МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний технічний університет «Дніпровська політехніка»

Изображение выглядит как текст

Автоматически созданное описание

Кафедра САіУ

ЗВІТ

з лабораторних робіт

з дисципліни «Програмування в середовищі Java»

Виконав: ст. гр. 122-20-3

Клішевич Владислав Михайлович

Перевірив: Мінєєв О.С.

Дніпро

2024

Зміст

1. Лабораторна робота №0…………………………………………………………..3

2. Лабораторна робота №1…………………………………………………………..4

3. Лабораторна робота №2…………………………………………………………..5

4. Лабораторна робота №3…………………………………………………………..8

5. Лабораторна робота №4…………………………………………………………15

6. Лабораторна робота №5…………………………………………………………23

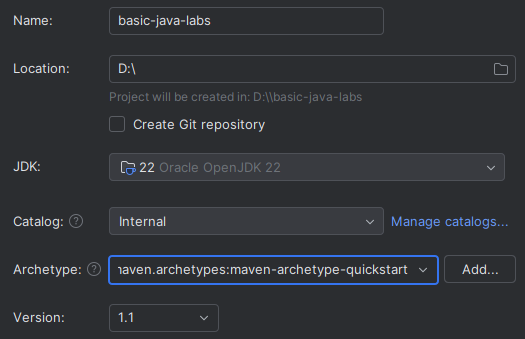
**Лабораторна робота №0**

**Тема:** Hello world.

**Завдання:** Встановити Intelliji Idea та Java jdk останньої версії. Cтворити maven проект та розробити в цьому проекті типову програму Hello world. Програма повинна видавати на екран напис Hello world та закінчувати свою роботу.

**Хід роботи:**

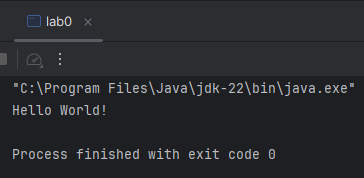
1. Створення maven проекту.



2. Лістинг коду програми.

package edu.ntudp.fit.klishevych.lab0;  
  
public class lab0  
{  
 public static void main( String[] args )  
 {  
 System.*out*.println( "Hello World!" );  
 }  
}

3. Результат виконання програми.



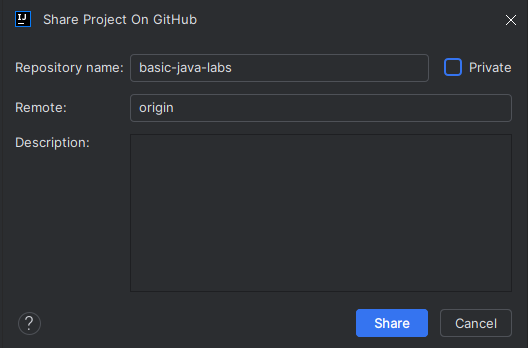
**Лабораторна робота №1**

**Тема:** CVS. GIT.

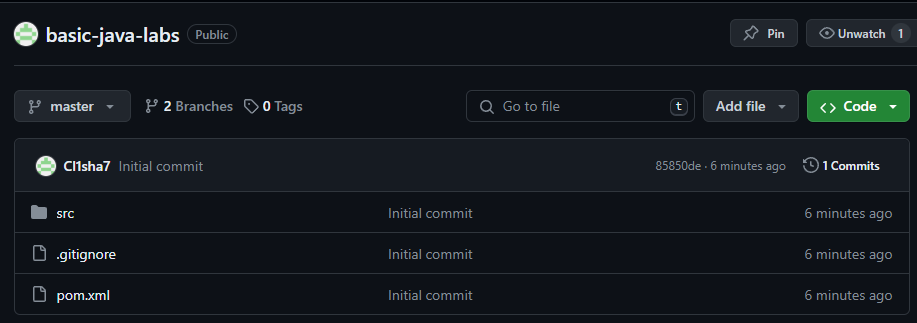
**Завдання:** Під'єднати до intellij Idea систему CVS. А саме GIT. Створити аккаунт в хмарному середовищі github, під'єднати свій проект в intellij Idea до свого аккаунту github та завантажити нульову лабораторну роботу на github аккаунт. Кожну нову лабораторну роботу робити в окремії гілці(з іменем лабораторної наприклад «LR\_3») а потім після того як її написали мержити гілку до мастера.

**Хід роботи:**

1. Під’єднуємо проект до github.



2. Репозиторій на github.



**Лабораторна робота №2**

**Тема:** Основи.

**Завдання:** Розробити програму, що дозволить вам створити, як з клавіатури так і рандомно матрицю цілих чисел типу int заданої ширини та висоти(ввести з клавіатури), але не більше 20 на 20. Створити можливість пошуку в цій матриці мінімального і максимального елементу та розрахунок середнього арифметичного. Програма може бути написана в одному класі, обов'язково розбиття на методи. Обов'язкове використання клавіатури, під час вибору ручного чи рандомного створення матриці. Створення системи зчитування з клавіатури зробити будь-яким способом, наприклад завдяки класу Scanner. Scanner являє собою найпростішу систему сканування клавіатури. Диапазон рандомних чисел для створення елементів матриці повинен зверігатись в спеціальних константах.

