки**. * 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.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.).   **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) * head : вывести первый элемент коллекции * add\_if\_max {element} : добавить новый элемент в коллекцию, если его значение превышает значение наибольшего элемента этой коллекции * remove\_lower : удалить из коллекции все элементы, меньшие, чем заданный * history: вывести последние 11 команд(без их аргументов) * filter\_contains\_name name : вывести элементы, значение поля name которых содержит данную подстроку * print\_ascending :вывести элементы коллекции в порядке возрастания * print\_descending :вывести элементы коллекции 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 Integer id = 0;//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 LocalDateTime creationDate;//Поле не может быть null, Значение этого поля должно генерироваться автоматически  private float price;//Значение поля должно быть больше 0  private Long discount;//Поле может быть null, Значение поля должно быть больше 0, Максимальное значение поля: 100  private TicketType type;//Поле не может быть null  private Person person;//Поле может быть null }  public class Coordinates {  private float x;//Значение поля должно быть больше -948  private Long y;//Поле не может быть null }  public class Person {    private float height;//Значение поля должно быть больше 0  private Color eyeColor;//Поле может быть null  private Color hairColor;//Поле не может быть null  private Country nationality;//Поле может быть null  private Location location;//Поле не может быть null }  Location {  private final Long x;//Поле не может быть null  private Double y;//Поле не может быть null  private String name;//Длина строки не должна быть больше 370, Поле может быть nul  }  public enum TicketType {  VIP,  USUAL,  BUDGETARY,  CHEAP; }  public enum Color {  GREEN,  WHITE,  BROWN,  ORANGE; }  public enum Country {  UNITED\_KINGDOM,  USA,  SPAIN,  THAILAND;} |

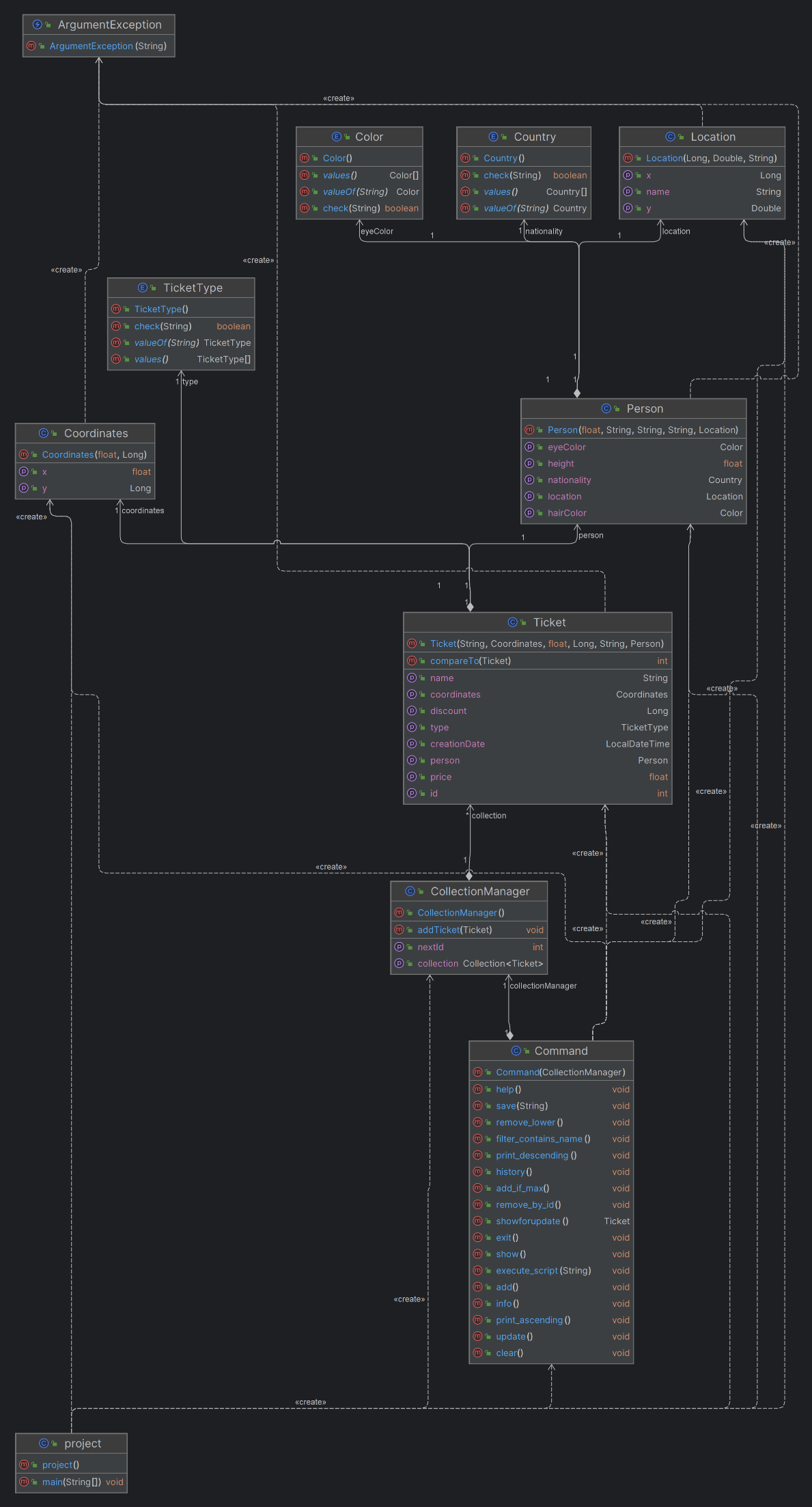
1. **Исходный код программы.**

Репозиторий:

<https://github.com/VsevKokhan/ITMOProg/tree/master/lab5>

1. **Диаграмма классов реализованной объектной модели.**

<https://github.com/VsevKokhan/ITMOProg/blob/master/lab5/lab5.png>



1. **Вывод**

Во время выполнения данной лабораторной работы я научился работать с различными структурами данных в Java и файлами, а также углубил свои знания о ООП в Java, изучил параметризованные типы, wildcard-параметры и утилиту javadoc.