УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Программирование

Лабораторная работа №5

Вариант 1085

Преподаватель: Горбунов Михаил Витальевич

Выполнил: Хюинь Тан Куонг

R3138

Санкт-Петербург

2022

Задание:

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.HashSet`
- 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.BufferedInputStream`
- 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_max {element}` : добавить новый элемент в коллекцию, если его значение превышает значение наибольшего элемента этой коллекции
- `remove_lower {element}` : удалить из коллекции все элементы, меньшие, чем заданный
- `history` : вывести последние 9 команд (без их аргументов)
- `remove_all_by_person person` : удалить из коллекции все элементы, значение поля person которого эквивалентно заданному
- `min_by_price` : вывести любой объект из коллекции, значение поля price которого является минимальным
- `filter_greater_than_price price` : вывести элементы, значение поля price 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 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.LocalDate creationDate; //Поле не может быть null,
Значение этого поля должно генерироваться автоматически
    private Long price; //Поле не может быть null, Значение поля должно
быть больше 0
    private TicketType type; //Поле может быть null
    private Person person; //Поле не может быть null
}

public class Coordinates {
    private long x;
    private long y;
}

public class Person {
    private java.util.Date birthday; //Поле не может быть null
    private long height; //Значение поля должно быть больше 0
    private String passportID; //Строка не может быть пустой, Значение
этого поля должно быть уникальным, Длина строки должна быть не меньше
10, Поле не может быть null
}