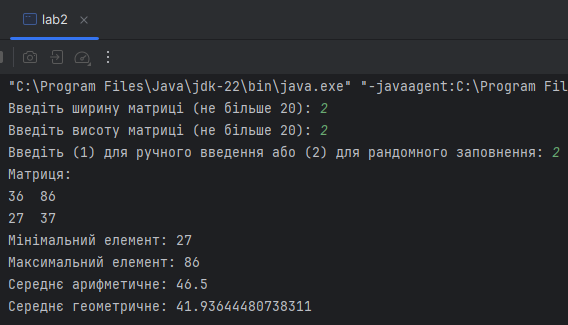
*Як завдання підвищеної складності додати розрахунок середнього геометричного елементів матриці.*

**Хід роботи:**

1. Лістинг коду програми.

package org.example;  
import java.util.Scanner;  
import java.util.Random;  
  
public class App {  
 private static final int *MAX\_SIZE* = 20;  
 private static final int *RANDOM\_MIN* = 1;  
 private static final int *RANDOM\_MAX* = 100;  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 int width, height;  
 do {  
 System.*out*.print("Введіть ширину матриці (не більше " + *MAX\_SIZE* + "): ");  
 width = scanner.nextInt();  
 System.*out*.print("Введіть висоту матриці (не більше " + *MAX\_SIZE* + "): ");  
 height = scanner.nextInt();  
 } while (width <= 0 || height <= 0 || width > *MAX\_SIZE* || height > *MAX\_SIZE*);  
  
 int[][] matrix = *createMatrix*(width, height, scanner);  
 if (matrix != null) {  
 *printMatrix*(matrix);  
 *findMinMaxAverage*(matrix);  
 *findGeometricMean*(matrix);  
 } else {  
 System.*out*.println("Помилка: неможливо створити матрицю.");  
 }  
  
 scanner.close();  
 }  
  
 private static int[][] createMatrix(int width, int height, Scanner scanner) {  
 System.*out*.print("Введіть (1) для ручного введення або (2) для рандомного заповнення: ");  
 int choice = scanner.nextInt();  
  
 int[][] matrix = new int[width][height];  
 Random random = new Random();  
  
 if (choice == 1) {  
 System.*out*.println("Введіть елементи матриці:");  
 for (int i = 0; i < width; i++) {  
 for (int j = 0; j < height; j++) {  
 matrix[i][j] = scanner.nextInt();  
 }  
 }  
 } else if (choice == 2) {  
 for (int i = 0; i < width; i++) {  
 for (int j = 0; j < height; j++) {  
 matrix[i][j] = random.nextInt(*RANDOM\_MAX* - *RANDOM\_MIN* + 1) + *RANDOM\_MIN*;  
 }  
 }  
 } else {  
 System.*out*.println("Неправильний вибір. Використано рандомне заповнення.");  
 for (int i = 0; i < width; i++) {  
 for (int j = 0; j < height; j++) {  
 matrix[i][j] = random.nextInt(*RANDOM\_MAX* - *RANDOM\_MIN* + 1) + *RANDOM\_MIN*;  
 }  
 }  
 }  
  
 return matrix;  
 }  
  
 private static void printMatrix(int[][] matrix) {  
 System.*out*.println("Матриця:");  
 for (int[] row : matrix) {  
 for (int value : row) {  
 System.*out*.print(value + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
  
 private static void findMinMaxAverage(int[][] matrix) {  
 int min = Integer.*MAX\_VALUE*;  
 int max = Integer.*MIN\_VALUE*;  
 int sum = 0;  
 int count = 0;  
  
 for (int[] row : matrix) {  
 for (int value : row) {  
 if (value < min) {  
 min = value;  
 }  
 if (value > max) {  
 max = value;  
 }  
 sum += value;  
 count++;  
 }  
 }  
  
 double average = (double) sum / count;  
  
 System.*out*.println("Мінімальний елемент: " + min);  
 System.*out*.println("Максимальний елемент: " + max);  
 System.*out*.println("Середнє арифметичне: " + average);  
 }  
 private static void findGeometricMean(int[][] matrix) {  
 long product = 1;  
 int count = 0;  
  
 for (int[] row : matrix) {  
 for (int value : row) {  
 product \*= value;  
 count++;  
 }  
 }  
  
 double geometricMean = Math.*pow*(product, 1.0 / count);  
  
 System.*out*.println("Середнє геометричне: " + geometricMean);  
 }  
}

2. Результат виконання програми.



**Лабораторна робота №3**

**Тема:** 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, по батькові 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 контролер (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.

**Хід роботи:**

1. Лістинг коду програми.

