|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

**по лабораторной работе № 3**

**Вариант № 12**

**Название:** классы, наследование и полиморфизм

**Дисциплина:** языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | Н.М. Кулинич |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель:** освоить принципы ООП на языке программирования Java.

**Задание 1:** определить класс Вектор в R3. Реализовать методы для проверки векторов на ортогональность, проверки пересечения не ортогональных векторов, сравнения векторов. Создать массив из m объектов. Определить, какие из векторов компланарны.

Код класса VectorR3:

import java.util.ArrayList;

import java.util.Scanner;

public class VectorR3 {

private double x, y, z;

public VectorR3(double x, double y, double z) {

this.x = x;

this.y = y;

this.z = z;

}

// Метод для проверки ортогональности с другим вектором

public boolean isOrthogonal(VectorR3 other) {

return (this.x \* other.x + this.y \* other.y + this.z \* other.z) == 0;

}

// Метод для проверки пересечения с другим неортогональным вектором

public boolean intersects(VectorR3 other) {

return !this.isCollinear(other) && !this.isOrthogonal(other);

}

// Проверка на коллинеарность

private boolean isCollinear(VectorR3 other) {

double kx = this.x / other.x;

double ky = this.y / other.y;

double kz = this.z / other.z;

return (kx == ky) && (ky == kz);

}

// Проверка на компланарность

public static boolean areCoplanar(VectorR3 v1, VectorR3 v2, VectorR3 v3) {

double volume = v1.x \* (v2.y \* v3.z - v3.y \* v2.z) -

v1.y \* (v2.x \* v3.z - v3.x \* v2.z) +

v1.z \* (v2.x \* v3.y - v3.x \* v2.y);

return volume == 0;

}

// Метод для сравнения векторов

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (!(obj instanceof VectorR3)) return false;

VectorR3 other = (VectorR3) obj;

return this.x == other.x && this.y == other.y && this.z == other.z;

}

@Override

public String toString() {

return String.format("Vector(%f, %f, %f)", x, y, z);

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Введите количество векторов:");

int n = scanner.nextInt();

ArrayList<VectorR3> vectors = new ArrayList<>();

for (int i = 0; i < n; i++) {

System.out.println("Введите координаты вектора " + (i + 1) + " (x y z):");

double x = scanner.nextDouble();

double y = scanner.nextDouble();

double z = scanner.nextDouble();

vectors.add(new VectorR3(x, y, z));

}

// Проверяем ортогональность всех пар векторов

for (int i = 0; i < vectors.size(); i++) {

for (int j = i + 1; j < vectors.size(); j++) {

if (vectors.get(i).isOrthogonal(vectors.get(j))) {

System.out.printf("%s is orthogonal to %s%n", vectors.get(i), vectors.get(j));

}

}

}

// Проверяем компланарность всех троек векторов

for (int i = 0; i < vectors.size(); i++) {

for (int j = i + 1; j < vectors.size(); j++) {

for (int k = j + 1; k < vectors.size(); k++) {

if (areCoplanar(vectors.get(i), vectors.get(j), vectors.get(k))) {

System.out.printf("%s, %s, and %s are coplanar%n", vectors.get(i), vectors.get(j), vectors.get(k));

}

}

}

}

scanner.close();

}

}}Работа программы показана на рисунке 1.

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание

Рисунок 1 – Работа программы

**Задание 2:** определить класс Матрица размерности (n x n). Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения матриц. Объявить массив объектов. Создать методы, вычисляющие первую и вторую нормы матрицы. Определить, какая из матриц имеет наименьшую первую и вторую нормы.

Код класса Matrix:

public class Matrix {

private final double[][] data;

private final int n;

public Matrix(int size) {

this.n = size;

data = new double[size][size];

}

public Matrix(double[][] data) {

if (data.length != data[0].length) {

throw new IllegalArgumentException("Matrix must be square.");

}

this.n = data.length;

this.data = new double[n][n];

for (int i = 0; i < n; i++) {

System.arraycopy(data[i], 0, this.data[i], 0, n);

}

}

// Сложение матриц

public Matrix add(Matrix other) {

Matrix result = new Matrix(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

result.data[i][j] = this.data[i][j] + other.data[i][j];

}

}

return result;

}

// Вычитание матриц

public Matrix subtract(Matrix other) {

Matrix result = new Matrix(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

result.data[i][j] = this.data[i][j] - other.data[i][j];

}

}

return result;

}

// Умножение матриц

public Matrix multiply(Matrix other) {

Matrix result = new Matrix(n);

