

МАГИСТЕРСКАЯ ПРОГРАММА 09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных

Вариант 13

Дисциплина: Языки программирования для работы с большими данными

(Подпись, дата)

П.В. Степанов
 (И.О. Фамилия)

Цель лабораторной работы: получение навыков работы с классами и объектами языка программирования Java, а также исследование механизмов наследования и полиморфизма.

Ход работы:

Задание №1:

1. Определить класс Вектор в R^3 . Реализовать методы для проверки векторов на ортогональность, проверки пересечения не ортогональных векторов, сравнения векторов. Создать массив из m объектов. Определить, какие из векторов компланарны.
2. Определить класс Матрица размерности $(n \times n)$. Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения матриц. Объявить массив объектов. Создать методы, вычисляющие первую и вторую нормы матрицы

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

Листинг выполнения подзадачи 1:

Класс MyVectorClass:

```
package laba_3.task_1;

public class MyVectorClass {
    private float x_1;
    private float x_2;
    private float y_1;
    private float y_2;
    private float z_1;
    private float z_2;

    public MyVectorClass() {
    }

    public MyVectorClass(float x_1, float x_2, float y_1, float y_2, float z_1, float
z_2) {
        this.x_1 = x_1;
        this.x_2 = x_2;
        this.y_1 = y_1;
        this.y_2 = y_2;
        this.z_1 = z_1;
        this.z_2 = z_2;
    }

    public float scalarnost(MyVectorClass other){
        return (this.x_2 - this.x_1) * (other.x_2 - other.x_1) + (this.y_2 -
this.y_1) * (other.y_2 - other.y_1) + (this.z_2 - this.z_1) * (other.z_2 -
other.z_1);
    }
}
```

```

    public boolean ortogonalnost(MyVectorClass other){
        //два вектора являются ортогональными (перпендикулярными), если их скалярное
        произведение равно нулю.
        if(scalarnost(other) == 0.){
            return true;
        } else {
            return false;
        }
    }

    public boolean peresechenie(MyVectorClass other, float eps){
        //https://habr.com/ru/post/267037/
        double s = ((other.y_2 - this.y_2)/(this.y_1 - this.y_2) - (other.x_2 -
this.x_2)/(this.x_1 - this.x_2))/((other.x_1 - other.x_2)/(this.x_1 - this.x_2) -
(other.y_1 - other.y_2)/(this.y_1 - this.y_2));
        double t = s*((other.x_1 - other.x_2)/(this.x_1 - this.x_2)) + (other.x_2 -
this.x_2)/(this.x_1 - this.x_2);
        double left = this.z_1 * t + this.z_2 * (1 - t);
        double right = other.z_1 * s + other.z_2 * (1 - s);

        if((left - right) <= eps){
            return true;
        } else {
            return false;
        }
    }

    public String compareVectors(MyVectorClass other){
        double firstModulus = Math.sqrt(Math.pow((this.x_2 - this.x_1), 2) +
Math.pow((this.y_2 - this.y_1), 2) + Math.pow((this.z_2 - this.z_1), 2));
        double secondModulus = Math.sqrt(Math.pow((other.x_2 - other.x_1), 2) +
Math.pow((other.y_2 - other.y_1), 2) + Math.pow((other.z_2 - other.z_1), 2));
        if(firstModulus > secondModulus){
            return "First vector is bigger";
        } else if(firstModulus < secondModulus){
            return "Second vector is bigger";
        } else {
            return "Vectors have the same length";
        }
    }

    public boolean complenarnost(MyVectorClass other, MyVectorClass another) {
        //векторы, которые параллельны одной плоскости или лежат на одной плоскости
        // если смешанное произведение 3-х векторов равно нулю, то эти три вектора
        компланарны.
        float this_x = this.x_2 - this.x_1;
        float this_y = this.y_2 - this.y_1;
        float this_z = this.z_2 - this.z_1;

        float other_x = other.x_2 - other.x_1;
        float other_y = other.y_2 - other.y_1;
        float other_z = other.z_2 - other.z_1;

        float tmpX = this_y * other_z - this_z * other_y;
        float tmpY = -(this_x * other_z - this_z * other_x);
        float tmpZ = this_z * other_y - this_y * other_x;

        float another_x = another.x_2 - another.x_1;
        float another_y = another.y_2 - another.y_1;
        float another_z = another.z_2 - another.z_1;

        float sumOfMuls = tmpX * another_x + tmpY * another_y + tmpZ * another_z;

        if(sumOfMuls == 0.){

```

```

        return true;
    } else {
        return false;
    }
}

@Override
public String toString() {
    return
        "x_1=" + x_1 +
        ", x_2=" + x_2 +
        ", y_1=" + y_1 +
        ", y_2=" + y_2 +
        ", z_1=" + z_1 +
        ", z_2=" + z_2 ;
}
}

```

Применение MyVectorClass в MyVector:

```

package laba_3.task_1;
import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;

public class MyVector {

    public static void main(String[] args) {

        MyVectorClass vec1 = new MyVectorClass(0, 1, 0, 0, 0, 0);
        MyVectorClass vec2 = new MyVectorClass(0, 0, 0, 1, 0, 1);
        MyVectorClass vec3 = new MyVectorClass(0, 1, 0, 1, 0, 1);
        MyVectorClass vec4 = new MyVectorClass(0, 1, 0, -1, 0, 1);
        MyVectorClass vec5 = new MyVectorClass(0, 0, 0, 1, 0, 0);
        MyVectorClass vec6 = new MyVectorClass(0, 0, 0, 2, 0, 0);
        MyVectorClass vec7 = new MyVectorClass(0, 0, 0, 7, 0, 0);

        System.out.println("1 vector: "+vec1);
        System.out.println("2 vector: "+vec2);
        System.out.println("3 vector: "+vec3);
        System.out.println("4 vector: "+vec4);
        System.out.println("5 vector: "+vec5);
        System.out.println("6 vector: "+vec6);
        System.out.println("7 vector: "+vec7);

        System.out.println("Vectors 1 and 2:");
        System.out.println( vec1.ortogonalnost(vec2) ? "Vectors are orthogonal" :
"Vectors aren't orthogonal");
        System.out.println(vec1.compareVectors(vec2));

        System.out.println("Vectors 3 and 4:");
        System.out.println( vec3.ortogonalnost(vec4) ? "Vectors are orthogonal" :
"Vectors aren't orthogonal");
        System.out.println( vec3.peresechenie(vec4, 0.5f) ? "Vectors are crossed" :
"Vectors aren't crossed");
        System.out.println(vec3.compareVectors(vec4));

        System.out.println("Vectors 2 and 3:");
        System.out.println(vec1.complenarnost(vec2, vec3) ? "Vectors are complanar" :
"Vectors aren't complanar");
    }
}