1.1. DepartmentCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Group;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Department;  
import java.util.ArrayList;  
  
public class DepartmentCreator {  
 public Department createDepartment(String name, Human head, ArrayList<Group> groupList) {  
 return new Department(name, head, groupList);  
 }  
}

1.2. FacultyCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Department;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Faculty;  
import java.util.ArrayList;  
  
public class FacultyCreator {  
 public Faculty createFaculty(String name, Human head, ArrayList<Department> departmentList) {  
 return new Faculty(name, head, departmentList);  
 }  
}

1.3. GroupCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Student;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Group;  
import java.util.ArrayList;  
  
public class GroupCreator {  
 public Group createGroup(String name, Human head, ArrayList<Student> studentList) {  
 return new Group(name, head, studentList);  
 }  
}

1.4. HumanCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Sex;  
  
public class HumanCreator {  
 public Human createHuman(String firstName, String fathersName,String lastName, Sex sex) {  
 return new Human(firstName, fathersName, lastName, sex);  
 }  
}

1.5. StudentCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Student;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Sex;  
  
public class StudentCreator {  
 public Student createStudent(String firstName, String fathersName, String lastName, Sex sex, String id) {  
 HumanCreator humanCreator = new HumanCreator();  
 Human human = humanCreator.createHuman(firstName, fathersName, lastName, sex);  
 return new Student(human.getFirstName(), human.getFathersName(), human.getLastName(), human.getSex(), id);  
 }  
}

1.6. UniversityCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.\*;  
import java.util.ArrayList;  
  
public class UniversityCreator {  
 public University createUniversity(String name, Human head, ArrayList<Faculty> facultyList) {  
 return new University(name, head, facultyList);  
 }  
 public University createTypicalUniversity() {  
 HumanCreator HumanCreator = new HumanCreator();  
 StudentCreator StudentCreator = new StudentCreator();  
 GroupCreator GroupCreator = new GroupCreator();  
 DepartmentCreator DepartmentCreator = new DepartmentCreator();  
 FacultyCreator FacultyCreator = new FacultyCreator();  
  
 Human universityHead = HumanCreator.createHuman("Олександр","Олександрович","Азюковський", Sex.*MALE*);  
 Human facultyFHead = HumanCreator.createHuman("Ірина","Михайлівна","Удовик ",Sex.*FEMALE*);  
 Human departmentHead = HumanCreator.createHuman("Михайло","Олександрович","Алексєєв",Sex.*MALE*);  
 Human groupFirstHead = HumanCreator.createHuman("Дар'я","Олегівна","Райгородська",Sex.*FEMALE*);  
 Human groupSecondHead = HumanCreator.createHuman("Максим","Сергійович","Гончаров",Sex.*MALE*);  
  
 Student studentFirst = StudentCreator.createStudent("Максим","Сергійович","Попов",Sex.*MALE*,"101");  
 Student studentSecond = StudentCreator.createStudent("Анастасія","Сергіївна","Бут",Sex.*FEMALE*,"102");  
 Student studentThird = StudentCreator.createStudent("Софія","Костянтинівна","Гейко",Sex.*FEMALE*,"201");  
 Student studentFourth = StudentCreator.createStudent("Вячеслав","Олександрович","Кастєров",Sex.*MALE*,"202");  
  
 ArrayList<Student> groupFirstStudents = new ArrayList<>();  
 groupFirstStudents.add(studentFirst);  
 groupFirstStudents.add(studentSecond);  
 ArrayList<Student> groupSecondStudents = new ArrayList<>();  
 groupSecondStudents.add(studentThird);  
 groupSecondStudents.add(studentFourth);  
 Group groupFirst = GroupCreator.createGroup("122-20-1", groupFirstHead, groupFirstStudents);  
 Group groupSecond = GroupCreator.createGroup("122-20-2", groupSecondHead, groupSecondStudents);  
  
 ArrayList<Group> departmentGroups = new ArrayList<>();  
 departmentGroups.add(groupFirst);  
 departmentGroups.add(groupSecond);  
 Department department = DepartmentCreator.createDepartment("ПЗКС", departmentHead, departmentGroups);  
  
 ArrayList<Department> facultyDepartments = new ArrayList<>();  
 facultyDepartments.add(department);  
 Faculty faculty = FacultyCreator.createFaculty("ФІТ", facultyFHead, facultyDepartments);  
  
 ArrayList<Faculty> universityFaculties = new ArrayList<>();  
 universityFaculties.add(faculty);  
 return new UniversityCreator().createUniversity("НТУ ДП", universityHead,universityFaculties);  
 }  
}

1.7. Department.

package edu.ntudp.fit.klishevych.lab3.model;  
import java.util.List;  
  
public class Department extends Structure<Group> {  
 public Department(String name, Human head, List<Group> subElement) {  
 super(name, head, subElement);  
 }  
}

1.8. Faculty.

package edu.ntudp.fit.klishevych.lab3.model;  
import java.util.List;  
  
public class Faculty extends Structure<Department> {  
 public Faculty(String name, Human head, List<Department> subElement) {  
 super(name, head, subElement);  
 }  
}

1.9. Group.

package edu.ntudp.fit.klishevych.lab3.model;  
import java.util.List;  
  
public class Group extends Structure<Student> {  
 public Group(String name, Human head, List<Student> subElement) {  
 super(name, head, subElement);  
 }  
}