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

for (int k = 0; k < n; k++) {

result.data[i][j] += this.data[i][k] \* other.data[k][j];

}

}

}

return result;

}

// Вычисление первой нормы матрицы

public double firstNorm() {

double maxSum = 0;

for (int i = 0; i < n; i++) {

double sum = 0;

for (int j = 0; j < n; j++) {

sum += Math.abs(data[i][j]);

}

maxSum = Math.max(maxSum, sum);

}

return maxSum;

}

// Вычисление второй нормы матрицы

public double secondNorm() {

double maxSum = 0;

for (int j = 0; j < n; j++) {

double sum = 0;

for (int i = 0; i < n; i++) {

sum += Math.abs(data[i][j]);

}

maxSum = Math.max(maxSum, sum);

}

return maxSum;

}

// Вспомогательный метод для вывода матрицы

public void printMatrix() {

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.printf("%8.2f", data[i][j]);

}

System.out.println();

}

}

public static void main(String[] args) {

Matrix[] matrices = {

new Matrix(new double[][]{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}),

new Matrix(new double[][]{{9, 8, 7}, {6, 5, 4}, {3, 2, 1}}),

};

// Инициализация для отслеживания минимальных норм

double minFirstNorm = Double.MAX\_VALUE;

double minSecondNorm = Double.MAX\_VALUE;

Matrix minFirstNormMatrix = null;

Matrix minSecondNormMatrix = null;

// Вычисление минимальных норм для массива матриц

for (Matrix matrix : matrices) {

double firstNorm = matrix.firstNorm();

double secondNorm = matrix.secondNorm();

if (firstNorm < minFirstNorm) {

minFirstNorm = firstNorm;

minFirstNormMatrix = matrix;

}

if (secondNorm < minSecondNorm) {

minSecondNorm = secondNorm;

minSecondNormMatrix = matrix;

}

}

// Вывод матриц с наименьшей первой и второй нормами

System.out.println("Matrix with the smallest first norm:");

if (minFirstNormMatrix != null) {

minFirstNormMatrix.printMatrix();

}

System.out.println("\nMatrix with the smallest second norm:");

if (minSecondNormMatrix != null) {

minSecondNormMatrix.printMatrix();

}

}

} Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2 – Работа программы

Работа программы показана на рисунке 2.

**Задание 3:** создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. Patient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Номер медицинской карты, Диагноз. Создать массив объектов. Вывести: a) список пациентов, имеющих данный диагноз; b) список пациентов, номер медицинской карты у которых находится в заданном интервале.

Код класса Patient:

public class Patient {

private int id;

private String lastName;

private String firstName;

private String middleName;

private String address;

private String phone;

private String medicalCardNumber;

private String diagnosis;

// Конструктор

public Patient(int id, String lastName, String firstName, String middleName,

String address, String phone, String medicalCardNumber, String diagnosis) {

this.id = id;

this.lastName = lastName;

this.firstName = firstName;

this.middleName = middleName;

this.address = address;

this.phone = phone;

this.medicalCardNumber = medicalCardNumber;

this.diagnosis = diagnosis;

}

// Методы set и get для каждого поля

public void setId(int id) { this.id = id; }

public int getId() { return id; }

public void setLastName(String lastName) { this.lastName = lastName; }

public String getLastName() { return lastName; }

public void setFirstName(String firstName) { this.firstName = firstName; }

public String getFirstName() { return firstName; }

public void setMiddleName(String middleName) { this.middleName = middleName; }

public String getMiddleName() { return middleName; }

public void setAddress(String address) { this.address = address; }

public String getAddress() { return address; }

public void setPhone(String phone) { this.phone = phone; }

public String getPhone() { return phone; }

public void setMedicalCardNumber(String medicalCardNumber) { this.medicalCardNumber = medicalCardNumber; }

public String getMedicalCardNumber() { return medicalCardNumber; }

public void setDiagnosis(String diagnosis) { this.diagnosis = diagnosis; }

public String getDiagnosis() { return diagnosis; }

// toString метод для представления объекта в виде строки

@Override

public String toString() {

return "Patient{" +

"id=" + id +

", lastName='" + lastName + '\'' +

", firstName='" + firstName + '\'' +

", middleName='" + middleName + '\'' +

", address='" + address + '\'' +

", phone='" + phone + '\'' +

", medicalCardNumber='" + medicalCardNumber + '\'' +

", diagnosis='" + diagnosis + '\'' +

'}';

