

**Федеральное государственное автономное образовательное
учреждение высшего образования «Национальный
исследовательский ядерный университет «МИФИ»**

ОТЧЕТ
по лабораторной работе №3 по
дисциплине: «Теория и технология
программирования»

Выполнила: студентка группы Б23-902 Деробас Л. И.

Проверил: Смирнов Д. С.

Москва 2025 г

Оглавление

Постановка задачи	2
Ссылка на GitHub	2
Диаграммы	3
Интерфейс	6
Важные части кода.....	7
Файлы данных	9

Постановка задачи

1. Систематизировать информацию из файла Bestiarum.txt и представить ее в файлах формата JSON, XML, YAML
2. Дополнить полученные файлы еще 5 уникальными чудовищами
3. Импортировать информацию с этих файлов
4. Для реализации выбора исполняемой библиотеки использовать паттерн «Цепочка обязанностей»
5. Создать класс Java для хранения информации о типе чудовища. Обязательный атрибут – источник получения информации
6. Создать коллекции-хранилища, под каждый импортированный файл
7. Выбор файлов для импорта организовать с использованием класса JFileChooser, обязательно указать директорию вызова программы как директорию по умолчанию
8. Вывести на экран информацию о загруженных типах чудовищ с использованием представления JTree
9. Создать окно представления информации о конкретном типе чудовища, при клике на лист дерева
10. Сделать хотя бы один параметр в форме представления информации о конкретном типе чудовища редактируемым, с изменением значений атрибутов класса типа чудовища.
11. Реализовать обратный экспорт информации в файлы формата JSON, XML, YAML, учитывая внесенные пользователем изменения