1.10. Human.

package edu.ntudp.fit.klishevych.lab3.model;  
  
public class Human {  
 private String firstName;  
 private String fathersName;  
 private String lastName;  
 private Sex sex;  
  
 public Human(String firstName, String fathersName, String lastName, Sex sex) {  
 this.firstName = firstName;  
 this.fathersName = fathersName;  
 this.lastName = lastName;  
 this.sex = sex;  
 }  
 public String getFirstName() {  
 return firstName;  
 }  
 public String getFathersName() {  
 return fathersName;  
 }  
 public String getLastName() {  
 return lastName;  
 }  
 public Sex getSex() {  
 return sex;  
 }  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
 public void setFathersName(String fathersName) {  
 this.fathersName = fathersName;  
 }  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
 public void setSex(Sex sex) {  
 this.sex = sex;  
 }  
}

1.11. Sex.

package edu.ntudp.fit.klishevych.lab3.model;  
  
public enum Sex {  
 *MALE*,  
 *FEMALE*}

1.12. Structure.

package edu.ntudp.fit.klishevych.lab3.model;  
import java.util.List;  
  
public abstract class Structure<T> {  
 private String name;  
 private List<T> Subordinates;  
 private Human head;  
  
 public List<T> getSubordinates() {  
 return Subordinates;  
 }  
 public Structure(String name, Human head, List<T> Subordinates) {  
 this.name = name;  
 this.head = head;  
 this.Subordinates = Subordinates;  
 }  
 public void setSubordinates(List<T> Subordinates) {  
 this.Subordinates = Subordinates;  
 }  
 public String getName() {  
 return name;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public Human getHead() {  
 return head;  
 }  
 public void setHead(Human head) {  
 this.head = head;  
 }  
}

1.13. Student.

package edu.ntudp.fit.klishevych.lab3.model;  
  
public class Student extends Human {  
 private String id;  
 public String getId() {  
 return id;  
 }  
 public void setId(String id) {  
 this.id = id;  
 }  
 public Student(String firstName, String fathersName, String lastName, Sex sex, String id) {  
 super(firstName, fathersName, lastName, sex);  
 setId(id);  
 }  
}

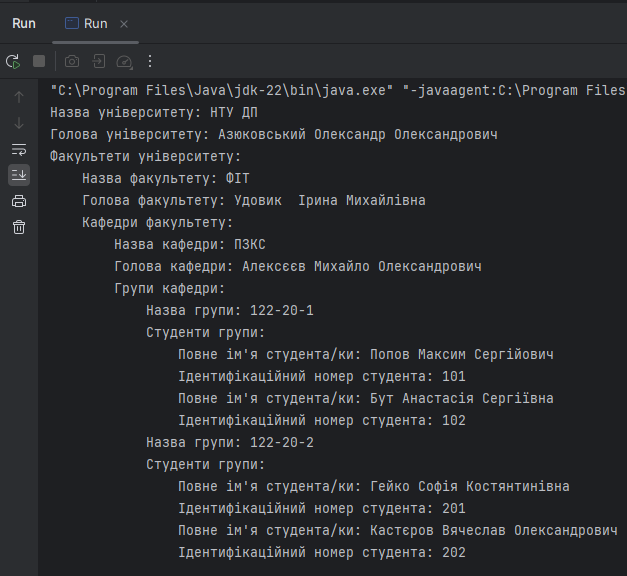
1.14. University.

package edu.ntudp.fit.klishevych.lab3.model;  
import java.util.List;  
  
public class University extends Structure<Faculty> {  
 public University(String name, Human head, List<Faculty> subElement ) {  
 super(name, head, subElement);  
 }  
}

1.15. Run.

package edu.ntudp.fit.klishevych.lab3;  
  
import edu.ntudp.fit.klishevych.lab3.controller.UniversityCreator;  
import edu.ntudp.fit.klishevych.lab3.model.University;  
  
public class Run {  
 public static void main(String[] args) {  
 UniversityCreator universityCreator = new UniversityCreator();  
 University university = universityCreator.createTypicalUniversity();  
  
 System.*out*.println("Назва університету: " + university.getName());  
 System.*out*.println("Голова університету: " + university.getHead().getLastName() + " " +  
 university.getHead().getFirstName() + " " + university.getHead().getFathersName());  
 System.*out*.println("Факультети університету:");  
 university.getSubordinates().forEach(faculty -> {  
 System.*out*.println("\tНазва факультету: " + faculty.getName());  
 System.*out*.println("\tГолова факультету: " + faculty.getHead().getLastName() + " " +  
 faculty.getHead().getFirstName() + " " + faculty.getHead().getFathersName());  
 System.*out*.println("\tКафедри факультету:");  
 faculty.getSubordinates().forEach(department -> {  
 System.*out*.println("\t\tНазва кафедри: " + department.getName());  
 System.*out*.println("\t\tГолова кафедри: " + department.getHead().getLastName() + " " +  
 department.getHead().getFirstName() + " " + department.getHead().getFathersName());  
 System.*out*.println("\t\tГрупи кафедри:");  
 department.getSubordinates().forEach(group -> {  
 System.*out*.println("\t\t\tНазва групи: " + group.getName());  
 System.*out*.println("\t\t\tСтуденти групи:");  
 group.getSubordinates().forEach(student -> {  
 System.*out*.println("\t\t\t\tПовне ім'я студента/ки: " +  
 student.getLastName() + " " + student.getFirstName() + " " + student.getFathersName());  
 System.*out*.println("\t\t\t\tІдентифікаційний номер студента: " + student.getId());  
 });  
 });  
 });  
 });  
 }  
}

2. Результат виконання програми.