public enum TicketType {
    VIP,
    USUAL,
    BUDGETARY,
    CHEAP;
}
```

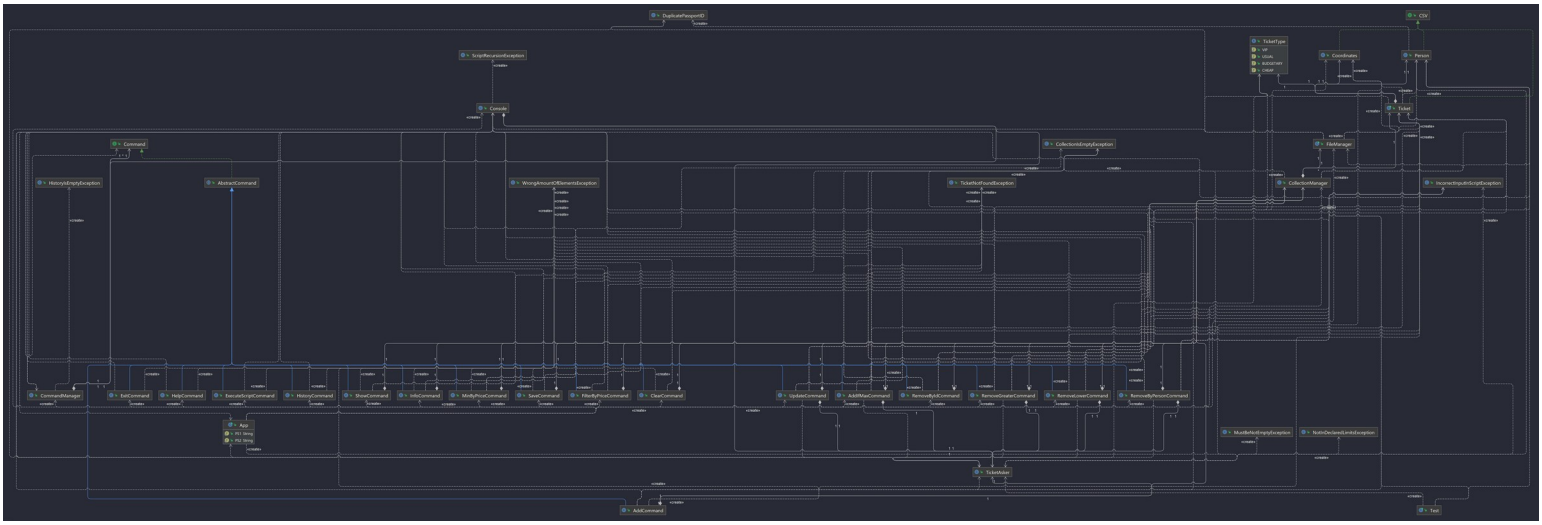
Отчёт по работе должен содержать:

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. Интерфейс `java.util.Map` и его 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.

Диаграмма классов:



Исходный код программы:

https://github.com/huynhtancuong/ITMO_LABS/tree/master/Lab5

Результат работы программы:

$help

add {element} add new item to collection

help display help on available commands

clear clear the collection

help display help on available commands

exit exit program (without saving to file)

add_if_max {element} add a new element if its value is less than that of the smallest

show show all items of collection

save save collection to file

info show information about collection

update <ID> {element} update the value of the collection element whose id is equal to the given

remove_by_id <ID> remove item from collection by ID

execute_script <file_name> execute script from specified file

history show history of used commands

remove_greater {element} remove all items from the collection that are greater than the given one

remove_lower {element} remove all items from the collection that are less than the given one

remove_all_by_person {element} remove all ticket of this person

min_by_price display any object from the collection whose price field value is the minimum

filter_by_price <Price> display elements whose price field value is greater than the specified one

$show

Collection is empty!

$add

Enter name:

>huynh

Enter X:

>2

Enter Y:

>2

Enter price:

>150

List of ticket types - VIP, USUAL, BUDGETARY, CHEAP

Please enter ticket types:

>vip

Enter Passport ID:

>this is a passport id

Enter birthday (dd-MM-yyyy):

>08-06-2002

Enter height:

>172

Created new item successfully

$show

Ticket №1 (created 2022-03-14)

Name: huynh

Coordinate:

-X = 2

-Y = 2

Price: 150

Type: VIP

Person:

-PassportID = this is a passport id

-Height = 172

-Birthday = Sat Jun 08 00:00:00 MSD 2002

$save

$show

Ticket №1 (created 2022-03-14)

Name: huynh

Coordinate:

-X = 2

-Y = 2

Price: 150

Type: VIP

Person:

-PassportID = this is a passport id

-Height = 172

-Birthday = Sat Jun 08 00:00:00 MSD 2002

$add

Enter name:

>chau

Enter X:

>1

Enter Y:

>1

Enter price:

>150

List of ticket types - VIP, USUAL, BUDGETARY, CHEAP

Please enter ticket types:

>vip

Enter Passport ID:

>this is a passport id 2

Enter birthday (dd-MM-yyyy):

>13-09-2002

Enter height:

>156

Created new item successfully

$show

Ticket №2 (created 2022-03-14)

Name: chau

Coordinate:

-X = 1

-Y = 1

Price: 150

Type: VIP

Person:

-PassportID = this is a passport id 2

-Height = 156

-Birthday = Fri Sep 13 00:00:00 MSD 2002

Ticket №1 (created 2022-03-14)

Name: huynh

Coordinate:

-X = 2

-Y = 2

Price: 150

Type: VIP

Person:

-PassportID = this is a passport id

-Height = 172

-Birthday = Sat Jun 08 00:00:00 MSD 2002

$

Вывод

This lab is so long and complicated, but through this lab I learned a lot about how to build a CLI program. This experience will be useful in the future. Because the program is designed to satisfied SOLID principle so they can be reuse easily in the future project.

I learned about:

1. Collections. Sorting the elements of the collection. Interfaces `java.util.Comparable`and `java.util.Comparator`.
2. Collection categories - lists, sets. Interface `java.util.Map`and its implementation.
3. Parameterized types. Creation of parameterizable classes. Wildcard parameters.
4. Wrapper classes. Purpose, scope, advantages and disadvantages. Autoboxing and autoboxing.
5. I / O streams in Java. Byte and character streams. Stream Chains.
6. Working with files in Java. Class `java.io.File`.
7. Package `java.nio`- purpose, main classes and interfaces.
8. Utility `javadoc`... Features of automatic code documentation in Java.