Ссылка на GitHub

<https://github.com/Shirouky/lab3-Java>

Диаграммы

Рис 1. DFD-диаграмма 0 уровня

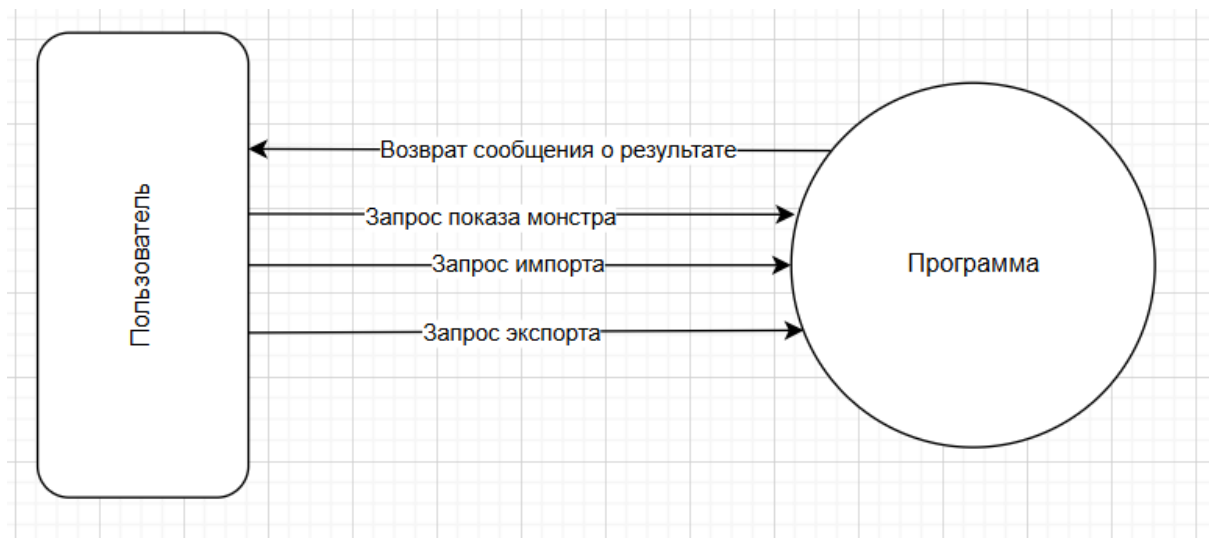