**Лабораторна робота №4**

**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 проекті 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 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 не в якому разі.

**Хід роботи:**

1. Лістинг коду програми.

1.1. DepartmentCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Group;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Department;  
import java.util.ArrayList;  
  
public class DepartmentCreator {  
 public Department createDepartment(String name, Human head, ArrayList<Group> groupList) {  
 return new Department(name, head, groupList);  
 }  
}

1.2. FacultyCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Department;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Faculty;  
import java.util.ArrayList;  
  
public class FacultyCreator {  
 public Faculty createFaculty(String name, Human head, ArrayList<Department> departmentList) {  
 return new Faculty(name, head, departmentList);  
 }  
}

1.3. GroupCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Student;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Group;  
import java.util.ArrayList;  
  
public class GroupCreator {  
 public Group createGroup(String name, Human head, ArrayList<Student> studentList) {  
 return new Group(name, head, studentList);  
 }  
}

1.4. HumanCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Sex;  
  
public class HumanCreator {  
 public Human createHuman(String firstName, String fathersName,String lastName, Sex sex) {  
 return new Human(firstName, fathersName, lastName, sex);  
 }  
}

1.5. StudentCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.Student;  
import edu.ntudp.fit.klishevych.lab3.model.Human;  
import edu.ntudp.fit.klishevych.lab3.model.Sex;  
  
public class StudentCreator {  
 public Student createStudent(String firstName, String fathersName, String lastName, Sex sex, String id) {  
 HumanCreator humanCreator = new HumanCreator();  
 Human human = humanCreator.createHuman(firstName, fathersName, lastName, sex);  
 return new Student(human.getFirstName(), human.getFathersName(), human.getLastName(), human.getSex(), id);  
 }  
}

1.6. UniversityCreator.

package edu.ntudp.fit.klishevych.lab3.controller;  
import edu.ntudp.fit.klishevych.lab3.model.\*;  
import java.util.ArrayList;  
  
public class UniversityCreator {  
 public University createUniversity(String name, Human head, ArrayList<Faculty> facultyList) {  
 return new University(name, head, facultyList);  
 }  
 public University createTypicalUniversity() {  
 HumanCreator HumanCreator = new HumanCreator();  
 StudentCreator StudentCreator = new StudentCreator();  
 GroupCreator GroupCreator = new GroupCreator();  
 DepartmentCreator DepartmentCreator = new DepartmentCreator();  
 FacultyCreator FacultyCreator = new FacultyCreator();  
  
 Human universityHead = HumanCreator.createHuman("Олександр","Олександрович","Азюковський", Sex.*MALE*);  
 Human facultyFHead = HumanCreator.createHuman("Ірина","Михайлівна","Удовик ",Sex.*FEMALE*);  
 Human departmentHead = HumanCreator.createHuman("Михайло","Олександрович","Алексєєв",Sex.*MALE*);  
 Human groupFirstHead = HumanCreator.createHuman("Дар'я","Олегівна","Райгородська",Sex.*FEMALE*);  
 Human groupSecondHead = HumanCreator.createHuman("Максим","Сергійович","Гончаров",Sex.*MALE*);  
  
 Student studentFirst = StudentCreator.createStudent("Максим","Сергійович","Попов",Sex.*MALE*,"101");  
 Student studentSecond = StudentCreator.createStudent("Анастасія","Сергіївна","Бут",Sex.*FEMALE*,"102");  
 Student studentThird = StudentCreator.createStudent("Софія","Костянтинівна","Гейко",Sex.*FEMALE*,"201");  
 Student studentFourth = StudentCreator.createStudent("Вячеслав","Олександрович","Кастєров",Sex.*MALE*,"202");  
  
 ArrayList<Student> groupFirstStudents = new ArrayList<>();  
 groupFirstStudents.add(studentFirst);  
 groupFirstStudents.add(studentSecond);  
 ArrayList<Student> groupSecondStudents = new ArrayList<>();  
 groupSecondStudents.add(studentThird);  
 groupSecondStudents.add(studentFourth);  
 Group groupFirst = GroupCreator.createGroup("122-20-1", groupFirstHead, groupFirstStudents);  
 Group groupSecond = GroupCreator.createGroup("122-20-2", groupSecondHead, groupSecondStudents);  
  
 ArrayList<Group> departmentGroups = new ArrayList<>();  
 departmentGroups.add(groupFirst);  
 departmentGroups.add(groupSecond);  
 Department department = DepartmentCreator.createDepartment("ПЗКС", departmentHead, departmentGroups);  
  
 ArrayList<Department> facultyDepartments = new ArrayList<>();  
 facultyDepartments.add(department);  
 Faculty faculty = FacultyCreator.createFaculty("ФІТ", facultyFHead, facultyDepartments);  
  