```

```

        System.out.println("Vectors 6 and 7:");
        System.out.println(vec5.complenarnost(vec6, vec7) ? "Vectors are complanar" :
"Vectors aren't complanar");

        //создадим случайный массив для определения кол-ти векторов
        ArrayList<MyVectorClass> vectorArrayList = new ArrayList<>();

        Scanner in = new Scanner(System.in);
        System.out.print("Enter number of vectors: ");
        int s = in.nextInt();
        Random random = new Random();
        float x_1, x_2, y_1, y_2, z_1, z_2;
        float bound_1 = -3.f;
        float bound_2 = 3.f;
        for (int i = 0; i < s; i++) {
            x_1 = random.nextFloat(bound_1, bound_2);
            x_2 = random.nextFloat(bound_1, bound_2);
            y_1 = random.nextFloat(bound_1, bound_2);
            y_2 = random.nextFloat(bound_1, bound_2);
            z_1 = random.nextFloat(bound_1, bound_2);
            z_2 = random.nextFloat(bound_1, bound_2);
            vectorArrayList.add(new MyVectorClass(x_1, x_2, y_1, y_2, z_1, z_2));
        }
        for (int i = 0; i < s-2; i++) {
            for (int j = i+1; j < s-1; j++) {
                for (int k = j+1; k < s; k++) {
                    System.out.println(vectorArrayList.get(i).complenarnost(vectorArrayList.get(j),
vectorArrayList.get(k)) ? "Vectors are complanar" : "Vectors aren't complanar");
                }
            }
        }
    }
}

```

Результат выполнения подзадачи 1:

```
MyVector x
C:\Users\metel\IdeaProjects\my_lab_3\out\production\my_lab_3 laba_3.task_1.MyVector
1 vector: x_1=0.0, x_2=1.0, y_1=0.0, y_2=0.0, z_1=0.0, z_2=0.0
2 vector: x_1=0.0, x_2=0.0, y_1=0.0, y_2=1.0, z_1=0.0, z_2=1.0
3 vector: x_1=0.0, x_2=1.0, y_1=0.0, y_2=1.0, z_1=0.0, z_2=1.0
4 vector: x_1=0.0, x_2=1.0, y_1=0.0, y_2=-1.0, z_1=0.0, z_2=1.0
5 vector: x_1=0.0, x_2=0.0, y_1=0.0, y_2=1.0, z_1=0.0, z_2=0.0
6 vector: x_1=0.0, x_2=0.0, y_1=0.0, y_2=2.0, z_1=0.0, z_2=0.0
7 vector: x_1=0.0, x_2=0.0, y_1=0.0, y_2=7.0, z_1=0.0, z_2=0.0
Vectors 1 and 2:
Vectors are orthogonal
Second vector is bigger
Vectors 3 and 4:
Vectors aren't orthogonal
Vectors are crossed
Vectors have the same length
Vectors 2 and 3:
Vectors aren't complanar
Vectors 6 and 7:
Vectors are complanar
Enter number of vectors: 3
Vectors aren't complanar

Process finished with exit code 0
```

Листинг выполнения подзадачи 2:

Класс SquareMatrix:

```
package laba_3.task_1;

public class SquareMatrix {
    private int nDim;
    private int[][] matrix;
    public SquareMatrix() {
        nDim = 1;
        matrix = new int[1][1];
        matrix[0][0] = 0;
    }
    public SquareMatrix(int n) {
        nDim = n;
        matrix = new int[nDim][nDim];
        for (int i = 0; i < nDim; i++) {
            for (int j = 0; j < nDim; j++) {
                matrix[i][j] = 0;
            }
        }
    }
    public SquareMatrix(int[][] mtx) {
        set(mtx);
    }
}
```

```

}
public void set(int[][] mtx) {
    nDim = mtx.length;
    matrix = new int[nDim][nDim];
    for (int i = 0; i < nDim; i++) {
        for (int j = 0; j < nDim; j++) {
            matrix[i][j] = mtx[i][j];
        }
    }
}
public int[][] get() {
    return matrix;
}

public SquareMatrix add( SquareMatrix rMtx) {
    int[][] resMtx = new int[nDim][nDim];
    int[][] rightMtx = rMtx.get();

    for (int i = 0; i < nDim; i++) {
        for (int j = 0; j < nDim; j++) {
            resMtx[i][j] = matrix[i][j] + rightMtx[i][j];
        }
    }

    return new SquareMatrix(resMtx);
}

public SquareMatrix sub(SquareMatrix rMtx) {
    int[][] resMtx = new int[nDim][nDim];
    int[][] rightMtx = rMtx.get();

    for (int i = 0; i < nDim; i++) {
        for (int j = 0; j < nDim; j++) {
            resMtx[i][j] = matrix[i][j] - rightMtx[i][j];
        }
    }

    return new SquareMatrix(resMtx);
}

public SquareMatrix mul(SquareMatrix rMtx) {
    int[][] resMtx = new int[nDim][nDim];
    int[][] rightMtx = rMtx.get();

    for (int i = 0; i < nDim; i++) {
        for (int j = 0; j < nDim; j++) {
            resMtx[i][j] = 0;
            for (int k = 0; k < nDim; k++) {
                resMtx[i][j] += matrix[i][k] * rightMtx[k][j];
            }
        }
    }

    return new SquareMatrix(resMtx);
}

public int firstNorma() {
    int max = 0;
    for (int i = 0; i < nDim; i++) {
        max += matrix[0][i];
    }
    for (int i = 1; i < nDim; i++) {
        int sum = 0;
        for (int j = 0; j < nDim; j++) {

```

```

        sum += matrix[i][j];
    }
    max = Math.max(max, sum);
}
return max;
}

public int secondNorma() {
    int max = 0;
    for (int i = 0; i < nDim; i++) {
        max += matrix[i][0];
    }
    for (int i = 0; i < nDim; i++) {
        int sum = 0;
        for (int j = 1; j < nDim; j++) {
            sum += matrix[j][i];
        }
        max = Math.max(max, sum);
    }
    return max;
}

@Override
public String toString() {
    StringBuilder mtxString = new StringBuilder("");
    for (int i = 0; i < nDim; i++) {
        for (int j = 0; j < nDim-1; j++) {
            mtxString.append(matrix[i][j]);
            mtxString.append("\t");
        }
        mtxString.append(matrix[i][nDim-1]);
        mtxString.append("\n");
    }
    return mtxString.toString();
}
}

