Міністерство освіти і науки України
НТУ «Дніпровська політехніка»
Кафедра системного аналізу та управління

Звіт з лабораторної роботи № 4

З дисципліни «Поглиблене програмування в середовищі Java»

Виконав:
ст. гр. 122-21-2
Помазан Микита
Максимович

Дніпро
2025

Лабораторна робота: JUnit. Json

Мета роботи: Розширити функціонал лабораторної роботи №3, додавши можливість збереження університету у форматі JSON, запису у файл, зчитування з файлу та відновлення об'єкта. Реалізувати JUnit тестування для перевірки коректності серіалізації та десеріалізації.

Хід роботи:

university.json

```
1  {"name":"Дніпровська політехніка","head":{"firstName":"Олена","lastName":"Мороз","middleName":"Юріївна","sex":"FEMALE"},
```

Рис. 1 - збереження університету у форматі JSON

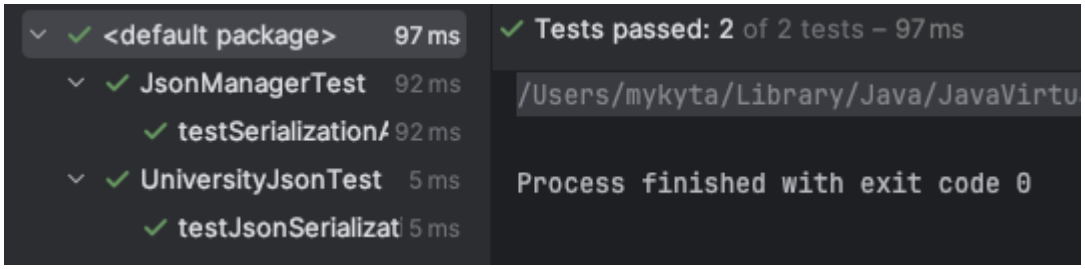

Рис. 2 - JUnit тестування для перевірки коректності

Код програми:

https://github.com/mykytapomazan/java-basic/tree/LR_4

або:

Залежності:

```
<dependencies>
   <dependency>
       <groupId>com.google.code.gson</groupId>
       <artifactId>gson</artifactId>
       <version>2.10.1</version>
   </dependency>

   <dependency>
       <groupId>org.junit.jupiter</groupId>
       <artifactId>junit-jupiter-api</artifactId>
```

```
        <version>5.9.2</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-engine</artifactId>
        <version>5.9.2</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

JsonManager:

```
package controller;


import com.google.gson.Gson;
import model.University;


import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;


public class JsonManager {


    public static void writeUniversityToJson(University university, String filePath) {
        Gson gson = new Gson();
        try (FileWriter writer = new FileWriter(filePath)) {
            gson.toJson(university, writer);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }


    public static University readUniversityFromJson(String filePath) {
        Gson gson = new Gson();
```

```
        try (FileReader reader = new FileReader(filePath)) {
            return gson.fromJson(reader, University.class);
        } catch (IOException e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

JsonManagerTest:

```
import controller.*;
import model.*;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

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

public class JsonManagerTest {

    private static final String FILE_PATH = "university.json";

    @BeforeEach
    public void setUp() {
        File file = new File(FILE_PATH);
        if (file.exists()) {
            file.delete();
        }
    }

    @Test
    public void testSerializationAndDeserialization() {
```

```
        List<Student> students = new ArrayList<>();

        students.add(StudentCreator.createStudent("Микита", "Помазан", "Максимович", Sex.MALE, "1"));

        students.add(StudentCreator.createStudent("Марія", "Петренко", "Анатоліївна", Sex.FEMALE, "2"));



        Human departmentHead = new Human("Михаїл", "Джонсон", "Якович", Sex.MALE);



        List<Group> groups = new ArrayList<>();

        groups.add(GroupCreator.createGroup("Група А", departmentHead, students));



        List<Department> departments = new ArrayList<>();

        departments.add(new Department("Комп'ютерні науки", departmentHead, groups));



        Human facultyHead = new Human("Олена", "Шевченко", "Іванівна", Sex.FEMALE);

        List<Faculty> faculties = new ArrayList<>();

        faculties.add(FacultyCreator.createFaculty("IT", facultyHead, departments));



        Human universityHead = new Human("Оксана", "Коваленко", "Миколаївна", Sex.FEMALE);

        University oldUniversity;

        oldUniversity = UniversityCreator.createUniversity("Дніпровська політехніка", universityHead,
faculties);



        JsonManager.writeUniversityToJson(oldUniversity, FILE_PATH);



        University newUniversity = JsonManager.readUniversityFromJson(FILE_PATH);



        assertEquals(oldUniversity, newUniversity, "The universities should be equal after serialization
and deserialization.");

    }

}
```