Рис 2. DFD-диаграмма 1 уровня

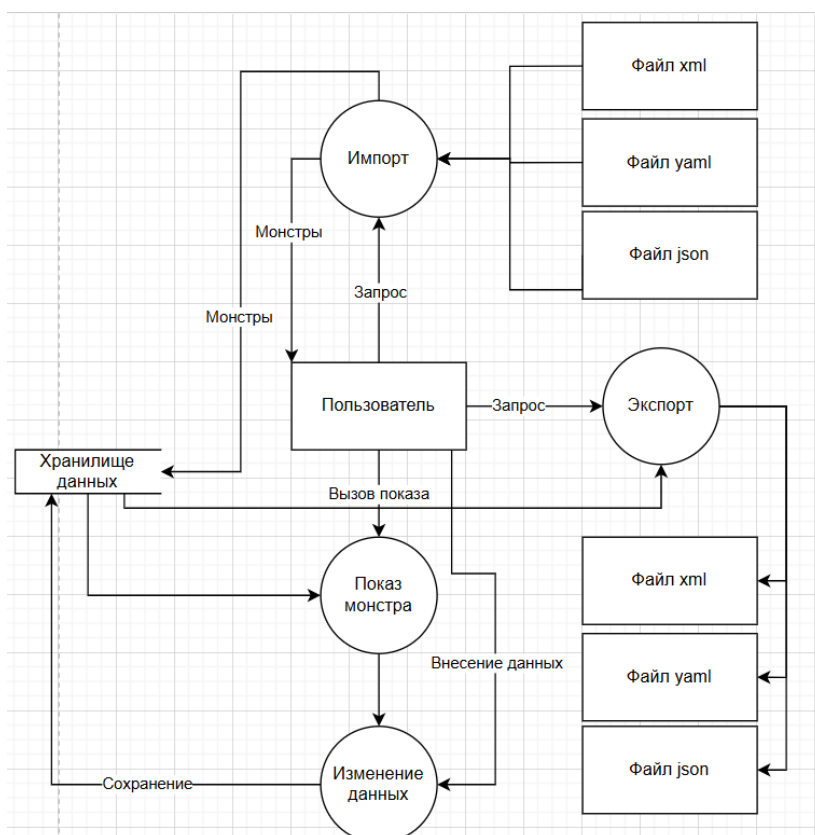
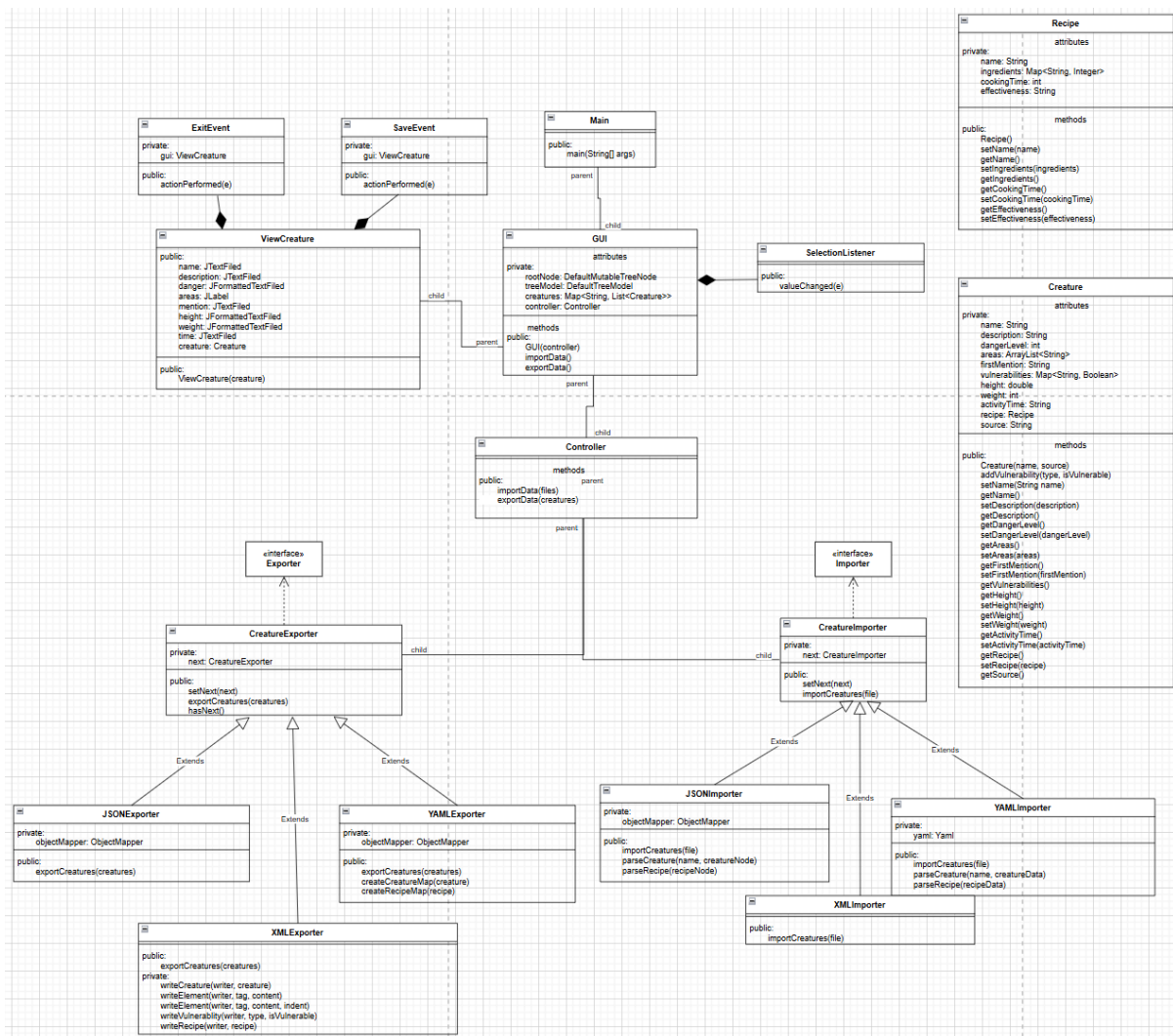


