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.**

О Т Ч Е Т

по лабораторной работе № 3

Вариант 14

Название: Арифметические операции

Дисциплина: Языки программирования для работы с большими данными

Студент	ИУ6-23М		В.Е. Санталов
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

Цель работы: получение навыков работы с классами Java, исследование механизмов наследования и полиморфизма.

Задание 1:

4. 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

$$||a||_1 = \max_{1 \le i \le n} \sum_{j=1}^{n} (a_{ij}), \|a\|_2 = \max_{1 \le j \le n} \sum_{i=1}^{n} (a_{ij})$$

Определить, какая из матриц имеет наименьшую первую и вторую нормы.

5. Определить класс Матрица размерности (m x n). Класс должен содержать несколько конструкторов. Объявить массив объектов. Передать объекты в метод, меняющий местами строки с максимальным и минимальным элементами k-го столбца. Создать метод, который изменяет i-ю матрицу путем возведения ее в квадрат.

Подзадача 1.

Код программы:

```java
import java.util.Arrays;
import java.util.Random;

public class Task1_4 {

    public static class Matrix {
        private Integer[][] value;

        public Matrix(Integer[][] value) {
            this.value = value;
        }

        public Matrix(int size, int bound, int bias) {
            Integer[][] value = new Integer[size][size];
            Random rand = new Random();

            for (int i = 0; i < size; i++) {
                for (int j = 0; j < size; j++) {
                    value[i][j] = rand.nextInt(bound) + bias;
                }
            }

            this.value = value;
        }

        public Integer[][] getValue() {
            return this.value;
        }

        public void add(Matrix matrix) {
            int size = matrix.getValue().length;

            if (size == this.value.length) {
                for (int i = 0; i < size; i++) {
                    for (int j = 0; j < size; j++) {
                        this.value[i][j] += matrix.getValue()[i][j];
                    }
                }
            }
            else System.out.println(x: "Matrix sizes are not equal");
        }

        public void sub(Matrix matrix) {
            int size = matrix.getValue().length;
```

```
        if (size == this.value.length) {
            for (int i = 0; i < size; i++) {
                for (int j = 0; j < size; j++) {
                    this.value[i][j] -= matrix.getValue()[i][j];
                }
            }
        }
        else System.out.println(x: "Matrix sizes are not equal");
    }

    public void mul(Matrix matrix) {
        int size = matrix.getValue().length;
        Integer[][] newMatrix = new Integer[size][size];

        if (size == this.value.length) {
            for (int i = 0; i < size; i++) {
                for (int j = 0; j < size; j++) {
                    int sum = 0;
                    for (int k = 0; k < matrix.getValue().length; k++) {
                        sum += this.value[i][k] * matrix.getValue()[k][j];
                    }
                    newMatrix[i][j] = sum;
                }
            }
        }
        else System.out.println(x: "Matrix sizes are not equal");

        this.value = newMatrix;
    }

    public Integer findNorm(int normNumber) {
        int size = this.value.length;

        Integer[] normArray = new Integer[size];

        for (int i = 0; i < size; i++) {
            int sum = 0;
            for (int j = 0; j < size; j++) {
                sum += this.value[i][j];
            }
            normArray[i] = sum;
        }

        Arrays.sort(normArray);
        System.out.println(normArray);
```

```
        Arrays.sort(normArray);
        System.out.println(normArray);

        if (normNumber < size) return normArray[size - normNumber - 1];
        else {
            System.out.println(x: "Not correct norm number");
            return null;
        }
    }

    public static Matrix maxNorm(int normNumber, Matrix matrix1, Matrix matrix2) {
        int m1Norm = matrix1.findNorm(normNumber);
        int m2Norm = matrix2.findNorm(normNumber);

        return m1Norm > m2Norm ? matrix1 : matrix2;
    }
}

Run | Debug
public static void main(String[] args) {
    Matrix[] matrixes = new Matrix[2];

    matrixes[0] = new Matrix(size: 8, bound: 11, bias: 6);
    matrixes[1] = new Matrix(size: 8, bound: 11, bias: 6);

    Arrays.stream(matrixes[0].getValue()).map(Arrays::toString).forEach(System.out::println);
    Arrays.stream(matrixes[1].getValue()).map(Arrays::toString).forEach(System.out::println);

    System.out.println(matrixes[0].findNorm(normNumber: 0));
    System.out.println(matrixes[1].findNorm(normNumber: 0));
    System.out.println(Matrix.maxNorm(normNumber: 0, matrixes[0], matrixes[2]));
}
```

