Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники



Вариант №367081

Лабораторная работа №5

По дисциплине

Программирование

Выполнил студент группы P3115:

Барсуков Максим

Преподаватель:

Сорокин Роман Борисович

Письмак Алексей Евгеньевич

Санкт-Петербург 2023 г.

1. **Текст задания**

|  |
| --- |
| Реализовать консольное приложение, которое реализует управление коллекцией объектов в интерактивном режиме. В коллекции необходимо хранить объекты класса Product, 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.PriorityQueue * 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 json * Чтение данных из файла необходимо реализовать с помощью класса java.io.FileReader * 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 * add\_if\_min {element} : добавить новый элемент в коллекцию, если его значение меньше, чем у наименьшего элемента этой коллекции * sum\_of\_price : вывести сумму значений поля price для всех элементов коллекции * filter\_by\_price price : вывести элементы, значение поля price которых равно заданному * filter\_contains\_part\_number partNumber : вывести элементы, значение поля partNumber которых содержит заданную подстроку   **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. * Все аргументы команды, являющиеся стандартными типами данных (примитивные типы, классы-оболочки, String, классы для хранения дат), должны вводиться в той же строке, что **Описание хранимых в коллекции классов:**   public class Product {  private int 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 Long price; //Поле не может быть null, Значение поля должно быть больше 0  private String partNumber; //Строка не может быть пустой, Поле может быть null  private UnitOfMeasure unitOfMeasure; //Поле может быть null  private Organization manufacturer; //Поле может быть null  }  public class Coordinates {  private Integer x; //Поле не может быть null  private Long y; //Поле не может быть null  }  public class Organization {  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 long employeesCount; //Значение поля должно быть больше 0  private OrganizationType type; //Поле не может быть null  private Address postalAddress; //Поле не может быть null  }  public class Address {  private String street; //Строка не может быть пустой, Поле не может быть null  private String zipCode; //Длина строки должна быть не меньше 6, Поле может быть null  }  public enum UnitOfMeasure {  KILOGRAMS,  SQUARE\_METERS,  LITERS,  MILLILITERS;  }  public enum OrganizationType {  COMMERCIAL,  GOVERNMENT,  TRUST,  PRIVATE\_LIMITED\_COMPANY;  } |

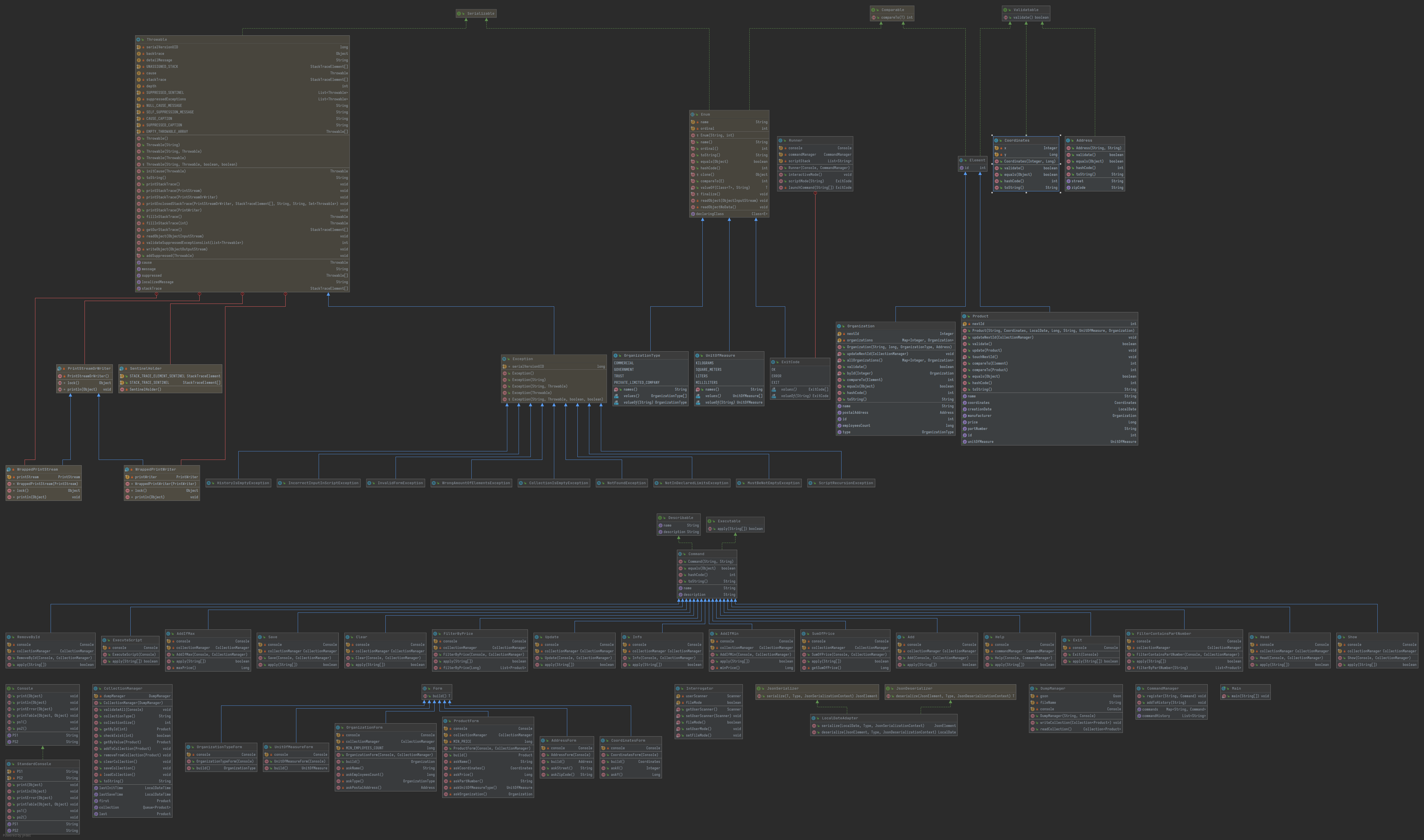
1. **Исходный код программы.**

Репозиторий: https://github.com/maxbarsukov/itmo/tree/master/%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab5

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

<https://github.com/maxbarsukov/itmo/blob/master/%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab5/docs/diagrams/uml-with-deps.png>

<https://github.com/maxbarsukov/itmo/blob/master/%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab5/docs/diagrams/uml-no-deps.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.