}

// Метод для вывода списка пациентов с данным диагнозом

public static void printPatientsWithDiagnosis(Patient[] patients, String diagnosis) {

for (Patient patient : patients) {

if (patient.getDiagnosis().equalsIgnoreCase(diagnosis)) {

System.out.println(patient);

}

}

}

// Метод для вывода списка пациентов, номер медицинской карты которых находится в заданном интервале

public static void printPatientsByMedicalCardRange(Patient[] patients, String minNumber, String maxNumber) {

for (Patient patient : patients) {

if (patient.getMedicalCardNumber().compareTo(minNumber) >= 0 &&

patient.getMedicalCardNumber().compareTo(maxNumber) <= 0) {

System.out.println(patient);

}

}

}

public static void main(String[] args) {

// Создание массива объектов класса Patient

Patient[] patients = {

new Patient(1, "Иванов", "Иван", "Иванович", "г. Москва", "327283", "123456", "Грипп"),

new Patient(2, "Петров", "Петр", "Петрович", "г. Москва", "459267", "203559", "Грипп"),

new Patient(3, "Иванов", "Николай", "Иванович", "г. Москва", "273829", "123453", "ОРВИ")

};

// Вывод пациентов с диагнозом "Грипп"

System.out.println("Пациенты с диагнозом Грипп:");

printPatientsWithDiagnosis(patients, "Грипп");

// Вывод пациентов с номером медицинской карты в заданном интервале

System.out.println("\nПациенты с номером медицинской карты от 123450 до 123459:");

printPatientsByMedicalCardRange(patients, "123450", "123459");

}

}}Работа программы показана на рисунке 3.

Изображение выглядит как текст, снимок экрана, Шрифт, информация

Автоматически созданное описание

Рисунок 3 – Работа программы

**Задание 4:** создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. Abiturient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки. Создать массив объектов. Вывести: a) список абитуриентов, имеющих неудовлетворительные оценки; b) список абитуриентов, средний балл у которых выше заданного; c) выбрать заданное число n абитуриентов, имеющих самый высокий средний балл (вывести также полный список абитуриентов, имеющих полупроходной балл).

Код модуля Main:

import java.util.Arrays;

import java.util.Comparator;

public class Abiturient {

private int id;

private String lastName;

private String firstName;

private String middleName;

private String address;

private String phone;

private int[] grades;

// Конструктор

public Abiturient(int id, String lastName, String firstName, String middleName,

String address, String phone, int[] grades) {

this.id = id;

this.lastName = lastName;

this.firstName = firstName;

this.middleName = middleName;

this.address = address;

this.phone = phone;

this.grades = grades;

}

// Методы set и get

public void setId(int id) { this.id = id; }

public int getId() { return id; }

public void setLastName(String lastName) { this.lastName = lastName; }

public String getLastName() { return lastName; }

public void setFirstName(String firstName) { this.firstName = firstName; }

public String getFirstName() { return firstName; }

public void setMiddleName(String middleName) { this.middleName = middleName; }

public String getMiddleName() { return middleName; }

public void setAddress(String address) { this.address = address; }

public String getAddress() { return address; }

public void setPhone(String phone) { this.phone = phone; }

public String getPhone() { return phone; }

public void setGrades(int[] grades) { this.grades = grades; }

public int[] getGrades() { return grades; }

// Вспомогательный метод для подсчета среднего балла

public double getAverageGrade() {

return Arrays.stream(grades).average().orElse(0.0);

}

// Переопределение метода toString

@Override

public String toString() {

return "Abiturient{" +

"id=" + id +

", lastName='" + lastName + '\'' +

", firstName='" + firstName + '\'' +

", middleName='" + middleName + '\'' +

", address='" + address + '\'' +

", phone='" + phone + '\'' +

", grades=" + Arrays.toString(grades) +

", average=" + String.format("%.2f", getAverageGrade()) +

'}';

}

// Методы для работы с массивом объектов

public static void printAbiturientsWithBadGrades(Abiturient[] abiturients, int minPassingGrade) {

for (Abiturient abiturient : abiturients) {

for (int grade : abiturient.getGrades()) {

if (grade < minPassingGrade) {

System.out.println(abiturient);

break;

}

}

}

}

public static void printAbiturientsWithAverageAbove(Abiturient[] abiturients, double average) {

for (Abiturient abiturient : abiturients) {

if (abiturient.getAverageGrade() > average) {

System.out.println(abiturient);

}

}

}

public static void printTopNAbiturients(Abiturient[] abiturients, int n) {

Arrays.sort(abiturients, Comparator.comparingDouble(Abiturient::getAverageGrade).reversed());

for (int i = 0; i < Math.min(n, abiturients.length); i++) {

System.out.println(abiturients[i]);

