МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний технічний університет «Дніпровська політехніка»

ЗВІТ

Лабораторна робота №4

з дисципліни «Поглиблене програмування в середовищі Java»

Виконала: ст. гр. 122-21-1

Пивонос Ольга Олександрівна

Перевірив: доц. Мінєєв О.С.

Дніпро

2025

Тема: JUnit. Json.

Завдання: Додати до лабораторної роботи 3 можливість запису університету у формат json, запис цього формату у файл, зчитування цього формату файлу, та створення об’єкту з текстового формату json. В проєкті повинен бути зроблений JUnit тест, який буде виглядати наступним чином: створити об’єкт університет (oldUniversity), в якому в кожному підрозділі маються два підрозділи нижчого рівня. Наприклад на факультеті дві кафедри, на кожній кафедрі дві групи, на кожній групі два студенти. Цей об’єкт повинен бути записаний в файл у форматі json. Потім з цього файлу зчитаний та відновлений як newUniversity. В тесті повинні бути порівняні newUniversity та oldUniversity за допомогою методу equals. Якщо все зроблено правильно, то університети повинні бути еквівалентні, а метод equals повинен повернути True. Для запису та зчитування університету у форматі json повинен бути зроблений клас JsonManager. Для безпосереднього перетворення університету у формат json та його відновлення з цього формату, можливо використання сторонніх бібліотек, наприклад Gson, Jackson чи будь-яких інших.

Для початку розробки лабораторної роботи номер 4 повністю скопіювати програмний код лабораторної роботи номер 3. Не змішувати ці роботи ні в якому разі.

Хід роботи

1. Для кожного класу групи model додаємо перевизначення методу equals() для порівняння об’єктів одного класу між собою. Класи групи controller залишаються без змін.

2. Створюємо клас JsonManager, який буде відповідати за серіалізацію та десеріалізацію – перетворення об’єкта University у JSON і запис його у файл та зчитування цього об’єкта з файлу JSON. Для цього використовуємо бібліотеку Gson, а також визначаємо адаптер для коректної обробки успадкування класу Human.

3. У класі UniversityTest створюємо університет та його підрозділи, записуємо це у файл, далі зчитуємо університет з файлу і виконуємо порівняння двох об’єктів – створеного у програмі та отриманого з файлу JSON. Код програми:

Human.java

```
package edu.ntudp.pzks.lab4.model;

import java.util.Objects;

public abstract class Human {
    private String firstName;
    private String lastName;
    private String patronymic;
    private Sex sex;

    public Human(String firstName, String lastName, String patronymic, Sex
sex) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.patronymic = patronymic;
        this.sex = sex;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public String getPatronymic() {
        return patronymic;
    }

    public Sex getSex() {
        return sex;
    }

    @Override
    public String toString() {
        return lastName + " " + firstName + " " + patronymic;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;

        if (!(o instanceof Human)) return false;

        Human human = (Human) o;

        return Objects.equals(firstName, human.firstName) &&
                Objects.equals(lastName, human.lastName) &&
                Objects.equals(patronymic, human.patronymic) &&
                sex == human.sex;
    }
```

```
    @Override
    public int hashCode() {
        return Objects.hash(firstName, lastName, patronymic, sex);
    }
}
```

Student.java

```
package edu.ntudp.pzks.lab4.model;

public class Student extends Human {
    private int studentId;

    public Student(String firstName, String lastName, String patronymic, Sex
sex, int studentId) {
        super(firstName, lastName, patronymic, sex);
        this.studentId = studentId;
    }

    public int getStudentId() {
        return studentId;
    }

    @Override
    public String toString() {
        return super.toString();
    }
}
```

Sex.enum

```
package edu.ntudp.pzks.lab4.model;

public enum Sex {
    MALE, FEMALE
}
```

