Міністерство науки і освіти України

Національний технічний університет

«Дніпровська Політехника»



**Звіт з лабораторної роботи №3**

**З дисципліни «Поглиблене програмування в середовищі Java»**

Виконав студент:

групи 121-21-1

Земляний Артем Сергійович

Прийняв:

Мінєєв О.С.

**м. Дніпро**

**2025 р.**

**Лабораторна робота №3**

**ООП.**

**Завдання:** Створити програму що буде створювати та обробляти комплексний об'єкт під назвою університет(university). Програма повинна складатися з трьох частин: модель вид та контролер згідно з парадигмою mvc (Model View Controller).  Кожній з цих груп повинна відповідати package з  відповідною назвою.  В моделі повинні знаходитись усі класи що відповідають за структурні підрозділи університету. Серед них:   університет, факультет, кафедра, група, студент, людина (Human). Усі вони повинні містити  назву  типу string та голову типу Human. Студент також повинен бути породжений від  Human.  Human повинен мати поля ім'я, прізвище, побатькові та стать.  Усі поля повинні бути строковими окрім поля стать. Стать повинна використовувати спеціальний  enum  типу Sex(стать).

В цій лабораторній роботі  група View Нам не потрібна.

 Що стосується групи контроллер (controller)  то вона повинна містити менеджери що дозволяють нам створити відповідні підрозділи наприклад  StudentCreator, FacultyCreator, GroupCreator  та інші, кожен з яких повинен використовувати можливості нижчого за рівнем   створювача. Програма повинна також містити клас Run, в якому буде знаходитись точка входу та методи, що повинні дати можливість створити університет.   Процес створення університету повинен бути зроблений в методі createTypycalUniversity.

  В програмі активно рекомендується використовувати абстрактні класи та інтерфейси

**Виконання лабораторної роботи:**

Наш основний код Run:

import controller.\*;

import model.\*;

import java.util.Arrays;

public class Run {

public static void main(String[] args) {

University typicalUniversity = *createTypicalUniversity*();

System.*out*.println("Створено університет: " + typicalUniversity);

System.*out*.println("\nСтруктура університету:");

// Структура

for (Faculty faculty : typicalUniversity.getFaculties()) {

System.*out*.println("- Факультет: " + faculty.getName() + " (Декан: " + faculty.getHead().getFullName() + ")");

for (Department department : faculty.getDepartments()) {

System.*out*.println(" - Кафедра: " + department.getName() + " (Зав. кафедри: " + department.getHead().getFullName() + ")");

for (Group group : department.getGroups()) {

System.*out*.println(" - Група: " + group.getName() + " (Староста: " + (group.getHead() != null ? group.getHead().getFullName() : "Немає") + ")");

for (Student student : group.getStudents()) {

System.*out*.println(" - Студент: " + student.getFullName());

}

}

}

}

}

public static University createTypicalUniversity() {

// Створюємо екземпляри Creator-ів

HumanCreator humanCreator = new HumanCreator();

StudentCreator studentCreator = new StudentCreator();

GroupCreator groupCreator = new GroupCreator();

DepartmentCreator departmentCreator = new DepartmentCreator();

FacultyCreator facultyCreator = new FacultyCreator();

UniversityCreator universityCreator = new UniversityCreator();

// 1. Створюємо людей

Human rector = humanCreator.createHuman("Віктор", "Андрущенко", "Петрович", Sex.*MALE*);

Human deanCS = humanCreator.createHuman("Ольга", "Черненко", "Іванівна", Sex.*FEMALE*);

Human deanMath = humanCreator.createHuman("Сергій", "Петренко", "Васильович", Sex.*MALE*);

Human headCSE = humanCreator.createHuman("Максим", "Зайцев", "Олегович", Sex.*MALE*);

Human headIS = humanCreator.createHuman("Ірина", "Білоус", "Михайлівна", Sex.*FEMALE*);

Human headAlgebra = humanCreator.createHuman("Андрій", "Волков", "Дмитрович", Sex.*MALE*);

Student student1 = studentCreator.createStudent("Іван", "Мельник", "Андрійович", Sex.*MALE*);

Student student2 = studentCreator.createStudent("Марія", "Ковальчук", "Сергіївна", Sex.*FEMALE*);

Student student3 = studentCreator.createStudent("Петро", "Бондаренко", "Вікторович", Sex.*MALE*);

Student student4 = studentCreator.createStudent("Оксана", "Шевченко", "Олегівна", Sex.*FEMALE*);

Student student5 = studentCreator.createStudent("Василь", "Поліщук", "Ігорович", Sex.*MALE*);

Student student6 = studentCreator.createStudent("Дарина", "Ткаченко", "Юріївна", Sex.*FEMALE*);

// 2. Університет

University university = universityCreator.createUniversity("Національний Технічний Університет", rector);

// 3. Факультети

Faculty csFaculty = facultyCreator.createFaculty("Факультет Комп'ютерних Наук", deanCS);

Faculty mathFaculty = facultyCreator.createFaculty("Фізико-Математичний Факультет", deanMath);