}

}

public static void main(String[] args) {

// Создание массива объектов класса Abiturient

Abiturient[] abiturients = {

new Abiturient(1, "Иванов", "Иван", "Иванович", "г. Москва", "287273", new int[]{3, 4, 5,5,}),

new Abiturient(2, "Петров", "Петр", "Петрович", "г. Москва", "459267", new int[]{3,4,4,4,3}),

new Abiturient(3, "Иванов", "Николай", "Иванович", "г. Москва", "273829",new int[]{5,5,4,5,4})

};

// Вывод абитуриентов с неудовлетворительными оценками

System.out.println("Абитуриенты с неудовлетворительными оценками:");

printAbiturientsWithBadGrades(abiturients, 3);

// Вывод абитуриентов средний балл у которых выше заданного

System.out.println("\nАбитуриенты со средним баллом выше 4.5:");

printAbiturientsWithAverageAbove(abiturients, 4.5);

// Выбор и вывод n абитуриентов с самым высоким средним баллом

System.out.println("\nЛучшие 2 абитуриента по среднему баллу:");

printTopNAbiturients(abiturients, 2);

}

}

Работа программы показана на рисунке 4.

Изображение выглядит как снимок экрана, Шрифт, текст

Автоматически созданное описание

Рисунок 4 – Работа программы

**Задание 5:** создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString(). Создать объект класса Простая дробь, используя класс Число. Методы: вывод на экран, сложение, вычитание, умножение, деление.

Код класса Main:

package Zadanie\_5;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner console = new Scanner(System.in);

System.out.println("Введите числитель 1: ");

double ch11 = console.nextDouble();

System.out.println("Введите знаменатель 1: ");

double ch12 = console.nextDouble();

System.out.println("Введите числитель 2: ");

double ch21 = console.nextDouble();

System.out.println("Введите знаменатель 2: ");

double ch22 = console.nextDouble();

chislo chislos11 = new chislo(ch11);

chislo chislos12 = new chislo(ch12);

chislo chislos21 = new chislo(ch21);

chislo chislos22 = new chislo(ch22);

eze\_drob eze\_drob1 = new eze\_drob(chislos11, chislos12);

eze\_drob eze\_drob2 = new eze\_drob(chislos21, chislos22);

eze\_drob1.sloj(eze\_drob2);

eze\_drob1.vich(eze\_drob2);

eze\_drob1.del(eze\_drob2);

eze\_drob1.umnoj(eze\_drob2);

}

}

Код класса Chislo:

package Zadanie\_5;

import java.util.Objects;

public class chislo {

private double ch1;

public chislo(double ch1){

this.ch1 = ch1;

}

public double getCh1() {

return ch1;

}

public void setCh1(double ch1) {

this.ch1 = ch1;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

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

return Double.compare(chislo.getCh1(), getCh1()) == 0;

}

@Override

public int hashCode() {

return Objects.hash(getCh1());

}

@Override

public String toString() {

return "chislo{" +

"ch1=" + ch1 +

'}';

}

}

Код класса Eze\_drob:

package Zadanie\_5;

import java.util.Objects;

public class eze\_drob {

private chislo chisl;

private chislo znam;

protected void sloj(eze\_drob eze\_drob2){

double chislitel = chisl.getCh1() \* eze\_drob2.znam.getCh1() + znam.getCh1() \* eze\_drob2.chisl.getCh1();

double znamen = znam.getCh1() \* eze\_drob2.znam.getCh1();

System.out.println("Сложение: ");

System.out.println("Числитель: " + chislitel);

System.out.println("Знаменатель: " + znamen);

}

protected void vich(eze\_drob eze\_drob2){

double chislitel = chisl.getCh1() \* eze\_drob2.znam.getCh1() - znam.getCh1() \* eze\_drob2.chisl.getCh1();

double znamen = znam.getCh1() \* eze\_drob2.znam.getCh1();

System.out.println("Вычитание: ");

System.out.println("Числитель: " + chislitel);

System.out.println("Знаменатель: " + znamen);

}

protected void umnoj(eze\_drob drob2){

double chislitel = chisl.getCh1() \* drob2.chisl.getCh1();

double znamen = znam.getCh1() \* drob2.znam.getCh1();

System.out.println("Умножение: ");

System.out.println("Числитель: " + chislitel);

System.out.println("Знаменатель: " + znamen);

}

protected void del(eze\_drob drob2){

double chislitel = chisl.getCh1() \* drob2.znam.getCh1();

double znamen = znam.getCh1() \* drob2.chisl.getCh1();