Group.java

```
package edu.ntudp.pzks.lab4.model;

import java.util.ArrayList;
import java.util.List;

public class Group {
    private String name;
    private Human head;
    private List<Student> students;

    public Group(String name, Human head) {
        this.name = name;
        this.head = head;
        this.students = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public Human getHead() {
```

```
        return head;
    }

    public List<Student> getStudents() {
        return students;
    }

    public void addStudent(Student student) {
        students.add(student);
    }

    @Override
    public String toString() {
        return "Група " + name + ", кількість студентів: " + students.size()
+ ", староста - " + head ;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Group group = (Group) obj;
        return name.equals(group.name) &&
                head.equals(group.head) &&
                students.equals(group.students);
    }
}
```

Department.java

```
package edu.ntudp.pzks.lab4.model;

import java.util.ArrayList;
import java.util.List;

public class Department {
    private String name;
    private Human head;
    private List<Group> groups;

    public Department(String name, Human head) {
        this.name = name;
        this.head = head;
        this.groups = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public Human getHead() {
        return head;
    }

    public List<Group> getGroups() {
        return groups;
    }

    public void addGroup(Group group) {
        groups.add(group);
    }

    @Override
```

```
    public String toString() {
        return "Кафедра " + name + ", завідувач - " + head;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Department department = (Department) obj;
        return name.equals(department.name) &&
                head.equals(department.head) &&
                groups.equals(department.groups);
    }
}
```

Faculty.java

```
package edu.ntudp.pzks.lab4.model;

import java.util.ArrayList;
import java.util.List;

public class Faculty {
    private String name;
    private Human head;
    private List<Department> departments;

    public Faculty(String name, Human head) {
        this.name = name;
        this.head = head;
        this.departments = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public Human getHead() {
        return head;
    }

    public List<Department> getDepartments() {
        return departments;
    }

    public void addDepartment(Department department) {
        departments.add(department);
    }

    @Override
    public String toString() {
        return "Факультет " + name + ", декан - " + head + ", кафедри: " +
departments;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Faculty faculty = (Faculty) obj;
        return name.equals(faculty.name) &&
                head.equals(faculty.head) &&
                departments.equals(faculty.departments);
```

```
    }
}
```

University.java

```
package edu.ntudp.pzks.lab4.model;

import java.util.ArrayList;
import java.util.List;

public class University {
    private String name;
    private Human head;
    private List<Faculty> faculties;

    public University(String name, Human head) {
        this.name = name;
        this.head = head;
        this.faculties = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public Human getHead() {
        return head;
    }

    public List<Faculty> getFaculties() {
        return faculties;
    }

    public void addFaculty(Faculty faculty) {
        faculties.add(faculty);
    }

    @Override
    public String toString() {
        return "Університет " + name + ", факультети: " + faculties;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        University university = (University) obj;
        return name.equals(university.name) &&
                head.equals(university.head) &&
                faculties.equals(university.faculties);
    }
}
```

JsonManager.java

```
package edu.ntudp.pzks.lab4.json;

import com.google.gson.*;
import edu.ntudp.pzks.lab4.model.*;
import java.io.*;
import java.lang.reflect.Type;
```

```
public class JsonManager {
    private static final Gson gson = new GsonBuilder()
            .setPrettyPrinting()
            .registerTypeAdapter(Human.class, new HumanAdapter())
            .create();

    public static void writeToJsonFile(University university, String
filename) throws IOException {
        try (FileWriter writer = new FileWriter(filename)) {
            gson.toJson(university, writer);
        }
    }

    public static University readFromJsonFile(String filename) throws
IOException {
        try (FileReader reader = new FileReader(filename)) {
            return gson.fromJson(reader, University.class);
        }
    }

    private static class HumanAdapter implements JsonSerializer<Human>,
JsonDeserializer<Human> {
        @Override
        public JsonElement serialize(Human src, Type typeOfSrc,
JsonSerializationContext context) {
            JsonObject jsonObject = context.serialize(src).getAsJsonObject();
            jsonObject.addProperty("type", src.getClass().getSimpleName());
            return jsonObject;
        }

        @Override
        public Human deserialize(JsonElement json, Type typeOfT,
JsonDeserializationContext context) throws JsonParseException {
            JsonObject jsonObject = json.getAsJsonObject();
            String type = jsonObject.get("type").getAsString();
            Class<? extends Human> clazz = "Student".equals(type) ?
Student.class : Human.class;
            return context.deserialize(json, clazz);
        }
    }
}
```