 ArrayList<Faculty> universityFaculties = new ArrayList<>();  
 universityFaculties.add(faculty);  
 return new UniversityCreator().createUniversity("НТУ ДП", universityHead,universityFaculties);  
 }  
}

1.7. Department.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.List;  
import java.util.Objects;  
  
public class Department extends Structure<Group> {  
 public Department(String name, Human head, List<Group> subElement) {  
 super(name, head, subElement);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Department)) return false;  
 Department that = (Department) o;  
 return Objects.*equals*(getName(), that.getName()) &&  
 Objects.*equals*(getHead(), that.getHead()) &&  
 Objects.*equals*(getSubordinates(), that.getSubordinates());  
 }  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(getName(), getHead(), getSubordinates());  
 }  
}

1.8. Faculty.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.List;  
import java.util.Objects;  
  
public class Faculty extends Structure<Department> {  
 public Faculty(String name, Human head, List<Department> subElement) {  
 super(name, head, subElement);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Faculty)) return false;  
 Faculty faculty = (Faculty) o;  
 return Objects.*equals*(getName(), faculty.getName()) &&  
 Objects.*equals*(getHead(), faculty.getHead()) &&  
 Objects.*equals*(getSubordinates(), faculty.getSubordinates());  
 }  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(getName(), getHead(), getSubordinates());  
 }  
}

1.9. Group.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.List;  
import java.util.Objects;  
  
public class Group extends Structure<Student> {  
 public Group(String name, Human head, List<Student> subElement) {  
 super(name, head, subElement);  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Group)) return false;  
 Group group = (Group) o;  
 return Objects.*equals*(getName(), group.getName()) &&  
 Objects.*equals*(getHead(), group.getHead()) &&  
 Objects.*equals*(getSubordinates(), group.getSubordinates());  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(getName(), getHead(), getSubordinates());  
 }  
}

1.10. Human.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.Objects;  
  
public class Human {  
 private String firstName;  
 private String fathersName;  
 private String lastName;  
 private Sex sex;  
  
 public Human(String firstName, String fathersName, String lastName, Sex sex) {  
 this.firstName = firstName;  
 this.fathersName = fathersName;  
 this.lastName = lastName;  
 this.sex = sex;  
 }  
 public String getFirstName() {  
 return firstName;  
 }  
 public String getFathersName() {  
 return fathersName;  
 }  
 public String getLastName() {  
 return lastName;  
 }  
 public Sex getSex() {  
 return sex;  
 }  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
 public void setFathersName(String fathersName) {  
 this.fathersName = fathersName;  
 }  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
 public void setSex(Sex sex) {  
 this.sex = 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*(fathersName, human.fathersName) &&  
 Objects.*equals*(lastName, human.lastName) &&  
 sex == human.sex;  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(firstName, fathersName, lastName, sex);  
 }  
}

1.11. Sex.

package edu.ntudp.fit.klishevych.lab3.model;  
  
public enum Sex {  
 *MALE*,  
 *FEMALE*}

1.12. Structure.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.Objects;  
import java.util.List;  
  
public abstract class Structure<T> {  
 private String name;  
 private List<T> Subordinates;  
 private Human head;  
  
 public List<T> getSubordinates() {  
 return Subordinates;  
 }  
 public Structure(String name, Human head, List<T> Subordinates) {  
 this.name = name;  
 this.head = head;  
 this.Subordinates = Subordinates;  
 }  
 public void setSubordinates(List<T> Subordinates) {  
 this.Subordinates = Subordinates;  
 }  
 public String getName() {  
 return name;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public Human getHead() {  
 return head;  
 }  
 public void setHead(Human head) {  
 this.head = head;  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Structure)) return false;  
 Structure<?> structure = (Structure<?>) o;  
 return Objects.*equals*(name, structure.name) &&  
 Objects.*equals*(Subordinates, structure.Subordinates) &&  
 Objects.*equals*(head, structure.head);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(name, Subordinates, head);  
 }  
}

1.13. Student.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.Objects;  
  
public class Student extends Human {  
 private String id;  
 public String getId() {  
 return id;  
 }  
 public void setId(String id) {  
 this.id = id;  
 }  
 public Student(String firstName, String fathersName, String lastName, Sex sex, String id) {  
 super(firstName, fathersName, lastName, sex);  
 setId(id);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Student)) return false;  
 if (!super.equals(o)) return false; // Перевірка наслідуваного методу equals з класу Human  
 Student student = (Student) o;  
 return Objects.*equals*(getId(), student.getId());  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(super.hashCode(), getId());  
 }  
}

1.14. University.

package edu.ntudp.fit.klishevych.lab4.model;  
import java.util.List;  
import java.util.Objects;  
  
public class University extends Structure<Faculty> {  
 public University(String name, Human head, List<Faculty> subElement ) {  
 super(name, head, subElement);  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof University)) return false;  
 University that = (University) o;  
 return Objects.*equals*(getName(), that.getName()) &&  
 Objects.*equals*(getHead(), that.getHead()) &&  
 Objects.*equals*(getSubordinates(), that.getSubordinates());  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(getName(), getHead(), getSubordinates());  
 }  
}