```

Применение класса SquareMatrix в основной программе Matrix:

```

package laba_3.task_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
public class Matrix {
    public static void main(String[] args) {
        int[][] n = {{1, 2}, {3, 4}};
        int[][] m = {{5, 6}, {7, 8}};

        SquareMatrix mtxA = new SquareMatrix(n);
        SquareMatrix mtxB = new SquareMatrix(m);
        SquareMatrix mtxC = new SquareMatrix(2);

        SquareMatrix[] mtxmas = new SquareMatrix[2];
        mtxmas[0] = mtxA;
        mtxmas[1] = mtxB;

        SquareMatrix addMtx = mtxmas[0].add(mtxmas[1]);
        SquareMatrix subMtx = mtxmas[0].sub(mtxmas[1]);
        SquareMatrix mulMtx = mtxmas[0].mul(mtxmas[1]);
    }
}

```



```

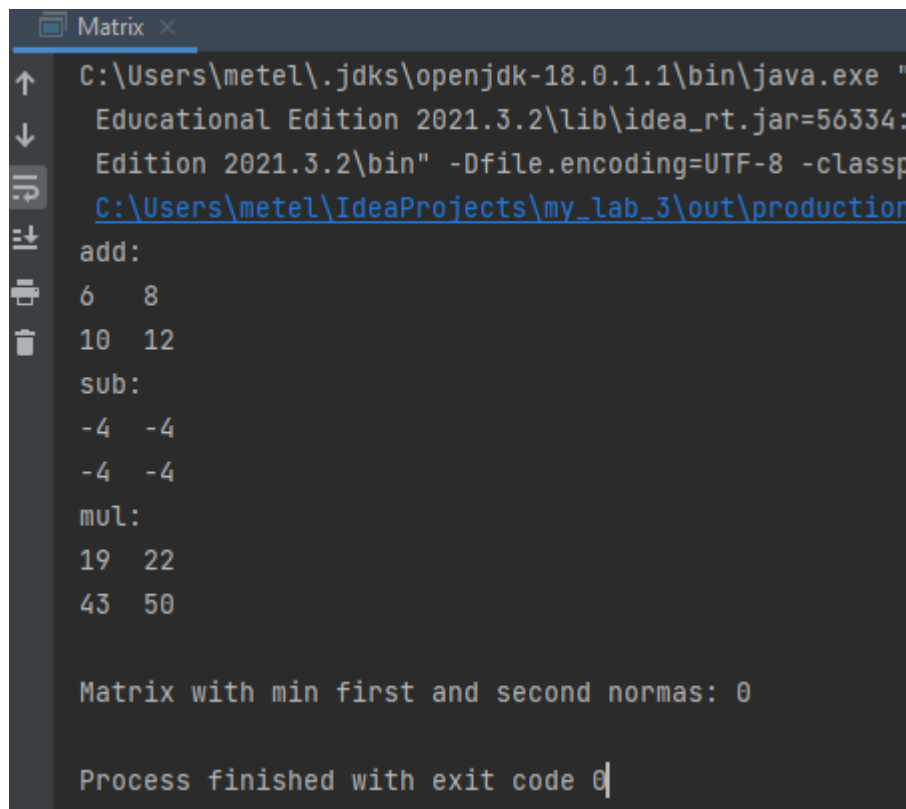
        System.out.println("add:\n" + mtxmas[0].add(mtxmas[1]).toString() +
            "sub:\n" + mtxmas[0].sub(mtxmas[1]).toString() +
            "mul:\n" + mtxmas[0].mul(mtxmas[1]).toString());

        int minIdx = -1;
        if ((mtxmas[0].firstNorma() < mtxmas[1].firstNorma()) &&
(mtxmas[0].secondNorma() < mtxmas[1].secondNorma())) {
            minIdx = 0;
        } else if ((mtxmas[1].firstNorma() < mtxmas[0].firstNorma()) &&
(mtxmas[1].secondNorma() < mtxmas[0].secondNorma())) {
            minIdx = 1;
        }

        System.out.println("Matrix with min first and second normas: " + (minIdx == -
1 ? "None" : minIdx));
    }
}

```

Результат выполнения подзадачи 2:



```

Matrix x
C:\Users\metel\.jdk\openjdk-18.0.1.1\bin\java.exe "-
Educational Edition 2021.3.2\lib\idea_rt.jar=56334:0
Edition 2021.3.2\bin" -Dfile.encoding=UTF-8 -classpa
C:\Users\metel\IdeaProjects\my_lab_3\out\production\

add:
6 8
10 12
sub:
-4 -4
-4 -4
mul:
19 22
43 50

Matrix with min first and second normas: 0

Process finished with exit code 0

```

Задание №2:

1. Patient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Номер медицинской карты, Диагноз. Создать массив объектов. Вывести: а) список пациентов, имеющих данный диагноз; б) список пациентов, номер медицинской карты у которых находится в заданном интервале.
2. Abiturient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки. Создать массив объектов. Вывести: а) список абитуриентов, имеющих неудовлетворительные оценки; б) список абитуриентов, средний балл у которых выше заданного; с) выбрать заданное число n абитуриентов, имеющих самый высокий средний балл (вывести также полный список абитуриентов, имеющих полупроходной балл).

Листинг выполнения подзадачи 1:

Класс Patient:

```
package laba_3.task_2.patient;

public class Patient {
    private int id;
    private String name;
    private String surname;
    private String lastname;
    private String address;
    private String phone;
    private int cardNumber;
    private String diagnosis;

    public Patient() {
    }

    public Patient(int id, String name, String surname, String lastname, String
address, String phone, int cardNumber, String diagnosis) {
        this.id = id;
        this.name = name;
        this.surname = surname;
        this.lastname = lastname;
        this.address = address;
        this.phone = phone;
        this.cardNumber = cardNumber;
        this.diagnosis = diagnosis;
    }

    public int getId() {
        return id;
    }

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

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

```

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

public String getSurname() {
    return surname;
}

public void setSurname(String surname) {
    this.surname = surname;
}

public String getLastname() {
    return lastname;
}

public void setLastname(String lastname) {
    this.lastname = lastname;
}

public String getAddress() {
    return address;
}

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

public String getPhone() {
    return phone;
}

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

public int getCardNumber() {
    return cardNumber;
}

public void setCardNumber(int cardNumber) {
    this.cardNumber = cardNumber;
}

public String getDiagnosis() {
    return diagnosis;
}

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

@Override
public String toString() {
    return "Patient{" +
        "id=" + id +
        ", name='" + name + '\'' +
        ", surname='" + surname + '\'' +
        ", lastname='" + lastname + '\'' +
        ", address='" + address + '\'' +
        ", phone='" + phone + '\'' +
        ", cardNumber=" + cardNumber +
        ", diagnosis='" + diagnosis + '\'' +
        '}';
}

```