// 4. Кафедри

Department cseDepartment = departmentCreator.createDepartment("Кафедра Комп'ютерної Інженерії", headCSE);

Department isDepartment = departmentCreator.createDepartment("Кафедра Інформаційних Систем", headIS);

Department algebraDepartment = departmentCreator.createDepartment("Кафедра Алгебри та Геометрії", headAlgebra);

// 5. Групи (староста - студент)

Group groupCS101 = groupCreator.createGroup("КН-101", student1); // student1 - староста

Group groupIS102 = groupCreator.createGroup("ІС-102", student4); // student4 - староста

Group groupMA101 = groupCreator.createGroup("МА-101", null); // Без старости

// 6. студенти до груп

groupCS101.addStudent(student1);

groupCS101.addStudent(student2);

groupCS101.addStudent(student3);

groupIS102.addStudent(student4);

groupIS102.addStudent(student5);

groupMA101.addStudent(student6);

cseDepartment.addGroup(groupCS101);

isDepartment.addGroup(groupIS102);

algebraDepartment.addGroup(groupMA101);

csFaculty.addDepartment(cseDepartment);

csFaculty.addDepartment(isDepartment);

mathFaculty.addDepartment(algebraDepartment);

university.addFaculty(csFaculty);

university.addFaculty(mathFaculty);

return university;

}

}

Далі створюємо папку та створюємо класи:

Department  
package model;

import java.util.ArrayList;

import java.util.List;

public class Department extends StructuralUnit {

private List<Group> groups;

public Department(String name, Human head) {

super(name, head);

this.groups = new ArrayList<>();

}

public void addGroup(Group group) {

this.groups.add(group);

}

public List<Group> getGroups() {

return new ArrayList<>(groups);

}

public void setGroups(List<Group> groups) {

this.groups = new ArrayList<>(groups);

}

@Override

public String toString() {

return "Кафедра {" + super.toString() + ", Кількість груп=" + groups.size() + "}";

}

}

Faculty  
package model;

import java.util.ArrayList;

import java.util.List;

public class Faculty extends StructuralUnit {

private List<Department> departments;

public Faculty(String name, Human head) {

super(name, head);

this.departments = new ArrayList<>();

}

public void addDepartment(Department department) {

this.departments.add(department);

}

public List<Department> getDepartments() {

return new ArrayList<>(departments);

}

public void setDepartments(List<Department> departments) {

this.departments = new ArrayList<>(departments);

}

@Override

public String toString() {

return "Факультет {" + super.toString() + ", Кількість кафедр=" + departments.size() + "}";

}

}

Group

package model;

import java.util.ArrayList;

import java.util.List;

public class Group extends StructuralUnit {

private List<Student> students;

public Group(String name, Human head) {

super(name, head);

if (!(head instanceof Student) && head != null) {

System.*err*.println("Попередження: Голова групи '" + name + "' не є студентом.");

}

this.students = new ArrayList<>();

}

public void addStudent(Student student) {

this.students.add(student);

}

public List<Student> getStudents() {

return new ArrayList<>(students);

}

public void setStudents(List<Student> students) {

this.students = new ArrayList<>(students);

}

@Override

public String toString() {

return "Група {" + super.toString() + ", Кількість студентів=" + students.size() + "}";

}

}

Human

package model;

import java.util.Objects;

public class Human {

protected String firstName;

protected String lastName;

protected String patronymic; // По батькові

protected Sex sex;

public Human(String firstName, String lastName, String patronymic, Sex sex) {

this.firstName = firstName;

this.lastName = lastName;

this.patronymic = patronymic;

this.sex = sex;

}

// Getters

public String getFirstName() {

return firstName;

}

public String getLastName() {

return lastName;

}

public String getPatronymic() {

return patronymic;

}

public Sex getSex() {

return sex;

}

// Setters (опціонально, можна зробити клас immutable)

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public void setPatronymic(String patronymic) {

this.patronymic = patronymic;

}

public void setSex(Sex sex) {

this.sex = sex;

}

public String getFullName() {

return lastName + " " + firstName + " " + patronymic;

}

@Override

public String toString() {

return "Human{" +

"ПІБ='" + getFullName() + '\'' +

", Стать=" + sex +

'}';

}

@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

package model;

import java.util.Objects;

public abstract class StructuralUnit {

protected String name;

protected Human head; // Голова підрозділу

public StructuralUnit(String name, Human head) {

this.name = name;

this.head = head;

}

// Getters

public String getName() {

return name;

}

public Human getHead() {

return head;

}

// Setters

public void setName(String name) {

this.name = name;

}

public void setHead(Human head) {

this.head = head;

}

@Override

public String toString() {

return "Назва='" + name + '\'' +

", Керівник=" + (head != null ? head.getFullName() : "Не призначено");

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

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

StructuralUnit that = (StructuralUnit) o;

return Objects.*equals*(name, that.name) && Objects.*equals*(head, that.head);

}

@Override

public int hashCode() {

return Objects.*hash*(name, head);

}

}

University

package model;

import java.util.ArrayList;

import java.util.List;

public class University extends StructuralUnit {

private List<Faculty> faculties;

public University(String name, Human head) {

super(name, head);

this.faculties = new ArrayList<>();