Рис 3. UML-диаграмма на концептуальном уровне



```
classDiagram
    class Recipe {
        attributes:
            name: String
            ingredients: Map<String, Integer>
            cookingTime: int
            effectiveness: String
        methods:
            Recipe()
            setName(String name)
            getName()
            setIngredients(Map<String, Integer> ingredients)
            getIngredients()
            getCookingTime()
            setCookingTime(int cookingTime)
            getEffectiveness()
            setEffectiveness(String effectiveness)
    }

    class Creature {
        attributes:
            name: String
            description: String
            dangerLevel: int
            areas: Array<String>
            firstMention: String
            vulnerabilities: Map<String, Boolean>
            height: double
            weight: int
            activityTime: String
            recipe: Recipe
            source: String
        methods:
            Creature(String name, String source)
            addVulnerability(String type, boolean isVulnerable)
            setName(String name)
            getDescription()
            setDescription(String description)
            getDangerLevel()
            setDangerLevel(int dangerLevel)
            getAreas()
            setAreas(Array<String> areas)
            getFirstMention()
            setFirstMention(String firstMention)
            getVulnerabilities()
            getWeight()
            setHeight(double height)
            getWeight()
            setWeight(int weight)
            getActivityTime()
            setActivityTime(String activityTime)
            getRecipe()
            setRecipe(Recipe recipe)
            getSource()
    }

    class Main {
        main(String[] args)
    }

    class GUI {
        attributes:
            rootNode: DefaultMutableTreeNode
            treeModel: DefaultTreeModel
            creatures: Map<String, List<Creature>>
            controller: Controller
        methods:
            GUI(Controller controller)
            importData()
            exportData()
    }

    class Controller {
        methods:
            importData(File) files
            exportData(Map<String, List<Creature>>) creatures
    }

    class View {
        attributes:
            name: JTextField
            description: JTextFiled
            danger: JFormattedTextFiled
            areas: JLabel
            mention: JTextField
            height: JFormattedTextFiled
            weight: JFormattedTextFiled
            time: JTextFiled
            creature: Creature
        methods:
            View(View Creature creature)
    }

    class SelectionListener {
        valueChanged(TreeSelectionEvent e)
    }

    class CreatureExporter {
        attributes:
            next: CreatureExporter
        methods:
            setNext(CreatureExporter next)
            exportCreatures(Map<String, List<Creature>>) creatures
            hasNext()
    }

    class CreatureImporter {
        attributes:
            next: CreatureImporter
        methods:
            setNext(CreatureImporter next)
            importCreatures(File file)
    }

    class JSONExporter {
        attributes:
            objectMapper: ObjectMapper
        methods:
            exportCreatures(Map<String, List<Creature>>) creatures
    }

    class YAMLExporter {
        attributes:
            objectMapper: ObjectMapper
        methods:
            exportCreatures(Map<String, List<Creature>>) creatures
            createCreatureMap(Creature creature)
            createRecipeMap(Recipe recipe)
    }

    class XMLExporter {
        methods:
            exportCreatures(Map<String, List<Creature>>) creatures
            writeCreature(XMLStreamWriter writer, Creature creature)
            writeElement(XMLStreamWriter writer, String tag, String content)
            writeVulnerability(XMLStreamWriter writer, String tag, String content, String indent)
            writeRecipe(XMLStreamWriter writer, Recipe recipe)
    }

    class JSONImporter {
        attributes:
            objectMapper: ObjectMapper
        methods:
            importCreatures(File file)
            parseCreature(String name, JsonNode creatureNode)
            parseRecipe(JsonNode recipeNode)
    }

    class YAMLImporter {
        attributes:
            yaml: Yaml
        methods:
            importCreatures(File file)
            parseCreature(String name, Map<String, Object> creatureData)
            parseRecipe(Map<String, Object> recipeData)
    }

    class XMLImporter {
        methods:
            importCreatures(File file)
    }

    Recipe "1" -- "*" Creature
    Main --> GUI
    GUI --> Controller
    GUI --> SelectionListener
    Controller --> GUI
    Controller --> CreatureExporter
    Controller --> CreatureImporter
    CreatureExporter --|> JSONExporter
    CreatureExporter --|> YAMLExporter
    CreatureExporter --|> XMLExporter
    CreatureImporter --|> JSONImporter
    CreatureImporter --|> YAMLImporter
    CreatureImporter --|> XMLImporter
    CreatureExporter --> CreatureImporter
    CreatureExporter --> CreatureImporter
```