Процесс работы программы:

```
[6, 16, 14, 14, 7, 7, 13, 11]
[10, 10, 10, 7, 16, 14, 12, 10]
[7, 7, 15, 7, 16, 9, 9, 8]
[10, 10, 10, 6, 12, 9, 8, 8]
[14, 12, 6, 7, 12, 13, 6, 6]
[10, 16, 12, 15, 16, 8, 13, 15]
[13, 15, 11, 13, 12, 7, 6, 6]
[6, 11, 6, 9, 9, 8, 7, 8]

[11, 6, 14, 11, 15, 9, 13, 7]
[8, 11, 8, 9, 14, 15, 8, 14]
[13, 12, 7, 8, 9, 8, 14, 10]
[13, 14, 6, 12, 8, 6, 9, 10]
[12, 13, 12, 15, 15, 7, 11, 10]
[10, 10, 7, 11, 10, 16, 7, 10]
[6, 9, 10, 8, 9, 12, 6, 7]
[10, 14, 8, 16, 10, 10, 15, 13]
[Ljava.lang.Integer;@681a9515
105
[Ljava.lang.Integer;@3af49f1c
96
```

Подзадача 2.

Код программы:

```
import java.util.Arrays;
import java.util.Random;

public class Task1_5 {

    public static class Matrix {
        public Integer[][] value;

        public Matrix(Integer[][] value) {
            this.value = value;
        }

        public Matrix(int n, int m, int bound, int bias) {
            Integer[][] value = new Integer[n][m];
            Random rand = new Random();

            for (int i = 0; i < n; i++) {
                for (int j = 0; j < m; j++) {
                    value[i][j] = rand.nextInt(bound) + bias;
                }
            }

            this.value = value;
        }

        static public Matrix swapMaxRows(Matrix matrix, int column) {
            int maxElem = -1000;
            int minElem = 1000;

            int maxElemRow = 0;
            int minElemRow = 0;

            for (int i = 0; i < matrix.value.length; i++) {
                int val = matrix.value[column][i];

                if (val > maxElem) {
                    maxElem = val;
                    maxElemRow = i;
                }

                if (val < minElem) {
                    minElem = val;
```

```
                minElemRow = i;
            }
        }

        Integer[] maxRow = matrix.value[maxElemRow];
        matrix.value[maxElemRow] = matrix.value[minElemRow];
        matrix.value[minElemRow] = maxRow;

        return matrix;
    }

    public void sqr() {
        int n = this.value.length;
        int m = this.value[0].length;

        Integer[][] newMatrix = new Integer[n][m];

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                int sum = 0;
                for (int k = 0; k < n; k++) {
                    sum += this.value[i][k] * this.value[k][j];
                }
                newMatrix[i][j] = sum;
            }
        }

        this.value = newMatrix;
    }

    public static void sqrElem(Matrix[] matrixes, int elem) {
        matrixes[elem].sqr();
    }
}
```

```java
Run | Debug
public static void main(String[] args) {
    Matrix[] matrixes = new Matrix[2];

    matrixes[0] = new Matrix(n: 5, m: 6, bound: 11, -6);
    matrixes[1] = new Matrix(n: 5, m: 6, bound: 11, -6);

    System.out.println(x: "Swapping rows:");
    Arrays.stream(matrixes[0].value).map(Arrays::toString).forEach(System.out::println);
    System.out.println();
    Matrix.swapMaxRows(matrixes[0], column: 3);
    Arrays.stream(matrixes[0].value).map(Arrays::toString).forEach(System.out::println);

    System.out.println(x: "Sqr matrix:");
    Arrays.stream(matrixes[1].value).map(Arrays::toString).forEach(System.out::println);
    System.out.println();
    Matrix.sqrElem(matrixes, elem: 1);
    Arrays.stream(matrixes[1].value).map(Arrays::toString).forEach(System.out::println);
}
```