UniversityJsonTest:

```
import model.*;

import org.junit.jupiter.api.Test;

import controller.JsonManager;

import java.util.List;
```

```
import static org.junit.jupiter.api.Assertions.*;


public class UniversityJsonTest {
   private static final String FILE_PATH = "university.json";


   @Test
   public void testJsonSerialization() {
       University oldUniversity = createSampleUniversity();


       JsonManager.writeUniversityToJson(oldUniversity, FILE_PATH);


       University newUniversity = JsonManager.readUniversityFromJson(FILE_PATH);


       assertEquals(oldUniversity, newUniversity, "Об'єкти університетів мають бути еквівалентні");
   }


   private University createSampleUniversity() {
       Student student1 = new Student("Іван", "Іванов", "Іванович", Sex.MALE, "S1");
       Student student2 = new Student("Марія", "Петренко", "Анатоліївна", Sex.FEMALE, "S2");


       Human groupHead = new Human("Олег", "Коваленко", "Миколайович", Sex.MALE);
       List<Student> students = List.of(student1, student2);
       Group group1 = new Group("Група А", groupHead, students);
       Group group2 = new Group("Група Б", groupHead, students);


       Human departmentHead = new Human("Анна", "Шевченко", "Володимирівна", Sex.FEMALE);
       Department department1 = new Department("Кафедра 1", departmentHead, List.of(group1, group2));
       Department department2 = new Department("Кафедра 2", departmentHead, List.of(group1, group2));


       Human facultyHead = new Human("Олександр", "Гончар", "Сергійович", Sex.MALE);
       Faculty faculty1 = new Faculty("Факультет IT", facultyHead, List.of(department1, department2));
       Faculty faculty2 = new Faculty("Факультет Економіки", facultyHead, List.of(department1,
department2));
```

```
        Human universityHead = new Human("Олена", "Мороз", "Юріївна", Sex.FEMALE);

        return new University("Дніпровська політехніка", universityHead, List.of(faculty1, faculty2));

    }

}
```

Оновленні файли моделей:

```
package model;


import java.util.List;

import java.util.Objects;


public class Department {

    private String name;

    private Human head;

    private List<Group> groups;


    public Department(String name, Human head, List<Group> groups) {

        this.name = name;

        this.head = head;

        this.groups = groups;

    }


    public String getName() {

        return name;

    }


    public Human getHead() {

        return head;

    }


    public List<Group> getGroups() {

        return groups;

    }
```

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

    @Override
    public int hashCode() {
        return Objects.hash(name, head, groups);
    }

    @Override
    public String toString() {
        return name + " headed by " + head.toString();
    }
}
```

```
package model;

import java.util.List;
import java.util.Objects;

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

    public Faculty(String name, Human head, List<Department> departments) {
        this.name = name;
        this.head = head;
        this.departments = departments;
    }
```

```
    public String getName() {
        return name;
    }

    public Human getHead() {
        return head;
    }

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

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Faculty that = (Faculty) o;
        return Objects.equals(name, that.name) &&
                Objects.equals(head, that.head) &&
                Objects.equals(departments, that.departments);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name, head, departments);
    }

    @Override
    public String toString() {
        return name + " headed by " + head.toString();
    }
}
package model;
```

```java
import java.util.List;

import java.util.Objects;



public class Group {

   private String name;

   private Human head;

   private List<Student> students;



   public Group(String name, Human head, List<Student> students) {

       this.name = name;

       this.head = head;

       this.students = students;

   }



   public String getName() {

       return name;

   }



   public Human getHead() {

       return head;

   }



   public List<Student> getStudents() {

       return students;

   }



   @Override

   public boolean equals(Object o) {

       if (this == o) return true;

       if (o == null || getClass() != o.getClass()) return false;

       Group group = (Group) o;

       return Objects.equals(name, group.name) &&

               Objects.equals(head, group.head) &&

               Objects.equals(students, group.students);

   }
```

```
    @Override
    public int hashCode() {
        return Objects.hash(name, head, students);
    }

    @Override
    public String toString() {
        return name + " headed by " + head.toString();
    }
}
```

```
package model;

import java.util.Objects;

public class Human {
    private String firstName;
    private String lastName;
    private String middleName;
    private Sex sex;

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

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }
```

```
    public String getMiddleName() {
        return middleName;
    }

    public Sex getSex() {
        return sex;
    }

    @Override
    public String toString() {
        return lastName + " " + firstName + " " + middleName + ", " + sex;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Human human = (Human) o;
        return Objects.equals(firstName, human.firstName) &&
                Objects.equals(lastName, human.lastName) &&
                Objects.equals(middleName, human.middleName) &&
                sex == human.sex;
    }

    @Override
    public int hashCode() {
        return Objects.hash(firstName, lastName, middleName, sex);
    }
}
package model;

import java.util.Objects;

public class Student extends Human {
```

```
    private String studentId;

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

    public String getStudentId() {
        return studentId;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        if (!super.equals(o)) return false;
        Student student = (Student) o;
        return Objects.equals(studentId, student.studentId);
    }

    @Override
    public int hashCode() {
        return Objects.hash(super.hashCode(), studentId);
    }

    @Override
    public String toString() {
        return super.toString() + ", номер студента: " + studentId;
    }
}
```

```
package model;

import java.util.List;
import java.util.Objects;
```

```java
public class University {

    private String name;

    private Human head;

    private List<Faculty> faculties;



    public University(String name, Human head, List<Faculty> faculties) {

        this.name = name;

        this.head = head;

        this.faculties = faculties;

    }



    public String getName() {

        return name;

    }



    public Human getHead() {

        return head;

    }



    public List<Faculty> getFaculties() {

        return faculties;

    }



    @Override

    public boolean equals(Object o) {

        if (this == o) return true;

        if (o == null || getClass() != o.getClass()) return false;

        University that = (University) o;

        return Objects.equals(name, that.name) &&

                Objects.equals(head, that.head) &&

                Objects.equals(faculties, that.faculties);

    }



    @Override

    public int hashCode() {
```

```
        return Objects.hash(name, head, faculties);

    }

}
```

Висновок: Розширив функціонал лабораторної роботи №3, додавши можливість збереження університету у форматі JSON, запису у файл, зчитування з файлу та відновлення об'єкта. Реалізувати JUnit тестування для перевірки коректності серіалізації та десеріалізації.