```
}  
}
```

Применение класса Patient в основной программе Main_Patient:

```
package laba_3.task_2.patient;  
import java.util.ArrayList;  
  
public class Main_Patient {  
    public static void main(String[] args) {  
  
        Patient[] patientsArray = createPatientsArray();  
        System.out.println("Patients:");  
        for (Patient p: patientsArray) {  
            System.out.println(p);  
        }  
  
        Patient[] patientsWithDiabetes = chooseByDiagnosis(patientsArray,  
"Diabetes");  
        System.out.println();  
        System.out.println("Patients with Diabetes:");  
        for (Patient p: patientsWithDiabetes) {  
            System.out.println(p);  
        }  
        Patient[] patientsInRange = chooseByCardNumber(patientsArray, 130, 140);  
        System.out.println();  
        System.out.println("Patients with CardNumbers in range 130...140:");  
        for (Patient p: patientsInRange) {  
            System.out.println(p);  
        }  
  
    }  
  
    private static Patient[] createPatientsArray(){  
        Patient p1 = new Patient(1,"Ivan", "Ivanov", "Ivanovich", "House 5", "8-968-  
374-26-47", 132, "Diabetes");  
        Patient p2 = new Patient(2,"Petr", "Petrov", "Petrovich", "House 3", "8-969-  
375-27-74", 148, "COVID-19");  
        Patient p3 = new Patient(3,"Dmitry", "Smirnov", "Ivanovich", "House 9", "8-  
977-234-86-07", 119, "Diabetes");  
        Patient p4 = new Patient(4,"Ivan", "Smirnov", "Andreevich", "House 5", "8-  
978-306-36-43", 135, "COVID-19");  
        Patient p5 = new Patient(5,"Alexander", "Ivanov", "Ilich", "House 11", "8-  
961-333-28-17", 138, "Flu");  
        return new Patient[]{p1, p2, p3, p4, p5};  
    }  
  
    private static Patient[] chooseByDiagnosis(Patient[] patientsArray, String  
diagnosis){  
        ArrayList<Patient> newPatientsArray = new ArrayList<>();  
        for (int i = 0; i < patientsArray.length; i++) {  
            if(patientsArray[i].getDiagnosis().equals(diagnosis)){  
                newPatientsArray.add(patientsArray[i]);  
            }  
        }  
        return (Patient[]) newPatientsArray.toArray(new  
Patient[newPatientsArray.size()]);  
    }  
  
    private static Patient[] chooseByCardNumber(Patient[] patientsArray, int  
startBound, int endBound){  
        ArrayList<Patient> newPatientsArray = new ArrayList<>();  
        for (int i = 0; i < patientsArray.length; i++) {
```

```

        if(patientsArray[i].getCardNumber() >= startBound &&
patientsArray[i].getCardNumber() <= endBound){
            newPatientsArray.add(patientsArray[i]);
        }
    }
    return (Patient[]) newPatientsArray.toArray(new
Patient[newPatientsArray.size()]);
}
}

```

Результат выполнения подзадачи 1:

```

C:\Users\metel\IdeaProjects\my_lab_3\out\production\my_lab_3 laba_3.task_2.patient.Main_Patient
Patients:
Patient{id=1, name='Ivan', surname='Ivanov', lastname='Ivanovich', address='House 5',
phone='8-968-374-26-47', cardNumber=132, diagnosis='Diabetes'}
Patient{id=2, name='Petr', surname='Petrov', lastname='Petrovich', address='House 3',
phone='8-969-375-27-74', cardNumber=148, diagnosis='COVID-19'}
Patient{id=3, name='Dmitry', surname='Smirnov', lastname='Ivanovich', address='House 9',
phone='8-977-234-86-07', cardNumber=119, diagnosis='Diabetes'}
Patient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5',
phone='8-978-306-36-43', cardNumber=135, diagnosis='COVID-19'}
Patient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', cardNumber=138, diagnosis='Flu'}

Patients with Diabetes:
Patient{id=1, name='Ivan', surname='Ivanov', lastname='Ivanovich', address='House 5',
phone='8-968-374-26-47', cardNumber=132, diagnosis='Diabetes'}
Patient{id=3, name='Dmitry', surname='Smirnov', lastname='Ivanovich', address='House 9',
phone='8-977-234-86-07', cardNumber=119, diagnosis='Diabetes'}

Patients with CardNumbers in range 130...140:
Patient{id=1, name='Ivan', surname='Ivanov', lastname='Ivanovich', address='House 5',
phone='8-968-374-26-47', cardNumber=132, diagnosis='Diabetes'}
Patient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5',
phone='8-978-306-36-43', cardNumber=135, diagnosis='COVID-19'}
Patient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', cardNumber=138, diagnosis='Flu'}

Process finished with exit code 0

```

Листинг выполнения подзадачи 2:

Класс Abiturient:

```

package laba_3.task_2.abitur;
import java.util.ArrayList;

public class Abiturient {
    private int id;
    private String name;
    private String surname;
    private String lastname;

```

```

private String address;
private String phone;
private ArrayList<Integer> marks;

public Abiturient() {
}

public Abiturient(int id, String name, String surname, String lastname, String
address, String phone, ArrayList<Integer> marks) {
    this.id = id;
    this.name = name;
    this.surname = surname;
    this.lastname = lastname;
    this.address = address;
    this.phone = phone;
    this.marks = marks;
}

public int getId() {
    return id;
}

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

public String getName() {
    return name;
}

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

public String getSurname() {
    return surname;
}

public void setSurname(String surname) {
    this.surname = surname;
}

public String getLastName() {
    return lastname;
}

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

public String getAddress() {
    return address;
}

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

public String getPhone() {
    return phone;
}

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

```

```

    }

    public ArrayList<Integer> getMarks() {
        return marks;
    }

    public void setMarks(ArrayList<Integer> marks) {
        this.marks = marks;
    }

    @Override
    public String toString() {
        return "Abiturient{" +
            "id=" + id +
            ", name='" + name + '\'' +
            ", surname='" + surname + '\'' +
            ", lastname='" + lastname + '\'' +
            ", address='" + address + '\'' +
            ", phone='" + phone + '\'' +
            ", marks=" + marks +
            '}';
    }
}

```

Применение класса Abiturient в основной программе Main_Abiturient:

```

package laba_3.task_2.abitur;

import java.util.*;

public class Main_Abiturient {
    public static void main(String[] args) {

        Abiturient[] abiturientsArray = createAbiturientsArray();
        System.out.println("Abiturients:");
        for (Abiturient a: abiturientsArray) {
            System.out.println(a);
        }
        Abiturient[] abiturientsWithNeuds = chooseWithNeuds(abiturientsArray);
        System.out.println();
        System.out.println("Abiturients with neuds:");
        for (Abiturient a: abiturientsWithNeuds) {
            System.out.println(a);
        }