System.out.println("Умножение: ");

System.out.println("Числитель: " + chislitel);

System.out.println("Знаменатель: " + znamen);

}

public eze\_drob(chislo chisl, chislo znam) {

this.chisl = chisl;

this.znam = znam;

}

public chislo getChisl() {

return chisl;

}

public void setChisl(chislo chisl) {

this.chisl = chisl;

}

public chislo getZnam() {

return znam;

}

public void setZnam(chislo znam) {

this.znam = znam;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (!(o instanceof eze\_drob eze\_drob)) return false;

return Objects.equals(getChisl(), eze\_drob.getChisl()) && Objects.equals(getZnam(), eze\_drob.getZnam());

}

@Override

public int hashCode() {

return Objects.hash(getChisl(), getZnam());

}

@Override

public String toString() {

return "eze\_drob{" +

"chisl=" + chisl +

", znam=" + znam +

'}';

}

}

Работа программы показана на рисунке 5.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 5 – Работа программы

**Задание 6:** создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString(). Создать объект класса Дом, используя классы Окно, Дверь. Методы: закрыть на ключ, вывести на консоль количество окон, дверей.

Код класса Main:

package Zadanie\_6;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner console = new Scanner(System.in);

System.out.println("сколько дома дверей?");

int n1 = console.nextInt();

door[] doors = new door[n1];

doors[0] = new door(false);

System.out.println("сколько дома окон?");

int n2 = console.nextInt();

window[] windows = new window[n2];

windows[0] = new window(false);

house houses = new house(windows, doors);

doors[0].lock\_on\_key();

if(doors[0].getLock()){

System.out.println("дверь закрыта");

} else {

System.out.println("дверь открыта");

}

windows[0].lock();

if(doors[0].getLock()){

System.out.println("окно закрыто");

} else {

System.out.println("окно открыто");

}

System.out.println("В этом доме столько дверей: " + houses.getDoors().length);

System.out.println("В этом доме столько окон: " + houses.getWindows().length);

}

}

Код класса House:

package Zadanie\_6;

public class house {

private window[] windows = new window[100];

private door[] doors = new door[100];

public house(window[] windows, door[] doors) {

this.windows = windows;

this.doors = doors;

}

public window[] getWindows() {

return windows;

}

public void setWindows(window[] windows) {

this.windows = windows;

}

public door[] getDoors() {

return doors;

}

public void setDoors(door[] doors) {

this.doors = doors;

}

}

Код класса Door:

package Zadanie\_6;

import java.util.Objects;

import java.util.Scanner;

public class door {

Scanner console = new Scanner(System.in);

private boolean lock;

public door(boolean lock) {

this.lock = lock;

}

public boolean lock\_on\_key(){

System.out.println("Что сделать с дверью?");

System.out.println("1 - открыть, 2 - закрыть на ключ");

String doing = console.next();

if (doing.equals("1")){

lock = true;

} else {

lock = false;

}

return lock;

}

public boolean getLock() {

return lock;

}

public void setLock(boolean lock) {

this.lock = lock;

}

@Override

public String toString() {

return "door{" +

"lock=" + lock +

'}';

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

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

return getLock() == door.getLock();

}

@Override

public int hashCode() {

return Objects.hash(getLock());

}

}

Код класса Window:

package Zadanie\_6;

import java.util.Objects;

import java.util.Scanner;

public class window {

Scanner console = new Scanner(System.in);

private boolean lock;

public window(boolean lock) {

this.lock = lock;

}

public boolean lock(){

System.out.println("Что сделать с окном?");

System.out.println("1 - открыть, 2 - закрыть");

String doing = console.next();

if (doing.equals("1")){

lock = true;

} else {

lock = false;

}

return lock;

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

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

return lock == window.lock;

}

@Override

public int hashCode() {

return Objects.hash(lock);

}

@Override

public String toString() {

return "window{" +

"lock=" + lock +

'}';

}

}

Работа программы показана на рисунке 6.

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 6 – Работа программы

**Задание 7:** построить модель программной системы. Система Больница. Пациенту назначается лечащий Врач. Врач может сделать назначение Пациенту (процедуры, лекарства, операции). Медсестра или другой Врач выполняют назначение. Пациент может быть выписан из Больницы по окончании лечения, при нарушении режима или при иных обстоятельствах.

