Санкт-Петербургский Национальный Исследовательский Университет ИТМО
Факультет Программной Инженерии и Компьютерной Техники

Вариант №329
Лабораторная работа №5
По дисциплине
Программирование

Выполнил студент группы P3132:
Ежелев Георгий

Преподаватель:
Данилов Павел Юрьевич

Санкт-Петербург 2026 г.

1. Текст задания

Введите вариант: 329

Внимание! У разных вариантов разный текст задания!

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.LinkedHashMap`
- 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.BufferedReader`
- Запись данных в файл необходимо реализовать с помощью класса `java.io.FileWriter`
- 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
- `insert null {element}` : добавить новый элемент с заданным ключом
- `update id {element}` : обновить значение элемента коллекции, id которого равен заданному
- `remove_key null` : удалить элемент из коллекции по его ключу
- `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)
- `replace_if_lowe null {element}` : заменить значение по ключу, если новое значение меньше старого
- `remove_greater_key null` : удалить из коллекции все элементы, ключ которых превышает заданный
- `remove_lower_key null` : удалить из коллекции все элементы, ключ которых меньше, чем заданный
- `remove_any_by_distance distance` : удалить из коллекции один элемент, значение поля distance которого эквивалентно заданному
- `max_by_distance` : вывести любой объект из коллекции, значение поля distance которого является максимальным
- `count_greater_than_distance distance` : вывести количество элементов, значение поля distance которых больше заданного

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, 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.util.Date creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически
    private Location from; //Поле может быть null
    private Location to; //Поле не может быть null
    private long distance; //Значение поля должно быть больше 1
}
public class Coordinates {
    private int x;
    private double y;
}
public class Location {
    private long x;
    private Float y; //Поле не может быть null
    private Double z; //Поле не может быть null
}
public class Location {
    private Double x; //Поле не может быть null
    private long y;
    private String name; //Поле не может быть null
}
```

Отчёт по работе должен содержать:

1. Текст задания.
2. Диаграмма классов разработанной программы.
3. Исходный код программы.
4. Выводы по работе.

Вопросы к защите лабораторной работы:

1. Коллекции. Сортировка элементов коллекции. Интерфейсы `java.util.Comparable` и `java.util.Comparator`.
2. Категории коллекций - списки, множества, очереди, отображения (словари). Интерфейсы `java.util.List`, `java.util.Set`, `java.util.Queue`, `java.util.Map` и их реализации.
3. Параметризованные типы. Создание параметризуемых классов. Wildcard-параметры.
4. Классы-оболочки. Назначение, область применения, преимущества и недостатки. Автоупаковка и автораспаковка.
5. Потоки ввода-вывода в Java. Байтовые и символьные потоки. "Цепочки" потоков (Stream Chains).
6. Работа с файлами в Java. Класс `java.io.File`.
7. Пакет `java.nio` - назначение, основные классы и интерфейсы.
8. Утилита `javadoc`. Особенности автоматического документирования кода в Java.

2. Диаграмма классов и исходный код

Диаграмма:
https://github.com/haaroner/ITMO_Clown/blob/main/Java/lab%205/Class%20Diagram1-1.pdf
Репозиторий с программой, отчетом и документацией:
https://github.com/haaroner/ITMO_Clown/tree/main/Java/lab%205

3. Вывод

Во время выполнения этой лабораторной работы я закрепил работу с ООП в Java. Освоил принципы работы с коллекциями, на примере java.util.LinkedHashMap. Изучил и применил шаблоны проектирования builder и command. Подробнее изучил и освоил работу с вводом-выводом, а также сериализацией и десериализации объектов по стандарту CSV с использованием библиотеки OpenCSV. Также познакомился с инструментом Javadoc который показался мне крайне удобным.