        Abiturient[] abiturientsWithHigherAVG = chooseHigherAVGMark(abiturientsArray,
4f);
        System.out.println();
        System.out.println("Abiturients with average mark higher then 4:");
        for (Abiturient a: abiturientsWithHigherAVG) {
            System.out.println(a);
        }

        Abiturient[] abiturientsBestN = chooseBest(abiturientsArray, 2);
        System.out.println();
        System.out.println("Best 2 abiturients:");
        for (Abiturient a: abiturientsBestN) {
            System.out.println(a);
        }

    }

    private static Abiturient[] createAbiturientsArray(){

```

```

        Abiturient a1 = new Abiturient(1,"Ivan", "Ivanov", "Ivanovich", "House 5",
"8-968-374-26-47", new ArrayList<Integer>(Arrays.asList(3, 2, 5)));
        Abiturient a2 = new Abiturient(2,"Petr", "Petrov", "Petrovich", "House 3",
"8-969-375-27-74", new ArrayList<Integer>(Arrays.asList(4, 4, 5)));
        Abiturient a3 = new Abiturient(3,"Dmitry", "Smirnov", "Ivanovich", "House 9",
"8-977-234-86-07", new ArrayList<Integer>(Arrays.asList(5, 4, 5)));
        Abiturient a4 = new Abiturient(4,"Ivan", "Smirnov", "Andreevich", "House 5",
"8-978-306-36-43", new ArrayList<Integer>(Arrays.asList(3, 2, 4)));
        Abiturient a5 = new Abiturient(5,"Alexander", "Ivanov", "Ilich", "House 11",
"8-961-333-28-17", new ArrayList<Integer>(Arrays.asList(5, 5, 5)));
        return new Abiturient[]{a1, a2, a3, a4, a5};
    }

    private static Abiturient[] chooseWithNeuds(Abiturient[] abiturientsArray){
        ArrayList<Abiturient> newAbiturientsArray = new ArrayList<>();
        for (int i = 0; i < abiturientsArray.length; i++) {
            if(abiturientsArray[i].getMarks().contains(2)){
                newAbiturientsArray.add(abiturientsArray[i]);
            }
        }
        return (Abiturient[]) newAbiturientsArray.toArray(new
Abiturient[newAbiturientsArray.size()]);
    }

    private static Abiturient[] chooseHigherAVGMark(Abiturient[] abiturientsArray,
float mark){
        ArrayList<Abiturient> newAbiturientsArray = new ArrayList<>();
        for (int i = 0; i < abiturientsArray.length; i++) {
            float avg = 0;
            for (Integer m : abiturientsArray[i].getMarks()) {
                avg += m;
            }
            avg = avg/abiturientsArray[i].getMarks().size();
            if(avg > mark){
                newAbiturientsArray.add(abiturientsArray[i]);
            }
        }
        return (Abiturient[]) newAbiturientsArray.toArray(new
Abiturient[newAbiturientsArray.size()]);
    }

    private static Abiturient[] chooseBest(Abiturient[] abiturientsArray, Integer n){
        ArrayList<Abiturient> newAbiturientsArray = new ArrayList<>();

        SortedMap<Float, ArrayList<Abiturient>> map = new TreeMap<>();
        for (int i = 0; i < abiturientsArray.length; i++){
            float avg = 0;
            for (Integer m : abiturientsArray[i].getMarks()) {
                avg += m;
            }
            avg = avg / abiturientsArray[i].getMarks().size();

            if (map.containsKey(avg)) {
                map.get(avg).add(abiturientsArray[i]);
            } else {
                map.put(avg, new ArrayList<>());
                map.get(avg).add(abiturientsArray[i]);
            }
        }
        System.out.println(map);

        int j = 0;
        int avg_num = -1;
        List<Float> floatList = new ArrayList<Float>(map.keySet());
    }

```



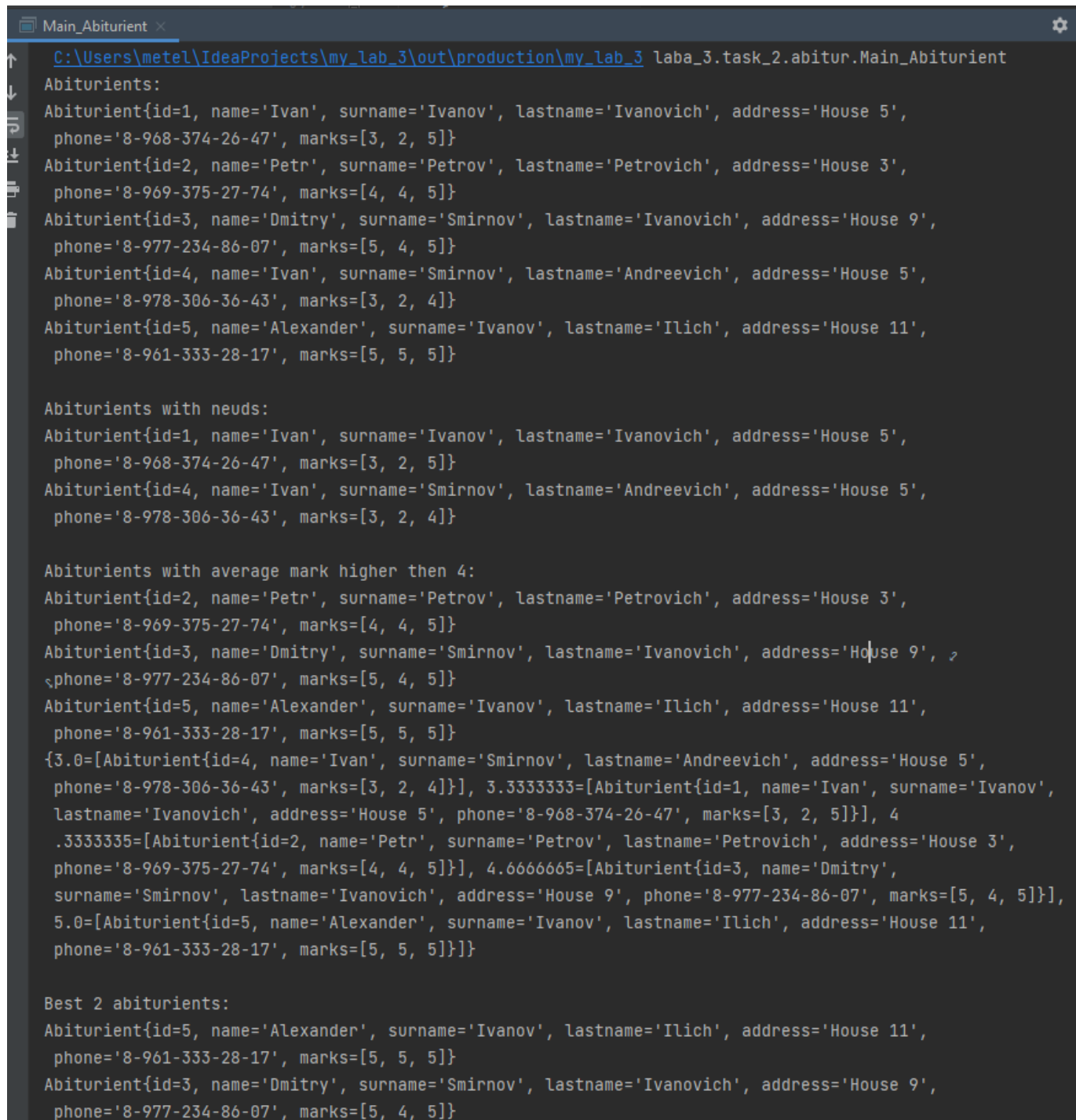
```