Код класса Main:

package Zadanie\_7;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner console = new Scanner(System.in);

Pacient pacients1 = new Pacient( "Роман", "Тетраков",

"Таблеткович", 1, "Шиза", 1, false, false);

lech\_vrach lechVrach1 = new lech\_vrach (1, "Роман", "Тетраков",

"Таблеткович", "от шизи", "пить таблетки");

med\_sister medSister1 = new med\_sister("Мария", "Кариева",

"Оранжевна", "Тетраков");

Pacient[] pacients = new Pacient[100];

lech\_vrach[] lechVrach = new lech\_vrach[100];

med\_sister[] medSister = new med\_sister[100];

int id\_cart = 2;

int id\_vrach = 2;

int id\_vracha = 0;

int id\_pacienta = 0;

int id\_sister = 0;

System.out.println("С кем будет работа?");

System.out.println("1 - пациент");

System.out.println("2 - врач");

System.out.println("3 - медсестра");

int work = console.nextInt();

if(work == 1){

System.out.println("Пациент");

System.out.println("добавить пациента?");

String otv = console.next();

if(otv.equals("да")){

String first\_name = console.next();

String second\_name = console.next();

String third\_name = console.next();

String lechnie = console.next();

int vrach = console.nextInt();

id\_cart++;

pacients[id\_pacienta] = new Pacient(first\_name, second\_name, third\_name, id\_cart,

lechnie, vrach, false, false);

id\_pacienta++;

}

System.out.println("с каким пациоентом рабоать?");

int id\_cart\_pac = console.nextInt();

System.out.println("Назначить врача?");

String da = console.next();

if(da.equals("да")){

pacients[id\_cart\_pac].setLech\_vrach\_naz(lechVrach[id\_vracha].getId());

}

if(pacients[id\_cart\_pac].getEnd\_lech() || pacients[id\_cart\_pac].getNarush()){

System.out.println("пациента выписан");

System.out.println("Причина");

if(pacients[id\_cart\_pac].getEnd\_lech()){

System.out.println("Закончено лечение");

} else {

System.out.println("Буянит");

}

}

} else if (work == 2) {

System.out.println("Врач");

System.out.println("добавить врача?");

String otv = console.next();

if(otv.equals("да")){

String first\_name = console.next();

String second\_name = console.next();

String third\_name = console.next();

String spec = console.next();

String lechnie = console.next();

id\_vrach++;

lechVrach[id\_vracha] = new lech\_vrach(id\_vrach, first\_name, second\_name, third\_name, spec, lechnie);

id\_vracha++;

}

System.out.println("с каким врачом рабоать?");

int id\_vrach\_work = console.nextInt();

System.out.println("какому пациенту назначать?");

int id\_cart\_pac = console.nextInt();

System.out.println("Назначить процедуры?");

String da = console.next();

if(da.equals("да")){

pacients[id\_cart\_pac].setProc(lechVrach[id\_vrach\_work].procedur());

} else {

System.out.println("Назначить лекарство?");

da = console.next();

if(da.equals("да")){

pacients[id\_cart\_pac].setProc(lechVrach[id\_vrach\_work].new\_lech());

} else {

System.out.println("Назначить операцию?");

da = console.next();

if(da.equals("да")){

pacients[id\_cart\_pac].setProc(lechVrach[id\_vrach\_work].operasion());

}

}

}

} else {

System.out.println("Медсестра");

System.out.println("добавить медсестру?");

String otv = console.next();

if(otv.equals("да")){

String first\_name = console.next();

String second\_name = console.next();

String third\_name = console.next();

String boss = console.next();

medSister[id\_sister] = new med\_sister(first\_name, second\_name, third\_name, boss);

id\_sister++;

}

System.out.println("с какой медсестрой рабоать?");

int id\_med\_pac = console.nextInt();

medSister[id\_med\_pac].naznach();

}

}

}

Код класса Lech\_vrach:

package Zadanie\_7;

import java.util.Scanner;

public class lech\_vrach {

Scanner console = new Scanner(System.in);

public int id;

public String first\_name;

public String second\_name;

public String third\_name;

public String specialnost;

public String lechenie;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String new\_lech(){

System.out.println("Новое лечение");

String lech = console.next();

return lech;

}

public String operasion(){

System.out.println("Какую операцию назначить: ");

String op = console.next();

return op;

}

public String procedur(){

System.out.println("Какую процедуру назначить: ");

String pr = console.next();

return pr;

}

public lech\_vrach(int id, String first\_name, String second\_name, String third\_name,

String specialnost, String lechenie) {

this.id = id;

this.first\_name = first\_name;

this.second\_name = second\_name;

this.third\_name = third\_name;

this.specialnost = specialnost;

this.lechenie = lechenie;