Процесс работы программы:

```
Swapping rows:
[-1, -3, -4, 2, -6, -4]
[3, 4, 4, 1, -2, -5]
[-4, -2, -1, 0, 0, 4]
[-5, 4, 0, -1, -6, -6]
[-2, -3, -4, -5, 3, -3]

[-1, -3, -4, 2, -6, -4]
[-2, -3, -4, -5, 3, -3]
[-4, -2, -1, 0, 0, 4]
[-5, 4, 0, -1, -6, -6]
[3, 4, 4, 1, -2, -5]
Sqr matrix:
[-4, 4, 3, 4, 2, -6]
[-3, -2, 3, 4, 3, 1]
[-3, 4, 4, 0, -4, 4]
[3, -6, 2, -3, 3, -4]
[-6, 4, -6, -1, -2, 4]

[-5, -28, 8, -14, 0, 32]
[3, -8, -13, -35, -18, 24]
[12, -20, 43, 8, -2, 22]
[-27, 62, -25, -6, -35, 8]
[39, -58, -20, -3, 25, 12]
```

Задание 2:

4. Abiturient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки. Создать массив объектов. Вывести: а) 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).

5. Book: id, Название, Автор(ы), Издательство, Год издания, Количество страниц, Цена, Переплет. Создать массив объектов. Вывести: a) список книг заданного автора; b) список книг, выпущенных заданным издательством; c) список книг, выпущенных после заданного года.

Подзадача 1.

Код программы:

```java
import java.util.Objects;

public class Abiturient {
    int id;
    String name;
    String surname;
    String address;
    String phone;
    int[] scores;

    public Abiturient(int id, String name, String surname, String address, String phone, int[] scores) {
        this.id = id;
        this.name = name;
        this.surname = surname;
        this.address = address;
        this.phone = phone;
        this.scores = scores;
    }

    public int getId() {
        return this.id;
    }


    public String getName() {
        return this.name;
    }


    public String getSurname() {
        return this.surname;
    }


    public String getAddress() {
        return this.address;
    }


    public String getPhone() {
        return this.phone;
    }
```

```java
public int[] getScores() {
    return this.scores;
}

public void setId(int id) {
    this.id = id;
}
public void setName(String name) {
    this.name = name;
}
public void setSurname(String surname) {
    this.surname = surname;
}
public void setAddress(String address) {
    this.address = address;
}
public void setPhone(String phone) {
    this.phone = phone;
}
public void setScores(int[] scores) {
    this.scores = scores;
}

@Override
public boolean equals(Object o) {
    if (o == this)
        return true;
    if (!(o instanceof Abiturient)) {
        return false;
    }
    Abiturient abiturient = (Abiturient) o;
    return id == abiturient.id && Objects.equals(name, abiturient.name) && Objects.equals(surname, abiturient.surname) && Objects.equals(address, abiturient.address) && Obj
}

@Override
public int hashCode() {
    return Objects.hash(id, name, surname, address, phone, scores);
}
```

```java
@Override
public String toString() {
    return "{" +
            " id='" + getId() + "'" +
            ", name='" + getName() + "'" +
            ", surname='" + getSurname() + "'" +
            ", address='" + getAddress() + "'" +
            ", phone='" + getPhone() + "'" +
            ", scores='" + getScores() + "'" +
            "}";
}
```

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.Scanner;

class Lr3_3 {
    static int GIVEN_SCORE = 3;

    Run | Debug
    public static void main(String[] args) {
        ArrayList<Abiturient> abiturients = new ArrayList<Abiturient>();

        abiturients.add(new Abiturient(0, "Vadim",  "Santalov",  "address",  "1", new int[]{ 1, 2, 3, 4, 5 }));
        abiturients.add(new Abiturient(1, "Vyacheslav",  "Eliseev",  "address",  "2", new int[]{ 5, 5, 5, 5, 6 }));
        abiturients.add(new Abiturient(2, "Viktor",  "Korneplod",  "address",  "3", new int[]{ 3, 4, 5, 2, 5 }));
        abiturients.add(new Abiturient(3, "Elon",  "Musk",  "address",  "4", new int[]{ 2, 2, 2, 2, 5 }));

        // A
        System.out.println(x: "Task A:");
        for (Abiturient abiturient : abiturients) {
            Boolean hasNegative = false;

            for (int score : abiturient.scores) {
                if (score == 2) {
                    hasNegative = true;
                    break;
                }
            }

            if (hasNegative) System.out.println(abiturient.name + " " + abiturient.surname + " has bad scores.");
        }