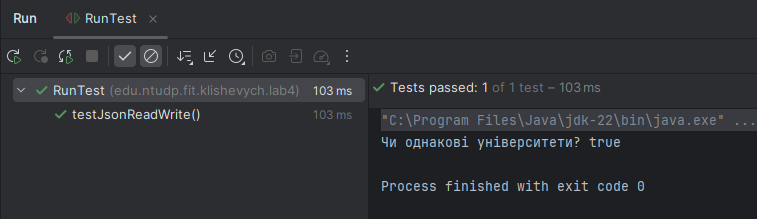
1.15. Run.

package edu.ntudp.fit.klishevych.lab3;  
  
import edu.ntudp.fit.klishevych.lab3.controller.UniversityCreator;  
import edu.ntudp.fit.klishevych.lab3.model.University;  
  
public class Run {  
 public static void main(String[] args) {  
 UniversityCreator universityCreator = new UniversityCreator();  
 University university = universityCreator.createTypicalUniversity();  
  
 System.*out*.println("Назва університету: " + university.getName());  
 System.*out*.println("Голова університету: " + university.getHead().getLastName() + " " +  
 university.getHead().getFirstName() + " " + university.getHead().getFathersName());  
 System.*out*.println("Факультети університету:");  
 university.getSubordinates().forEach(faculty -> {  
 System.*out*.println("\tНазва факультету: " + faculty.getName());  
 System.*out*.println("\tГолова факультету: " + faculty.getHead().getLastName() + " " +  
 faculty.getHead().getFirstName() + " " + faculty.getHead().getFathersName());  
 System.*out*.println("\tКафедри факультету:");  
 faculty.getSubordinates().forEach(department -> {  
 System.*out*.println("\t\tНазва кафедри: " + department.getName());  
 System.*out*.println("\t\tГолова кафедри: " + department.getHead().getLastName() + " " +  
 department.getHead().getFirstName() + " " + department.getHead().getFathersName());  
 System.*out*.println("\t\tГрупи кафедри:");  
 department.getSubordinates().forEach(group -> {  
 System.*out*.println("\t\t\tНазва групи: " + group.getName());  
 System.*out*.println("\t\t\tСтуденти групи:");  
 group.getSubordinates().forEach(student -> {  
 System.*out*.println("\t\t\t\tПовне ім'я студента/ки: " +  
 student.getLastName() + " " + student.getFirstName() + " " + student.getFathersName());  
 System.*out*.println("\t\t\t\tІдентифікаційний номер студента: " + student.getId());  
 });  
 });  
 });  
 });  
 }  
}

1.16. RunTest.

package edu.ntudp.fit.klishevych.lab4;  
  
import edu.ntudp.fit.klishevych.lab4.controller.UniversityCreator;  
import edu.ntudp.fit.klishevych.lab4.model.University;  
import org.junit.jupiter.api.Test;  
import static org.junit.jupiter.api.Assertions.*assertEquals*;  
  
public class RunTest {  
  
 @Test  
 void testJsonReadWrite() {  
 UniversityCreator universityCreator = new UniversityCreator();  
 University oldUniversity = universityCreator.createTypicalUniversity();  
  
 JsonManager jsonManager = new JsonManager();  
 jsonManager.writeToJsonFile(oldUniversity, "University.json");  
  
 University newUniversity = jsonManager.readFromJsonFile("University.json");  
  
 System.*out*.println("Чи однакові університети? " + oldUniversity.equals(newUniversity));  
 *assertEquals*(oldUniversity, newUniversity);  
 }  
}

2. Результат виконання програми.



**Лабораторна робота №5**

**Тема:** Jdbc.

**Завдання:** Створити базу даних в будь якому сервері баз даних. Створити таблицю з переліком студентів вказати їх прізвище, ім'я, по батькові, день народження номер залікової книжки та ID. Створити програму що буде дозволяти виводити на екран інформацію про студентів які народилися в тому чи іншому місяці року. Програма повинна завдяки системі jdbc під'єднатися до вашої бази даних та робити до неї запроси. Вимог до розробки бази даних немає. Програма ж має бути написана за усіма стандартами ООП. Та може бути спроектована за двох принципів:

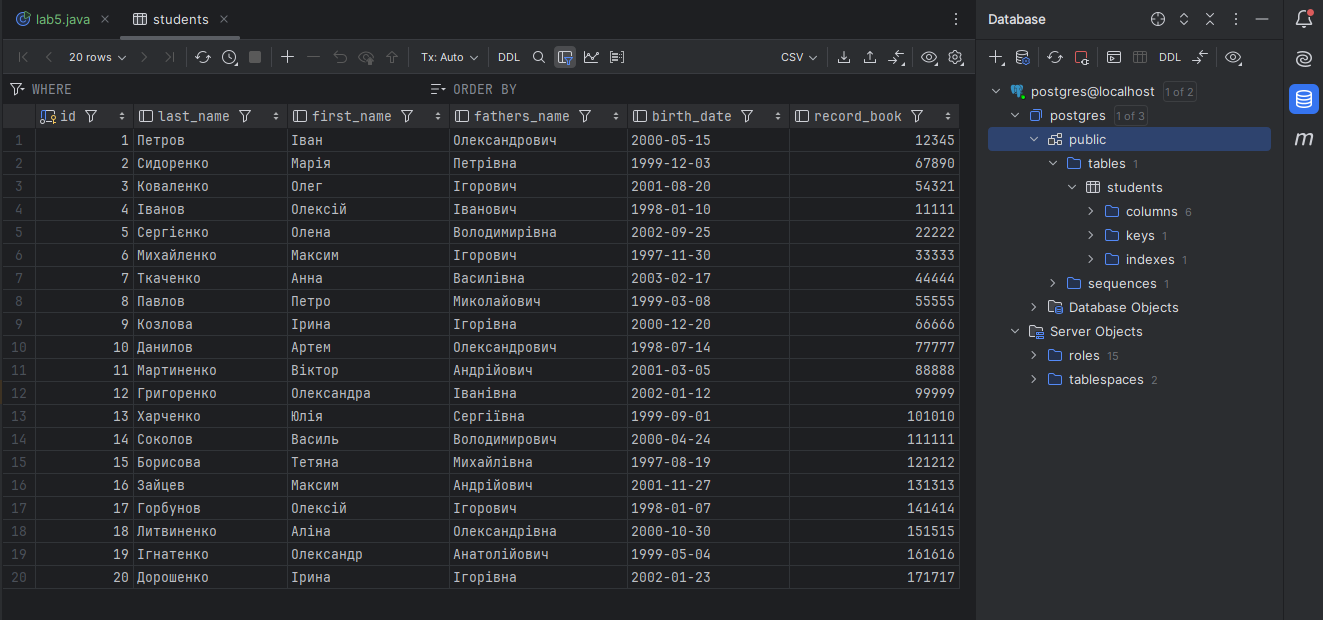
- при будь-якій ситуації буде забиратися весь перелік студентів, а вже на стороні java буде зроблено пошук необхідного

- SQL запрос буде сформований згідно запросу який зробив користувач і вже сервер управління баз даних буде вирішувати, які самі студенти народилися в тому чи іншому місяці.

У висновку обов'язково пояснити чому вибрали той чи інший принцип, які в нього переваги та недоліки. Оцінка не залежить від того який сервер управління баз даних вибрали. Перелік студентів зробити не менше 20 людей. Місяць червень зробити місяцем, коли в жодного зі студентів немає дня народження. SQL код створення бази даних розмістити в проекті 5 лабораторної роботи в файлі database в пекеджі resources. Для використання цієї лабораторної роботи рекомендується активно використовувати знання отримані на дисципліні що стосуються розробки баз даних.

**Хід роботи:**

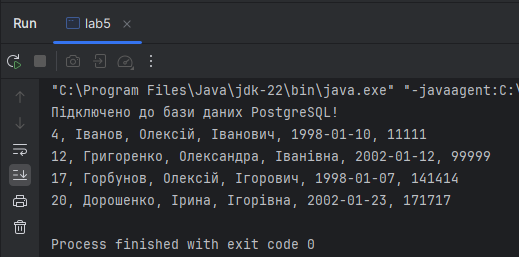
1. Створена таблиця.



2. Лістинг коду програми.

package edu.ntudp.fit.klishevych.lab5;  
  
import java.sql.\*;  
  
public class lab5 {  
 private static final String *URL* = "jdbc:postgresql://localhost/postgres";  
 private static final String *USER* = "postgres";  
 private static final String *PASSWORD* = "123";  
  
 public static void main(String[] args) {  
 try (Connection connection = DriverManager.*getConnection*(*URL*, *USER*, *PASSWORD*)) {  
 System.*out*.println("Підключено до бази даних PostgreSQL!");  
  
 int targetMonth = 1;  
  
 String sql = "SELECT id, last\_name, first\_name, fathers\_name, birth\_date, record\_book " +  
 "FROM students WHERE EXTRACT(MONTH FROM birth\_date) = ?";  
 try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {  
 preparedStatement.setInt(1, targetMonth);  
 ResultSet resultSet = preparedStatement.executeQuery();  
  
 while (resultSet.next()) {  
 int id = resultSet.getInt("id");  
 String lastName = resultSet.getString("last\_name");  
 String firstName = resultSet.getString("first\_name");  
 String fathersName = resultSet.getString("fathers\_name");  
 String birthDate = resultSet.getString("birth\_date");  
 int recordBook = resultSet.getInt("record\_book");  
  
 System.*out*.printf("%d, %s, %s, %s, %s, %d%n",  
 id, lastName, firstName, fathersName, birthDate, recordBook);  
 }  
 }  
 } catch (SQLException e) {  
 System.*out*.println("Помилка підключення до бази даних.");  
 e.printStackTrace();  
 }  
 }  
}

3. Результат виконання програми.



4. Висновок: Я обрав принцип «SQL запрос буде сформований згідно запросу який зробив користувач і вже сервер управління баз даних буде вирішувати, які самі студенти народилися в тому чи іншому місяці.», бо для мене він є зручнішим. Цей принцип дозволяє швидше та ефективніше виконувати запити, а також зменшує обсяг даних, що передаються по мережі.