}

public String getFirst\_name() {

return first\_name;

}

public void setFirst\_name(String first\_name) {

this.first\_name = first\_name;

}

public String getSecond\_name() {

return second\_name;

}

public void setSecond\_name(String second\_name) {

this.second\_name = second\_name;

}

public String getThird\_name() {

return third\_name;

}

public void setThird\_name(String third\_name) {

this.third\_name = third\_name;

}

public String getSpecialnost() {

return specialnost;

}

public void setSpecialnost(String specialnost) {

this.specialnost = specialnost;

}

public String getLechenie() {

return lechenie;

}

public void setLechenie(String lechenie) {

this.lechenie = lechenie;

}

}

Код класса Med\_sister:

package Zadanie\_7;

import java.util.Scanner;

public class med\_sister {

Scanner console = new Scanner(System.in);

public String first\_name;

public String second\_name;

public String third\_name;

public String boss;

public med\_sister(String first\_name, String second\_name, String third\_name, String boss) {

this.first\_name = first\_name;

this.second\_name = second\_name;

this.third\_name = third\_name;

this.boss = boss;

}

public void naznach(){

System.out.println("выполнить назначение");

}

public String getFirst\_name() {

return first\_name;

}

public void setFirst\_name(String first\_name) {

this.first\_name = first\_name;

}

public String getSecond\_name() {

return second\_name;

}

public void setSecond\_name(String second\_name) {

this.second\_name = second\_name;

}

public String getThird\_name() {

return third\_name;

}

public void setThird\_name(String third\_name) {

this.third\_name = third\_name;

}

public String getBoss() {

return boss;

}

public void setBoss(String boss) {

this.boss = boss;

}

}

Код класса Pacient:

package Zadanie\_7;

public class Pacient {

public String first\_name;

public String second\_name;

public String third\_name;

public int id\_medic\_cart;

public String lechenie;

public int vrach;

public boolean end\_lech;

public boolean narush;

public String oper;

public String proc;

public int lechushii;

public int lech\_vrach\_naz;

public boolean getEnd\_lech() {

return end\_lech;

}

public boolean getNarush() {

return narush;

}

public void setNarush(boolean narush) {

this.narush = narush;

}

public void setEnd\_lech(boolean end\_lech) {

this.end\_lech = end\_lech;

}

public int getLech\_vrach\_naz() {

return lech\_vrach\_naz;

}

public void setLech\_vrach\_naz(int lech\_vrach\_naz) {

this.lech\_vrach\_naz = lech\_vrach\_naz;

}

public int getLechushii() {

return lechushii;

}

public void setLechushii(int lechushii) {

this.lechushii = lechushii;

}

public String getOper() {

return oper;

}

public void setOper(String oper) {

this.oper = oper;

}

public String getProc() {

return proc;

}

public void setProc(String proc) {

this.proc = proc;

}

public Pacient(String first\_name, String second\_name, String third\_name,

int id\_medic\_cart, String lechenie, int vrach, boolean end\_lech, boolean narush) {

this.first\_name = first\_name;

this.second\_name = second\_name;

this.third\_name = third\_name;

this.id\_medic\_cart = id\_medic\_cart;

this.lechenie = lechenie;

this.vrach = vrach;

this.end\_lech = end\_lech;

this.narush = narush;

}

public String getFirst\_name() {

return first\_name;

}

public void setFirst\_name(String first\_name) {

this.first\_name = first\_name;

}

public String getSecond\_name() {

return second\_name;

}

public void setSecond\_name(String second\_name) {

this.second\_name = second\_name;

}

public String getThird\_name() {

return third\_name;

}

public void setThird\_name(String third\_name) {

this.third\_name = third\_name;

}

public int getId\_medic\_cart() {

return id\_medic\_cart;

}

public void setId\_medic\_cart(int id\_medic\_cart) {

this.id\_medic\_cart = id\_medic\_cart;

}

public String getLechenie() {

return lechenie;

}

public void setLechenie(String lechenie) {

this.lechenie = lechenie;

}

}

Работа программы показана на рисунке 7.

Изображение выглядит как снимок экрана, Мультимедийное программное обеспечение, программное обеспечение, текст

Автоматически созданное описание

Рисунок 7 – Работа программы

**Задание 8:** построить модель программной системы. Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает Экзамены. Преподаватель выставляет Оценку. Система подсчитывает средний балл и определяет Абитуриентов, зачисленных в учебное заведение.