        // B
        System.out.println(x: "\nTask B:");
        for (Abiturient abiturient : abiturients) {
            int scoreSum = 0;

            for (int score : abiturient.scores) scoreSum += score;

            if (scoreSum / abiturient.scores.length > GIVEN_SCORE) System.out.println(abiturient.name + " " + abiturient.surname + " more than average scores.");
        }
```

```java
        // C
        System.out.println(x: "\nTask C:");

        Scanner in = new Scanner(System.in);
        System.out.print("Input number of students (it must be less than " + (abiturients.size() + 1) + "): ");
        int count = in.nextInt();
        in.close();

        Collections.sort(abiturients, new Comparator<Abiturient>() {
            public int compare(Abiturient a1, Abiturient a2) {
                double sum1 = 0;
                double sum2 = 0;

                for (int score : a1.scores) sum1 += score;
                for (int score : a2.scores) sum2 += score;

                sum1 /= a1.scores.length;
                sum2 /= a2.scores.length;

                if (sum1 == sum2) return 0;
                return sum1 > sum2 ? -1 : 1;
            }
        });

        for (int i = 0; i < count; i++) {
            Abiturient ab = abiturients.get(i);

            double sum = 0;
            for (int score : ab.scores) sum += score;
            sum /= ab.scores.length;

            System.out.println("Abiturient " + ab.name + " " + ab.surname + " has AVG score " + sum);
        }
    }
}
```

Процесс работы программы:

```
Task A:
Vadim Santalov has bad scores.
Viktor Korneplod has bad scores.
Elon Musk has bad scores.

Task B:
Vyacheslav Eliseev more than average scores.

Task C:
Input number of students (it must be less than 5): 4
Abiturient Vyacheslav Eliseev has AVG score 5.2
Abiturient Viktor Korneplod has AVG score 3.8
Abiturient Vadim Santalov has AVG score 3.0
Abiturient Elon Musk has AVG score 2.6
```

Подзадача 2.

Код программы:

```java
import java.util.Objects;

public class Book {
    int id;
    String name;
    String author;
    int year;
    int pageCount;
    int price;
    String cover;

    public Book(int id, String name, String author, int year, int pageCount, int price, String cover) {
        this.id = id;
        this.name = name;
        this.author = author;
        this.year = year;
        this.pageCount = pageCount;
        this.price = price;
        this.cover = cover;
    }

    public int getId() {
        return this.id;
    }

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

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAuthor() {
        return this.author;
    }
```

```java
    public void setAuthor(String author) {
        this.author = author;
    }

    public int getYear() {
        return this.year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public int getPageCount() {
        return this.pageCount;
    }

    public void setPageCount(int pageCount) {
        this.pageCount = pageCount;
    }

    public int getPrice() {
        return this.price;
    }

    public void setPrice(int price) {
        this.price = price;
    }

    public String getCover() {
        return this.cover;
    }

    public void setCover(String cover) {
        this.cover = cover;
    }
```

```java
    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof Book)) {
            return false;
        }
        Book book = (Book) o;
        return id == book.id && Objects.equals(name, book.name) && Objects.equals(author, book.author) && year == book.year && pageCount == book.pageCount && price == book.pric
    }

    @Override
    public int hashCode() {
        return Objects.hash(id, name, author, year, pageCount, price, cover);
    }

    @Override
    public String toString() {
        return "{" +
            " id='" + getId() + "'" +
            ", name='" + getName() + "'" +
            ", author='" + getAuthor() + "'" +
            ", year='" + getYear() + "'" +
            ", pageCount='" + getPageCount() + "'" +
            ", price='" + getPrice() + "'" +
            ", cover='" + getCover() + "'" +
            "}";
    }
}
```

```java
import java.util.ArrayList;
import java.util.Scanner;

public class Lr3_4 {
    Run | Debug
    public static void main(String[] args) {
        ArrayList<Book> books = new ArrayList<Book>();

        books.add(new Book(id: 0, name: "Yes", author: "Me", year: 1999, pageCount: 9000, price: 100, cover: "Has one"));
        books.add(new Book(id: 1, name: "No", author: "Elon Musk", year: 2000, pageCount: 917, price: 100, cover: "Skin"));
        books.add(new Book(id: 2, name: "Maybe", author: "Joe Biden", year: 2021, pageCount: 1235, price: 100, cover: "Carton"));
        books.add(new Book(id: 3, name: "50 shades of green", author: "Vladimir Putin", year: 2077, pageCount: 512, price: 100, cover: "Grass"));