        Collections.reverse(floatList);
        while (j < n){
            avg_num++;
            for (int i = 0; i < map.get(floatList.get(avg_num)).size(); i++) {
                newAbiturientsArray.add(map.get(floatList.get(avg_num)).get(i));
                j++;
            }
        }

        return (Abiturient[]) newAbiturientsArray.toArray(new
Abiturient[newAbiturientsArray.size()]);
    }
}

```

Результат выполнения подзадачи 2:



```

Main_Abiturient x
C:\Users\metel\IdeaProjects\my_lab_3\out\production\my_lab_3 laba_3.task_2.abitur.Main_Abiturient
Abiturients:
Abiturient{id=1, name='Ivan', surname='Ivanov', lastname='Ivanovich', address='House 5',
phone='8-968-374-26-47', marks=[3, 2, 5]}
Abiturient{id=2, name='Petr', surname='Petrov', lastname='Petrovich', address='House 3',
phone='8-969-375-27-74', marks=[4, 4, 5]}
Abiturient{id=3, name='Dmitry', surname='Smirnov', lastname='Ivanovich', address='House 9',
phone='8-977-234-86-07', marks=[5, 4, 5]}
Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5',
phone='8-978-306-36-43', marks=[3, 2, 4]}
Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', marks=[5, 5, 5]}

Abiturients with neuds:
Abiturient{id=1, name='Ivan', surname='Ivanov', lastname='Ivanovich', address='House 5',
phone='8-968-374-26-47', marks=[3, 2, 5]}
Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5',
phone='8-978-306-36-43', marks=[3, 2, 4]}

Abiturients with average mark higher then 4:
Abiturient{id=2, name='Petr', surname='Petrov', lastname='Petrovich', address='House 3',
phone='8-969-375-27-74', marks=[4, 4, 5]}
Abiturient{id=3, name='Dmitry', surname='Smirnov', lastname='Ivanovich', address='House 9',
phone='8-977-234-86-07', marks=[5, 4, 5]}
Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', marks=[5, 5, 5]}
{3.0=[Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5',
phone='8-978-306-36-43', marks=[3, 2, 4]}], 3.3333333=[Abiturient{id=1, name='Ivan', surname='Ivanov',
lastname='Ivanovich', address='House 5', phone='8-968-374-26-47', marks=[3, 2, 5]}], 4
.3333335=[Abiturient{id=2, name='Petr', surname='Petrov', lastname='Petrovich', address='House 3',
phone='8-969-375-27-74', marks=[4, 4, 5]}], 4.6666665=[Abiturient{id=3, name='Dmitry',
surname='Smirnov', lastname='Ivanovich', address='House 9', phone='8-977-234-86-07', marks=[5, 4, 5]}],
5.0=[Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', marks=[5, 5, 5]}]}

Best 2 abiturients:
Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11',
phone='8-961-333-28-17', marks=[5, 5, 5]}
Abiturient{id=3, name='Dmitry', surname='Smirnov', lastname='Ivanovich', address='House 9',
phone='8-977-234-86-07', marks=[5, 4, 5]}

```

Задание №3:

- 1 Создать объект класса Одномерный массив, используя класс Массив. Методы: создать, вывести на консоль, выполнить операции (сложить, вычесть, перемножить).
- 2 Создать объект класса Простая дробь, используя класс Число. Методы: вывод на экран, сложение, вычитание, умножение, деление.

Листинг выполнения подзадачи 1:

Интерфейс Array:

```
package laba_3.task_3;

public interface Array {
    void summation(Object o);
    void subtraction(Object o);
    void multiplication(Object o);
}
```

Класс Array_class:

```
package laba_3.task_3;
import java.util.ArrayList;
import java.util.Objects;

public class Array_class implements Array{
    private ArrayList<Float> content;
    public Array_class() {
    }
    public Array_class(ArrayList<Float> content) {
        this.content = content;
    }

    @Override
    public void summation(Object o) {
        if(o.getClass().equals(Array_class.class)){
            if(this.content.size() == ((Array_class) o).content.size()){
                for (int i = 0; i < content.size(); i++) {
                    this.content.set(i, (this.content.get(i) + ((Array_class)
o).content.get(i)));
                }
            } else {
                System.out.println("Arrays length mismatch!");
            }
        } else {
            System.out.println("Class mismatch!");
        }
    }

    @Override
    public void subtraction(Object o) {
        if(o.getClass().equals(Array_class.class)){
            if(this.content.size() == ((Array_class) o).content.size()){
                for (int i = 0; i < content.size(); i++) {
                    this.content.set(i, (this.content.get(i) - ((Array_class)
```

```

o).content.get(i));
    }
    } else {
        System.out.println("Arrays length mismatch!");
    }
} else {
    System.out.println("Class mismatch!");
}
}

@Override
public void multiplication(Object o) {
    if(o.getClass().equals(Array_class.class)){
        if(this.content.size() == ((Array_class) o).content.size()){
            for (int i = 0; i < content.size(); i++) {
                this.content.set(i, (this.content.get(i) * ((Array_class)
o).content.get(i)));
            }
        } else {
            System.out.println("Arrays length mismatch!");
        }
    } else {
        System.out.println("Class mismatch!");
    }
}

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

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

@Override
public String toString() {
    return "Array1Dim{" +
        "content=" + content +
        '}';
}

public Array_class getClone(){
    return new Array_class((ArrayList)(this.content.clone()));
}
}

```

Основная программа Main_Array:

```

package laba_3.task_3;
import java.util.ArrayList;
import java.util.Arrays;

public class Main_Array {
    public static void main(String[] args) {

        Array_class ar1 = new Array_class(new ArrayList<Float>(Arrays.asList(1.f,
2.f, 3.f)));
    }
}

```

```

        Array_class ar2 = new Array_class(new ArrayList<Float>(Arrays.asList(4.f,
5.f, 6.f)));

        System.out.println("Array 1 : ");
        System.out.println(ar1);
        System.out.println("Array 2 : ");
        System.out.println(ar2);

        Array_class ar_sum = ar1.getClone();
        Array_class ar_sub = ar1.getClone();
        Array_class ar_mul = ar1.getClone();

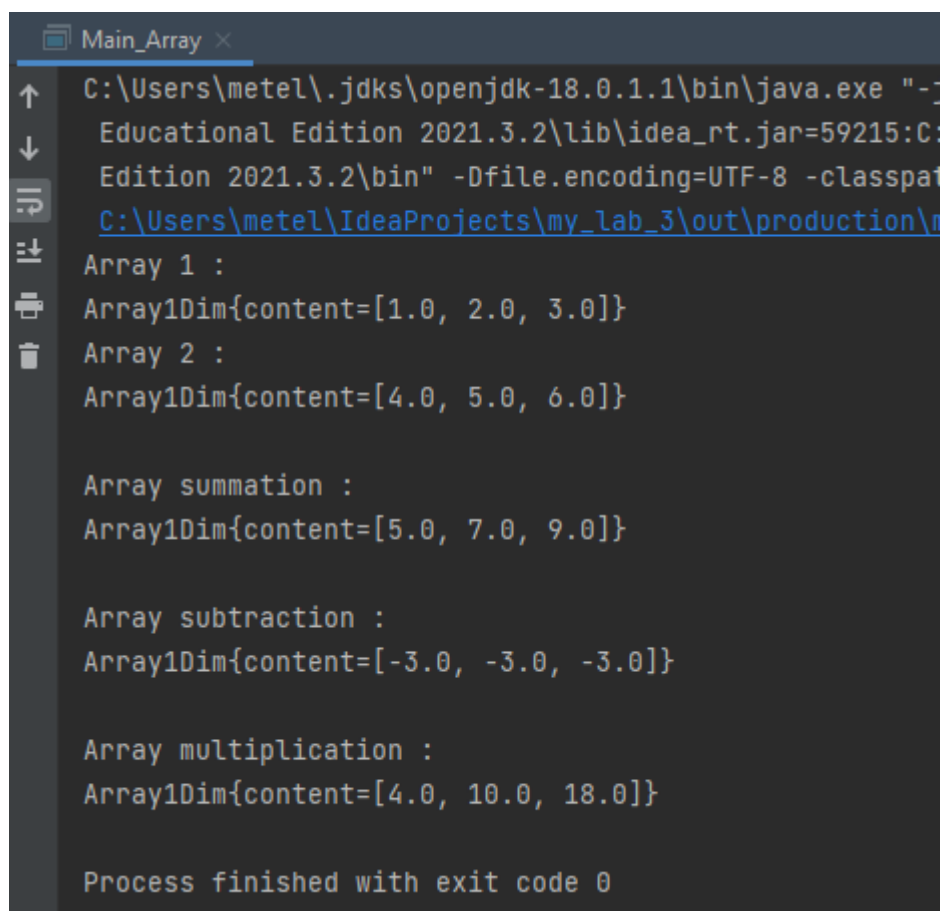
        System.out.println();
        System.out.println("Array summation : ");
        ar_sum.summation(ar2);
        System.out.println(ar_sum);

        System.out.println();
        System.out.println("Array subtraction : ");
        ar_sub.subtraction(ar2);
        System.out.println(ar_sub);

        System.out.println();
        System.out.println("Array multiplication : ");
        ar_mul.multiplication(ar2);
        System.out.println(ar_mul);
    }
}

```

Результат выполнения подзадачи 1:



```

Main_Array x
C:\Users\metel\.jdk\openjdk-18.0.1.1\bin\java.exe "-j
Educational Edition 2021.3.2\lib\idea_rt.jar=59215:C
Edition 2021.3.2\bin" -Dfile.encoding=UTF-8 -classpat
C:\Users\metel\IdeaProjects\my_lab_3\out\production\
Array 1 :
Array1Dim{content=[1.0, 2.0, 3.0]}
Array 2 :
Array1Dim{content=[4.0, 5.0, 6.0]}

Array summation :
Array1Dim{content=[5.0, 7.0, 9.0]}

Array subtraction :
Array1Dim{content=[-3.0, -3.0, -3.0]}

Array multiplication :
Array1Dim{content=[4.0, 10.0, 18.0]}

Process finished with exit code 0

```

Листинг выполнения подзадачи 2:

Класс Fraction:

```
package laba_3.task_3;
import java.util.Objects;

public class Fraction {
    private Number numerator;
    private Number denominator;

    public Fraction() {
    }

    public Fraction(Number numerator, Number denominator) {
        this.numerator = numerator;
        this.denominator = denominator;
    }

    public Fraction summation(Fraction other){
        Fraction result = new
Fraction(other.denominator.multiplication(this.numerator).summation(this.denominator.
multiplication(other.numerator)),
this.denominator.multiplication(other.denominator));
        result.normalize();
        return result;
    }

    public Fraction subtraction(Fraction other){
        Fraction result = new
Fraction(other.denominator.multiplication(this.numerator).subtraction(this.denominator.
multiplication(other.numerator)),
this.denominator.multiplication(other.denominator));
        result.normalize();
        return result;
    }

    public Fraction multiplication(Fraction other){
        Fraction result = new
Fraction(this.numerator.multiplication(other.numerator),
this.denominator.multiplication(other.denominator));
        result.normalize();
        return result;
    }

    public Fraction division(Fraction other){
        Fraction temp = new Fraction(other.denominator, other.numerator);
        Fraction result = this.multiplication(temp);
        result.normalize();
        return result;
    }