UniversityTest.java

```
package edu.ntudp.pzks.test;

import edu.ntudp.pzks.lab4.json.JsonManager;
import edu.ntudp.pzks.lab4.model.*;
import org.junit.jupiter.api.Test;

import java.io.IOException;

public class UniversityTest {
    @Test
    void testUniversityJson() throws IOException {
        University oldUniversity = createUniversity();
        JsonManager.writeToJsonFile(oldUniversity, "University.json");
        University newUniversity =
JsonManager.readFromJsonFile("University.json");
        System.out.println(oldUniversity.equals(newUniversity));
    }
```

```
    private University createUniversity() {
        Student st1 = new Student("Олег", "Кравченко", "Іванович", Sex.MALE,
1);
        Student st2 = new Student("Владислав", "Захарченко", "Сергійович",
Sex.MALE, 2);
        Student st3 = new Student("Валерія", "Мороз", "Романівна",
Sex.FEMALE, 3);
        Student st4 = new Student("Анастасія", "Нестеренко", "Вадимівна",
Sex.FEMALE, 4);
        Student st5 = new Student("Володимир", "Ткач", "Юрійович", Sex.MALE,
5);

        Group group1 = new Group("122-1", st1);
        group1.addStudent(st1);
        group1.addStudent(st2);

        Group group2 = new Group("121-1", st3);
        group2.addStudent(st3);

        Group group3 = new Group("124-1", st4);
        group3.addStudent(st4);
        group3.addStudent(st5);

        Student pzksHead = new Student("Ольга", "Зімниця", "Василівна",
Sex.FEMALE, 6);
        Department pzks = new Department("ПЗКС", pzksHead);
        pzks.addGroup(group1);
        pzks.addGroup(group2);

        Student sauHead = new Student("Сергій", "Кириленко", "Анатолійович",
Sex.MALE, 7);
        Department sau = new Department("САУ", sauHead);
        sau.addGroup(group3);

        Student facultyHead = new Student("Софія", "Олійник", "Денисівна",
Sex.FEMALE, 8);
        Faculty fit = new Faculty("ФІТ", facultyHead);
        fit.addDepartment(sau);
        fit.addDepartment(pzks);

        Student rector = new Student("Бойко", "Олександр", "Анатолійович",
Sex.MALE, 9);
        University ntudp = new University("НТУ ДП", rector);
        ntudp.addFaculty(fit);

        return ntudp;
    }
}
```

Результати роботи програми показано на рисунках 1 та 2.