        // A
        Scanner in = new Scanner(System.in);
        System.out.print(s: "Enter author's name: ");
        String author = in.nextLine();

        for (Book book : books) {
            if (book.author.equals(author)) System.out.println(book.name);
        }

        // B
        // Никакого издательства в задании нет, поэтому просто сделал обложку
        System.out.print(s: "Enter cover: ");
        String cover = in.nextLine();

        for (Book book : books) {
            if (book.cover.equals(cover)) System.out.println(book.name);
        }

        // C
        System.out.print(s: "Enter year: ");
        int year = in.nextInt();

        for (Book book : books) {
            if (book.year > year) System.out.println(book.name);
        }

        in.close();
    }
}
```

Процесс работы программы:

```
Enter author's name: Elon Musk
No
Enter cover: Skin
No
Enter year: 2021
50 shades of green
```

Задание 3:

4. Создать объект класса Простая дробь, используя класс Число. Методы: вывод на экран, сложение, вычитание, умножение, деление.

5. Создать объект класса Дом, используя классы Окно, Дверь. Методы: закрыть на ключ, вывести на консоль количество окон, дверей.

Подзадача 1.

Код программы:

```java
import java.util.Objects;

public class MNumber {
    private int value;

    public MNumber(int value) {
        this.value = value;
    }

    public int getValue() {
        return this.value;
    }

    public void setValue(int value) {
        this.value = value;
    }

    public MNumber value(int value) {
        setValue(value);
        return this;
    }

    public void print() {
        System.out.println(this.value);
    }

    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof MNumber)) {
            return false;
        }
        MNumber MNumber = (MNumber) o;
        return value == MNumber.value;
    }

    @Override
    public int hashCode() {
        return Objects.hashCode(value);
    }

    @Override
    public String toString() {
        return "{" +
            " value='" + getValue() + "'" +
            "}";
    }
```

```java
import java.util.Objects;

public class Fractional {
    private MNumber denominator;
    private MNumber numerator;

    Fractional(MNumber numerator, MNumber denominator) {
        this.numerator = numerator;
        this.denominator = denominator;
    }

    public double getValue() {
        return numerator.getValue() / denominator.getValue();
    }

    public double add(Fractional frac) {
        return this.getValue() + frac.getValue();
    }

    public double sub(Fractional frac) {
        return this.getValue() - frac.getValue();
    }

    public double mul(Fractional frac) {
        return this.getValue() * frac.getValue();
    }

    public double div(Fractional frac) {
        return this.getValue() - frac.getValue();
    }

    public MNumber getDenominator() {
        return this.denominator;
    }

    public void setDenominator(MNumber denominator) {
        this.denominator = denominator;
    }

    public MNumber getNumerator() {
        return this.numerator;
    }

    public void setNumerator(MNumber numerator) {
        this.numerator = numerator;
    }
```

```java
    public Fractional denominator(MNumber denominator) {
        setDenominator(denominator);
        return this;
    }

    public Fractional numerator(MNumber numerator) {
        setNumerator(numerator);
        return this;
    }

    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof Fractional)) {
            return false;
        }
        Fractional fractional = (Fractional) o;
        return Objects.equals(denominator, fractional.denominator) && Objects.equals(numerator, fractional.numerator);
    }

    @Override
    public int hashCode() {
        return Objects.hash(denominator, numerator);
    }

    @Override
    public String toString() {
        return "{" +
            " denominator='" + getDenominator() + "'" +
            ", numerator='" + getNumerator() + "'" +
            "}";
    }

}
```

Подзадача 2.

Код программы:

```java
import java.util.Objects;

public class Door {
    enum State {
        CLOSED,
        OPENED
    }

    State state;

    public Door(State state) {
        this.state = state;
    }

    public void open() {
        this.state = State.OPENED;
    }

    public void close() {
        this.state = State.CLOSED;
    }

    public State getState() {
        return this.state;
    }

    public void setState(State state) {
        this.state = state;
    }

    public Door state(State state) {
        setState(state);
        return this;
    }

    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof Door)) {
            return false;
        }
        Door Door = (Door) o;
        return Objects.equals(state, Door.state);
    }
```