}

public void addFaculty(Faculty faculty) {

this.faculties.add(faculty);

}

public List<Faculty> getFaculties() {

return new ArrayList<>(faculties);

}

public void setFaculties(List<Faculty> faculties) {

this.faculties = new ArrayList<>(faculties);

}

@Override

public String toString() {

return "Університет {" + super.toString() + ", Кількість факультетів=" + faculties.size() + "}";

}

}

Далі створюємо папку для контролерів та створюємо самі контролери:

DepartmentCreator  
package controller;

import model.Department;

import model.Human;

public class DepartmentCreator {

public Department createDepartment(String name, Human head) {

return new Department(name, head);

}

}

FacultyCreator  
package controller;

import model.Faculty;

import model.Human;

public class FacultyCreator {

public Faculty createFaculty(String name, Human head) {

return new Faculty(name, head);

}

}

GroupCreator  
package controller;

import model.Group;

import model.Human;

import model.Student; // Потрібен для перевірки типу голови групи

public class GroupCreator {

public Group createGroup(String name, Human head) {

if (head != null && !(head instanceof Student)) {

System.*err*.println("ПОМИЛКА: Голова групи '" + name + "' повинен бути студентом (або відсутнім). Встановлено null.");

head = null;

}

return new Group(name, head);

}

}

HumanCreator  
package controller;

import model.Human;

import model.Sex;

public class HumanCreator {

public Human createHuman(String firstName, String lastName, String patronymic, Sex sex) {

return new Human(firstName, lastName, patronymic, sex);

}

}

StudentCreator  
package controller;

import model.Sex;

import model.Student;

public class StudentCreator {

public Student createStudent(String firstName, String lastName, String patronymic, Sex sex) {

return new Student(firstName, lastName, patronymic, sex);

}

}

UniversityCreator

package controller;

import model.Human;

import model.University;

public class UniversityCreator {

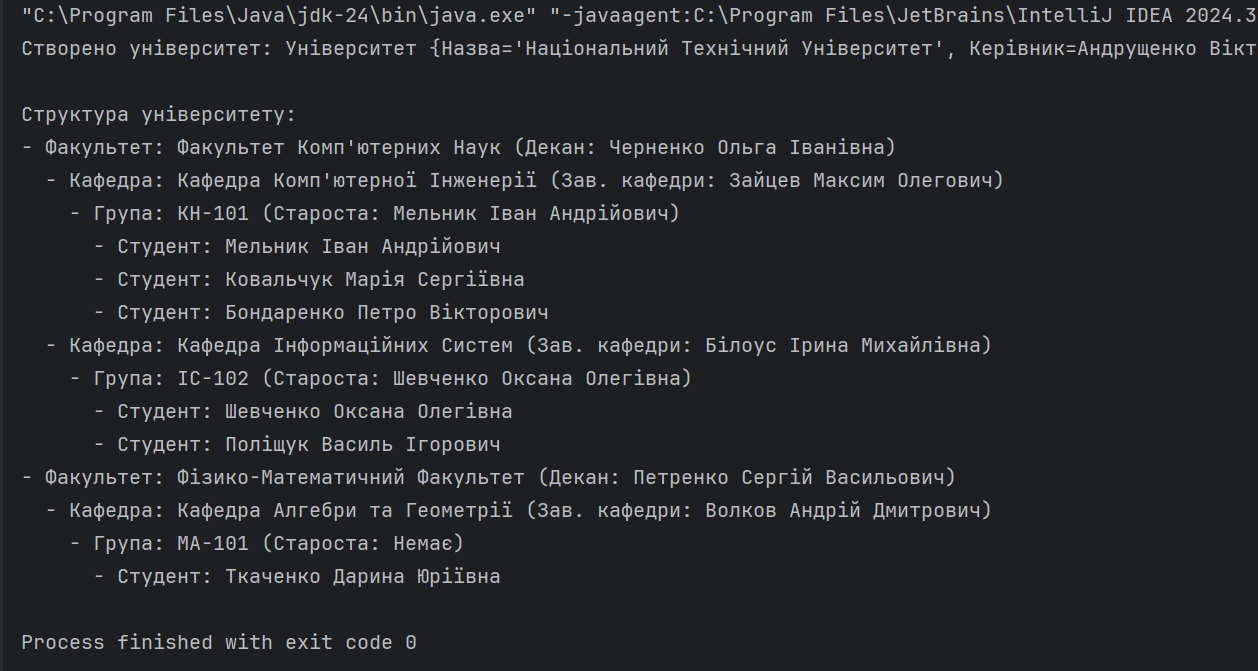
public University createUniversity(String name, Human head) {

return new University(name, head);

}

}

Результат:



**Висновок:** Ця лабораторна робота дозволила отримати практичний досвід у розробці програмної системи за патерном MVC, зосередившись на компонентах Моделі та Контролера. Було створено об'єктну модель університету з використанням ключових принципів ООП (наслідування, абстракція). Розробка класів-контролерів (Creator-ів) продемонструвала спосіб інкапсуляції логіки створення об'єктів, що покращує організацію коду.