Интерфейс

Рис 5. Главная страница

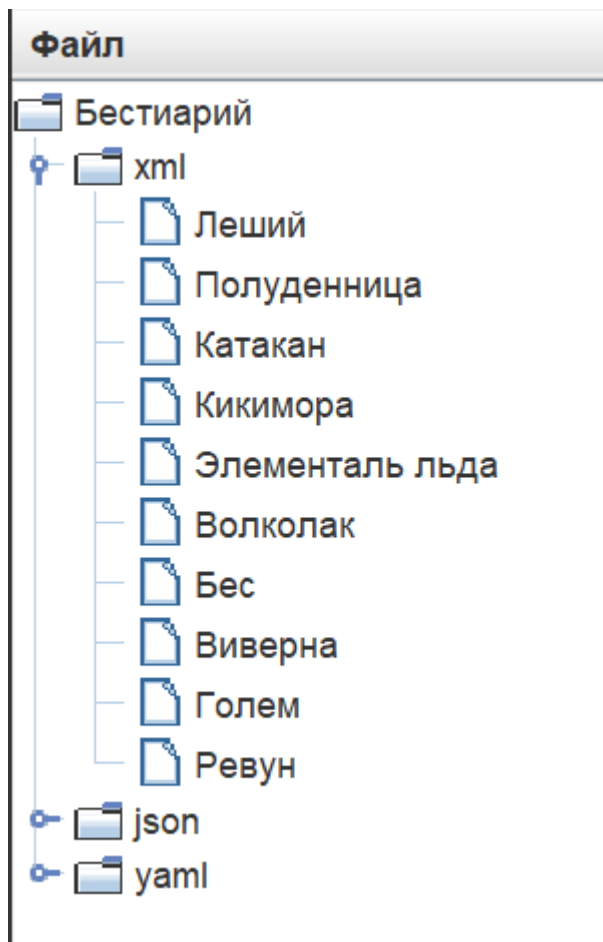


Рис 6. Страница существа

Name:	<input type="text" value="Кикимора"/>
Description:	<input type="text" value="Болотная нежить из гниющих останков"/>
Danger level:	<input type="text" value="6"/>
Area:	<input type="text" value="в топях Вендена"/>
First mention:	<input type="text" value="1221-09-03"/>
Height:	<input type="text" value="1"/>
Weight:	<input type="text" value="15"/>
Activity time:	<input type="text" value="Ночь"/>

Vulnerabilities: К удушью не уязвим К магии земли уязвим

Recipe: Масло Cooking time: 45 Effectiveness: Средняя Роса папоротника 3 Грибы-поганки 5 Глаз василиска 1

Важные части кода

Рис 7. Структура кода

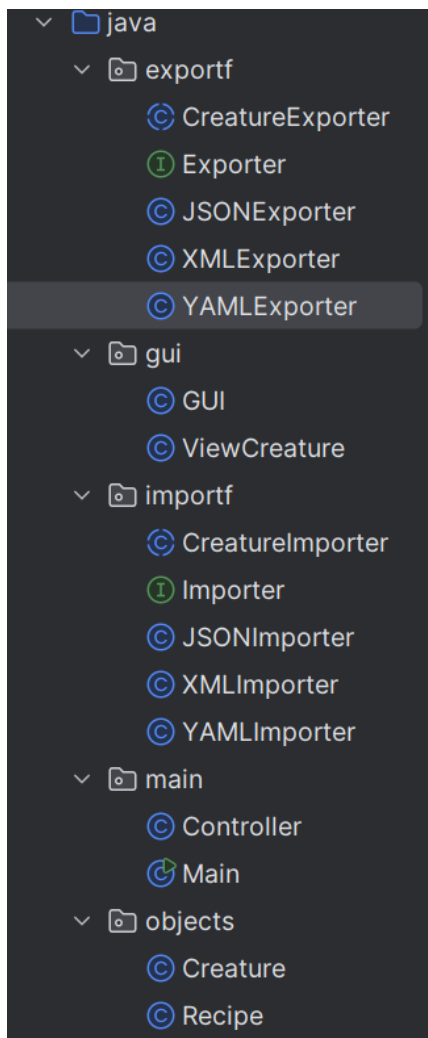


Рис 8. Класс CreatureExporter

```
package exportf;

import objects.Creature;

import java.util.List;
import java.util.Map;

public abstract class CreatureExporter implements Exporter {
    10 usages 3 inheritors 1 Shirouky
    protected CreatureExporter next;

    public void setNext(CreatureExporter next) {
        1 Shirouky
        this.next = next;
    }

    public abstract boolean exportCreatures(Map<String, List<Creature>> creatures);
    4 usages 3 implementations 1 Shirouky

    public boolean hasNext() {
        3 usages 1 Shirouky
        return next != null;
    }
}
```

Рис 9. Класс CreatureImporter

```
package importf;

import objects.Creature;

import java.io.File;
import java.util.List;

public abstract class CreatureImporter implements Importer{ 10 usages 3 inheritors 1 Shirouky
    protected CreatureImporter next;

    public void setNext(CreatureImporter next) { 1 Shirouky
        this.next = next;
    }

    public abstract List<Creature> importCreatures(File file); 4 usages 3 implementations 1 Shirouky
}
```