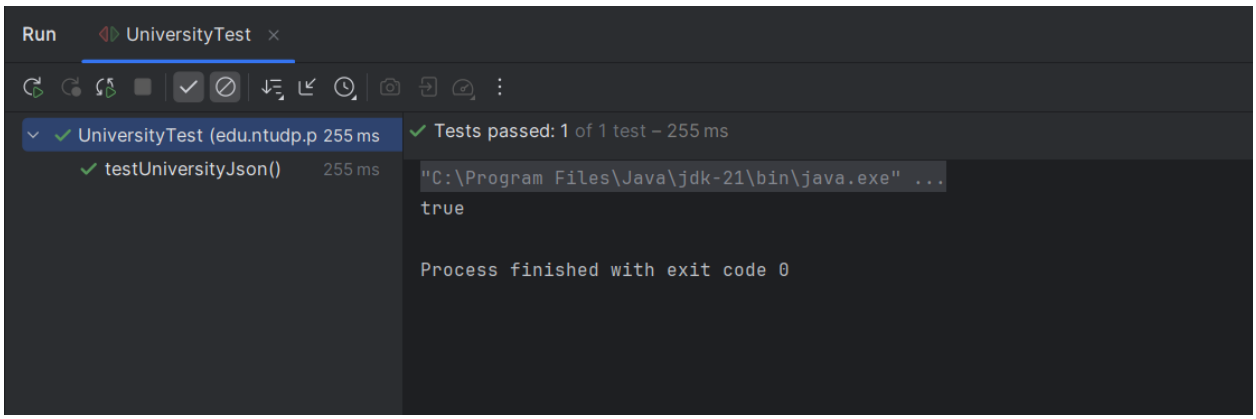

Рисунок 1 – Тест UniversityTest пройшов

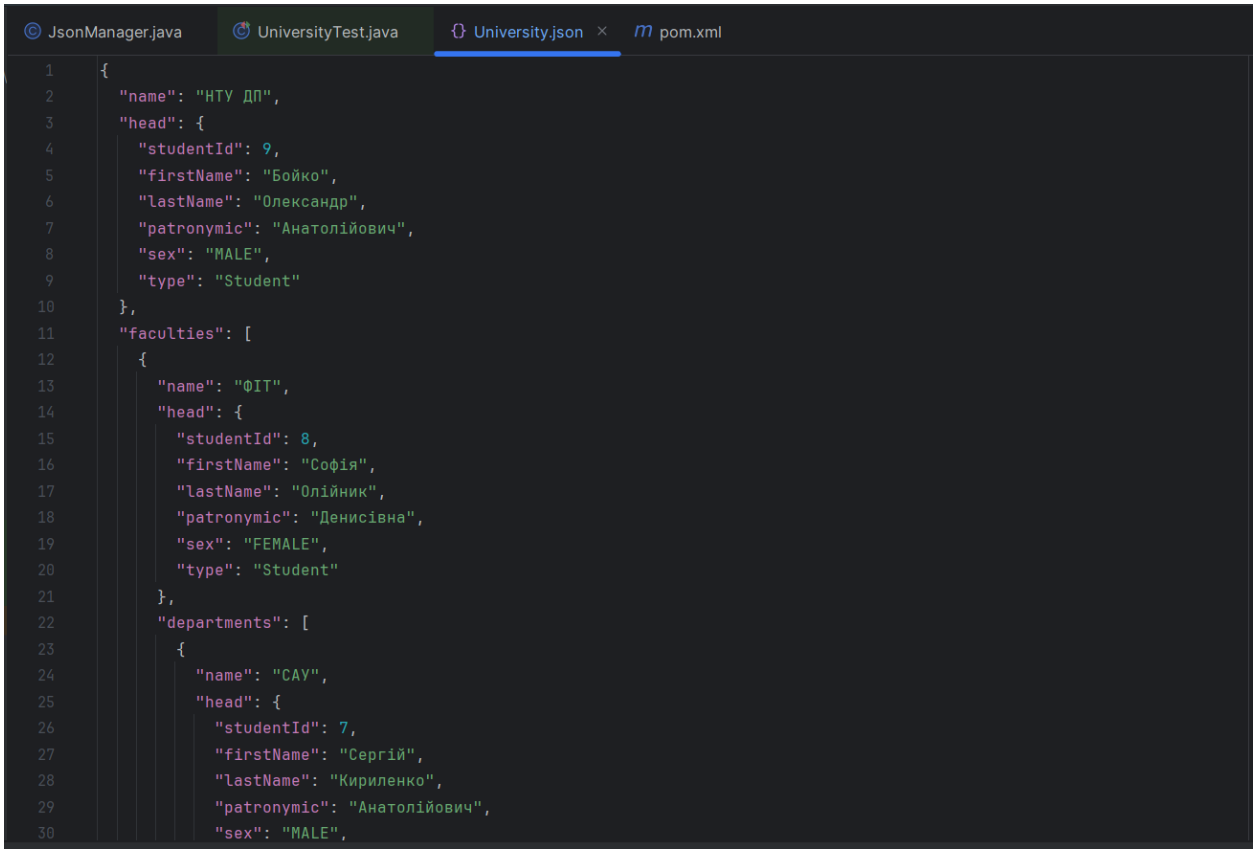

Рисунок 2 – Запис університету у файл JSON

Висновки: На даній лабораторній роботі було створено програму, що виконує серіалізацію та десеріалізацію з використанням формату JSON. За допомогою бібліотеки Gson програма здійснює перетворення об’єкта у текстовий формат та відновлює цей об’єкт з файлу, а після цього порівнює створений програмою об’єкт та об’єкт, отриманий з файлу.