```
    @Override
    public int hashCode() {
        return Objects.hashCode(state);
    }

    @Override
    public String toString() {
        return "{" +
            " state='" + getState() + "'" +
            "}";
    }

}
```

```java
import java.util.Objects;

public class Window {
    enum State {
        CLOSED,
        OPENED
    }

    State state;

    public Window(State state) {
        this.state = state;
    }

    public void open() {
        this.state = State.OPENED;
    }

    public void close() {
        this.state = State.CLOSED;
    }

    public State getState() {
        return this.state;
    }

    public void setState(State state) {
        this.state = state;
    }

    public Window state(State state) {
        setState(state);
        return this;
    }

    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof Window)) {
            return false;
        }
        Window window = (Window) o;
        return Objects.equals(state, window.state);
    }
```

```java
    @Override
    public int hashCode() {
        return Objects.hashCode(state);
    }

    @Override
    public String toString() {
        return "{" +
            " state='" + getState() + "'" +
            "}";
    }

}
```

```java
import java.util.Objects;

public class House {
    private Door[] doors;
    private Window[] windows;

    public House(Door[] doors, Window[] windows) {
        this.doors = doors;
        this.windows = windows;
    }

    public int getDoorCount() {
        return this.doors.length;
    }

    public int getWindowsCount() {
        return this.windows.length;
    }

    public void closeDoors() {
        for (Door door : doors) {
            door.close();
        }
    }

    public void closeWindows() {
        for (Window window : windows) {
            window.close();
        }
    }

    public void closeEverything() {
        this.closeDoors();
        this.closeWindows();
    }

    public Door[] getDoor() {
        return this.doors;
    }

    public void setDoor(Door[] door) {
        this.doors = door;
    }

    public Window[] getWindows() {
        return this.windows;
    }
```

```
    public void setWindows(Window[] windows) {
        this.windows = windows;
    }

    public House door(Door[] door) {
        setDoor(door);
        return this;
    }

    public House windows(Window[] windows) {
        setWindows(windows);
        return this;
    }

    @Override
    public boolean equals(Object o) {
        if (o == this)
            return true;
        if (!(o instanceof House)) {
            return false;
        }
        House house = (House) o;
        return Objects.equals(doors, house.doors) && Objects.equals(windows, house.windows);
    }

    @Override
    public int hashCode() {
        return Objects.hash(doors, windows);
    }

    @Override
    public String toString() {
        return "{" +
            " door='" + getDoor() + "'" +
            ", windows='" + getWindows() + "'" +
            "}";
    }

}
```

Задание 4:

4. Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает Экзамены. Преподаватель выставляет Оценку. Система подсчитывает средний балл и определяет Абитуриентов, зачисленных в учебное заведение.

5. Система Библиотека. Читатель оформляет Заказ на Книгу. Система осуществляет поиск в Каталоге. Библиотекарь выдает Читателю Книгу на

абонемент или в читальный зал. При невозвращении Книги Читателем он может быть занесен Администратором в «черный список».

Подзадача 1.

Код программы:

```java
import java.util.Random;

public class Abiturient extends Person {

    Abiturient(String name, String surname) {
        super(name, surname);
    }

    public Exam writeExam(int questions) {
        Random rand = new Random();

        return new Exam(
            this,
            questions,
            questions - rand.nextInt(questions + 1)
        );
    }
}
```

```java
public class Exam {
    Abiturient abiturient;
    int questions;
    int answers;

    Exam(Abiturient abiturient, int questions, int answers) {
        this.abiturient = abiturient;
        this.questions = questions;
        this.answers = answers;
    }
}
```

```java
import java.util.ArrayList;

public class Faculty {
    private ArrayList<Abiturient> abiturients;
    public Professor professor;

    int passScore;
    int questionsCount;

    Faculty(int passScore, int questionsCount) {
        this.abiturients = new ArrayList<Abiturient>();
        this.professor = new Professor("Galina", "Sergeevna", "Doctor");

        this.passScore = passScore;
        this.questionsCount = questionsCount;
    }

    void register(Abiturient abiturient) {
        this.abiturients.add(abiturient);
    }