Рис 10. Класс Controller

```
public class Controller { 4 usages 1 Shirouky
    public Map<String, List<Creature>> importData(File[] files) { 1 usage 1 Shirouky
        CreatureImporter jsonImporter = new JSONImporter();
        CreatureImporter xmlImporter = new XMLImporter();
        CreatureImporter yamlImporter = new YAMLImporter();

        jsonImporter.setNext(xmlImporter);
        xmlImporter.setNext(yamlImporter);
        Map<String, List<Creature>> creatures = new HashMap<>();
        for (File file : files) {
            List<Creature> importedCreatures = jsonImporter.importCreatures(file);
            if (importedCreatures != null) creatures.put(importedCreatures.getFirst().getSource(), importedCreatures);
        }
        return creatures;
    }

    public boolean exportData(Map<String, List<Creature>> creatures) { 1 usage 1 Shirouky
        CreatureExporter jsonExporter = new JSONExporter();
        CreatureExporter xmlExporter = new XMLExporter();
        CreatureExporter yamlExporter = new YAMLExporter();

        jsonExporter.setNext(xmlExporter);
        xmlExporter.setNext(yamlExporter);

        return jsonExporter.exportCreatures(creatures);
    }
}
```


Файлы данных

Рис 11. Файл JSON

```
{
  "Леший": {
    "description": "Гибрид человека и лесного духа, охраняющий древние рощи",
    "dangerLevel": 7,
    "areas": [
      "в чащах Брокилона",
      "в чащах Эдингеля"
    ],
    "firstMention": "1234-06-21",
    "vulnerability": {
      "К магии": false,
      "К оглушению": false,
      "К ядам": false
    },
    "height": 3,
    "weight": 300,
    "activityTime": "Ночь",
    "recipe": {
      "name": "Яд",
      "ingredients": {
        "Корень мандрагоры": 3,
        "Пепел дракониды": 1,
        "Эссенция полыни": 1
      },
      "cookingTime": 60,
      "effectiveness": "Умеренная"
    }
  }
}
```

Рис 12. Файл XML

```
<creature name="Леший">
  <description>Гибрид человека и лесного духа, охраняющий древние рощи</description>
  <dangerLevel>7</dangerLevel>
  <areas>
    <area>в чащах Брокилона</area>
    <area>в чащах Эдингель</area>
  </areas>
  <firstMention>1234-06-21</firstMention>
  <vulnerabilities>
    <vulnerability name="К магии" quantity="false"/>
    <vulnerability name="К оглушению" quantity="false"/>
    <vulnerability name="К ядам" quantity="false"/>
  </vulnerabilities>
  <height>3</height>
  <weight>300</weight>
  <activityTime>Ночь</activityTime>
  <recipe>
    <name>Яд</name>
    <ingredients>
      <ingredient name="Корень мандрагоры" quantity="3"/>
      <ingredient name="Пепел дракониды" quantity="1"/>
      <ingredient name="Эссенция полыни" quantity="1"/>
    </ingredients>
    <cookingTime>60</cookingTime>
    <effectiveness>Умеренная</effectiveness>
  </recipe>
</creature>
```

Рис 13. Файл YAML

```
creatures:
  Леший:
    description: "Гибрид человека и лесного духа, охраняющий древние рощи"
    dangerLevel: 7
    areas:
      - "в чащах Брокилона"
      - "в чащах Эдингеля"
    firstMention: "1234-06-21"
    vulnerability:
      К магии: false
      К оглушению: false
      К ядам: false
    height: 3
    weight: 300
    activityTime: "Ночь"
    recipe:
      name: "Яд"
      ingredients:
        Корень мандрагоры: 3
        Пепел драконида: 1
        Эссенция полыни: 1
      cookingTime: 60
      effectiveness: "Умеренная"
```