Код класса Main:

package Zadanie\_8;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner console = new Scanner(System.in);

abiturient[] abiturients = new abiturient[100];

facultet[] facultets = new facultet[10];

Prepodovatel[] prepodovatels = new Prepodovatel[40];

System.out.println("Сколько будет аббитуриентов: ");

int col1 = console.nextInt();

for (int i = 0; i < col1; i++) {

System.out.println("Введите имя: ");

String first\_name = console.next();

System.out.println("Введите фамилию: ");

String second\_name = console.next();

System.out.println("Введите отчество: ");

String third\_name = console.next();

abiturients[i] = new abiturient(i, first\_name, second\_name, third\_name);

}

System.out.println("Сколько будет учителей: ");

int col2 = console.nextInt();

for (int i = 0; i < col2; i++) {

String first\_name = console.next();

String second\_name = console.next();

String third\_name = console.next();

prepodovatels[i] = new Prepodovatel(i, first\_name, second\_name, third\_name);

}

System.out.println("с каким аббитуриентом работать?");

int id\_abb = console.nextInt();

System.out.println("на какой факультет поступает?");

int fak = console.nextInt();

abiturients[id\_abb].setFak(fak);

int n = 0;

System.out.println("с каким преподавателем работать?");

int id\_pr = console.nextInt();

if(abiturients[id\_abb].ekz()){

facultets[abiturients[id\_abb].fak].sr += prepodovatels[id\_pr].ocenka();

abiturients[id\_abb].setOcenka(prepodovatels[id\_pr].ocenka());

n++;

facultets[abiturients[id\_abb].fak].setN(n);

}

System.out.println("Средний бал какого факультета?");

int num\_fak = console.nextInt();

System.out.println(facultets[num\_fak].pr\_ball());

if (abiturients[id\_abb].getOcenka() > facultets[num\_fak].pr\_ball()){

System.out.println("аббитуриент прошел");

} else {

System.out.println("аббитуриент не прошел");

}

}

}

Код класса Abiturient:

package Zadanie\_8;

import java.util.Scanner;

public class abiturient {

Scanner console = new Scanner(System.in);

public int id;

public String first\_name;

public String second\_name;

public String third\_name;

public int fak;

public int ocenka;

public int getOcenka() {

return ocenka;

}

public void setOcenka(int ocenka) {

this.ocenka = ocenka;

}

public boolean ekz(){

boolean sdal;

System.out.println("Экзамен написан?");

String da = console.next();

if(da.equals("да")){

sdal = true;

} else {

sdal = false;

}

return sdal;

}

public int getFak() {

return fak;

}

public void setFak(int fak) {

this.fak = fak;

}

public abiturient(int id, String first\_name, String second\_name, String third\_name) {

this.id = id;

this.first\_name = first\_name;

this.second\_name = second\_name;

this.third\_name = third\_name;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getFirst\_name() {

return first\_name;

}

public void setFirst\_name(String first\_name) {

this.first\_name = first\_name;

}

public String getSecond\_name() {

return second\_name;

}

public void setSecond\_name(String second\_name) {

this.second\_name = second\_name;

}

public String getThird\_name() {

return third\_name;

}

public void setThird\_name(String third\_name) {

this.third\_name = third\_name;

}

}

Код класса Facultet:

package Zadanie\_8;

public class facultet {

public int id;

public double sr;

public int n;

public String name;

public int getN() {

return n;

}

public void setN(int n) {

this.n = n;

}

public double pr\_ball(){

double ball = 0;

ball = sr / n;

return ball;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public double getSr() {

return sr;

}

public void setSr(double sr) {

this.sr = sr;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

Код класса Prepodovatel:

package Zadanie\_8;

public class Prepodovatel {

public int id;

public String first\_name;

public String second\_name;

public String third\_name;

public int ocenka(){

int n2 = 5;

int n1 = 2;

int oc = (int) ((Math.random() \* ((n2 - n1))) + n1);

return oc;

}

public Prepodovatel(int id, String first\_name, String second\_name, String third\_name) {

this.id = id;

this.first\_name = first\_name;

this.second\_name = second\_name;

this.third\_name = third\_name;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getFirst\_name() {

return first\_name;

}

public void setFirst\_name(String first\_name) {

this.first\_name = first\_name;

}

public String getSecond\_name() {

return second\_name;

}

public void setSecond\_name(String second\_name) {

this.second\_name = second\_name;

}

public String getThird\_name() {

return third\_name;

}

public void setThird\_name(String third\_name) {

this.third\_name = third\_name;

}

}

Работа программы показана на рисунке 8.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рисунок 8 – Работа программы

**Вывод:** были освоены принципы ООП на языке программирования Java