    ArrayList<Exam> getExams() {
        ArrayList<Exam> exams = new ArrayList<Exam>();

        for (Abiturient abiturient : this.abiturients) exams.add(abiturient.writeExam(this.questionsCount));

        return exams;
    }

    ArrayList<Abiturient> applyAbiturients() {
        ArrayList<Abiturient> abiturients = new ArrayList<Abiturient>();

        ArrayList<Result> results = this.professor.checkExams(this.getExams());
        for (Result result : results) if (result.score >= passScore) abiturients.add(result.abiturient);

        return abiturients;
    }
}
```

```java
public class Person {
    String name;
    String surname;

    Person(String name, String surname) {
        this.name = name;
        this.surname = name;
    }
}
```

```java
import java.util.ArrayList;
import java.util.Random;

public class Professor extends Person {
    String degree;

    Professor(String name, String surname, String degree) {
        super(name, surname);
        this.degree = degree;
    }

    ArrayList<Result> checkExams(ArrayList<Exam> exams) {
        ArrayList<Result> results = new ArrayList<Result>();
        Random rand = new Random();

        for (Exam exam : exams) results.add(new Result(exam.abiturient, rand.nextDouble(1) * exam.questions));

        return results;
    }
}
```

```java
public class Result {
    public Abiturient abiturient;
    public double score;

    Result(Abiturient abiturient, double score) {
        this.abiturient = abiturient;
        this.score = score;
    }
}
```

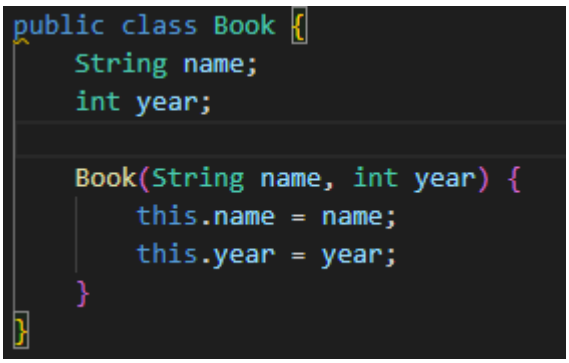
Подзадача 2.

Код программы:

```java
public class Administrator extends Person {

    Administrator(String name, String surname) {
        super(name, surname);
    }

    void banReader(Reader reader, Library library) {
        library.blacklist.add(reader);
    }
}
```


public class Book {
String name;
int year;
Book(String name, int year) {
this.name = name;
this.year = year;
}
}

```java
import java.util.ArrayList;

public class Library {
    ArrayList<Book> books;
    ArrayList<Reader> blacklist;
    ArrayList<Order> orders;
    Administrator administrator;

    int daysToReturn;

    Library(ArrayList<Book> books, ArrayList<Reader> blacklist) {
        this.books = books;
        this.blacklist = blacklist;
        this.orders = new ArrayList<Order>();

        this.administrator = new Administrator("Elon", "Musk");
        this.daysToReturn = 30;
    }

    Book getOrder(Order order) {
        if (books.contains(order.book)) {
            if (!blacklist.contains(order.reader)) {
                this.books.remove(order.book);
                orders.add(order);

                return order.book;
            }
            else {
                System.out.println(x: "This reader was banned from the library;");
                return null;
            }
        }
        else {
            System.out.println(x: "Library don't have such books.");
            return null;
        }
    }

    void checkReturns() {
        LocalDate date = LocalDate.now();

        for (Order order : orders) {
            if (Duration.between(order.date, date).toDays() > daysToReturn) {
                this.administrator.banReader(order.reader, this);
            }
        }
    }
}
```

```java
import java.time.LocalDate;

public class Order {
    Book book;
    Reader reader;
    LocalDate date;

    Order(Book book, Reader reader) {
        this.book = book;
        this.reader = reader;

        this.date = LocalDate.now();
    }
}
```

```java
public class Person {
    String name;
    String surname;

    Person(String name, String surname) {
        this.name = name;
        this.surname = surname;
    }
}
```

```java
public class Reader extends Person {
    String name;
    String surname;

    Reader(String name, String surname) {
        super(name, surname);
    }

    Order performOrder(Book book) {
        return new Order(book, this);
    }
}
```

Ссылка на программное решение:

https://github.com/Time2HackJS/BigDataLanguages/tree/master/lr3

Вывод: в ходе лабораторной работы были получены навыки работы с классами Java, были исследованы механизмы наследования и полиморфизма языка программирования Java.