    public Number getNumerator() {
        return numerator;
    }

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

```

public Number getDenominator() {
    return denominator;
}

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

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Fraction fraction = (Fraction) o;
    return Objects.equals(numerator, fraction.numerator) &&
Objects.equals(denominator, fraction.denominator);
}

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

@Override
public String toString() {
    return "Fraction{" +
        numerator +
        " / " + denominator +
        '}';
}

private void normalize(){

    if(this.numerator.getNumber() > this.denominator.getNumber()){
        if(this.numerator.getNumber() % this.denominator.getNumber() == 0){
            this.numerator = this.numerator.division(this.denominator);
            this.denominator = new Number(1);
        } else {
            int i = 2;
            while (i <= (int) Math.sqrt(this.denominator.getNumber())) {
                if(this.numerator.getNumber() % i == 0 &&
this.denominator.getNumber() % i == 0){
                    this.numerator = this.numerator.division(new Number(i));
                    this.denominator = this.denominator.division(new Number(i));
                } else {
                    i++;
                }
            }
        }
    } else if(this.numerator.getNumber() < this.denominator.getNumber()) {
        if(this.denominator.getNumber() % this.numerator.getNumber() == 0){
            this.denominator = this.denominator.division(this.numerator);
            this.numerator = new Number(1);
        } else {
            int i = 2;
            while (i <= (int) Math.sqrt(this.numerator.getNumber())) {
                if(this.numerator.getNumber() % i == 0 &&
this.denominator.getNumber() % i == 0){
                    this.numerator = this.numerator.division(new Number(i));
                    this.denominator = this.denominator.division(new Number(i));
                } else {
                    i++;
                }
            }
        }
    }
}

```

```

    }
} else {
    this.numerator = new Number(1);
    this.denominator = new Number(1);
}
}
}

```

Основная программа Main_Fraction:

```

package laba_3.task_3;

public class Main_Fraction {
    public static void main(String[] args) {

        Fraction fr1 = new Fraction(new Number(1), new Number(2));
        Fraction fr2 = new Fraction(new Number(1), new Number(4));

        System.out.println("Fraction 1:");
        System.out.println(fr1);
        System.out.println("Fraction 2:");
        System.out.println(fr2);

        System.out.println();
        Fraction fr_sum = fr1.summation(fr2);
        System.out.println("Summation: ");
        System.out.println(fr_sum);

        System.out.println();
        Fraction fr_sub = fr1.subtraction(fr2);
        System.out.println("Subtraction: ");
        System.out.println(fr_sub);

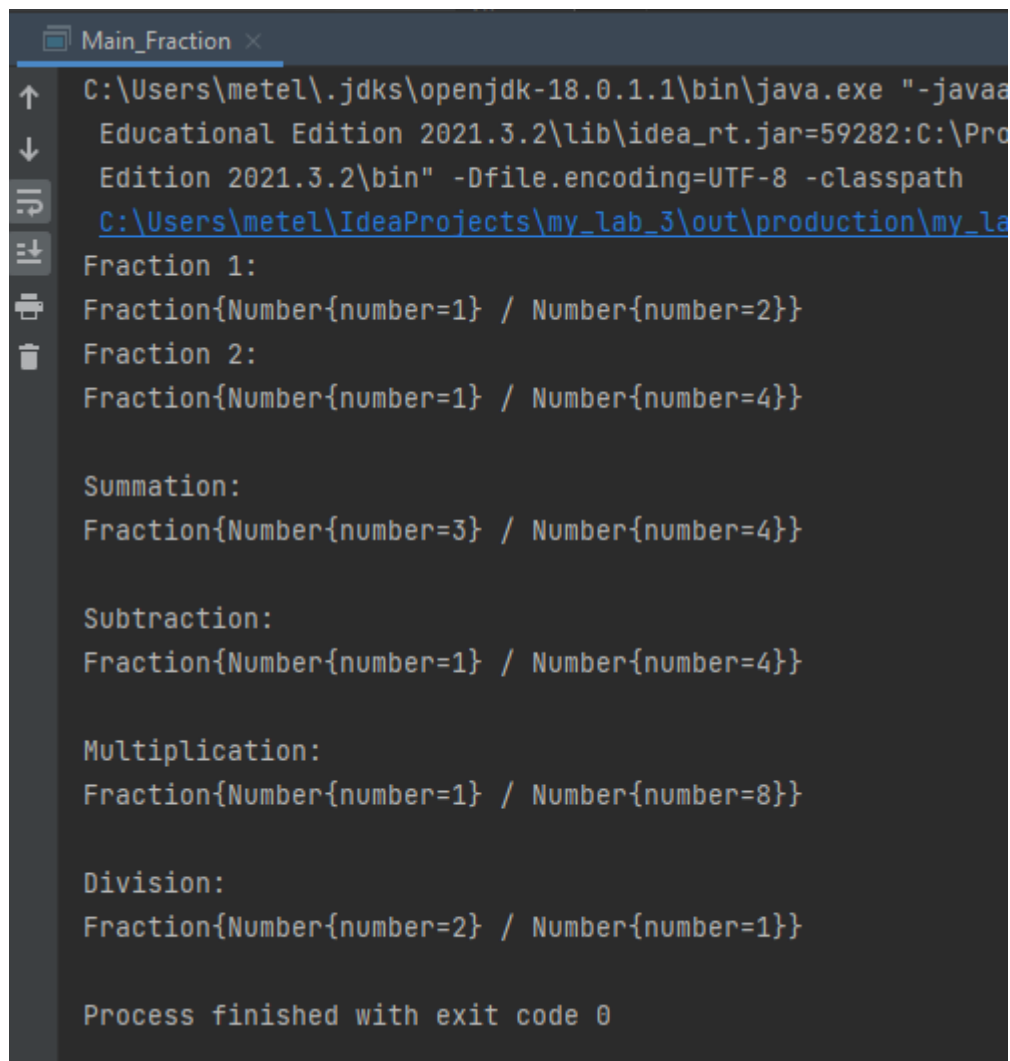
        System.out.println();
        Fraction fr_mul = fr1.multiplication(fr2);
        System.out.println("Multiplication: ");
        System.out.println(fr_mul);

        System.out.println();
        Fraction fr_div = fr1.division(fr2);
        System.out.println("Division: ");
        System.out.println(fr_div);

    }
}

```

Результат выполнения подзадачи 2:



```

C:\Users\metel\.jdk\openjdk-18.0.1.1\bin\java.exe "-javaa
Educational Edition 2021.3.2\lib\idea_rt.jar=59282:C:\Pro
Edition 2021.3.2\bin" -Dfile.encoding=UTF-8 -classpath
C:\Users\metel\IdeaProjects\my_lab_3\out\production\my_la
Fraction 1:
Fraction{Number{number=1} / Number{number=2}}
Fraction 2:
Fraction{Number{number=1} / Number{number=4}}

Summation:
Fraction{Number{number=3} / Number{number=4}}

Subtraction:
Fraction{Number{number=1} / Number{number=4}}

Multiplication:
Fraction{Number{number=1} / Number{number=8}}

Division:
Fraction{Number{number=2} / Number{number=1}}

Process finished with exit code 0

```

Задание №4:

1. 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.
2. Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает Экзамены. Преподаватель выставляет Оценку. Система подсчитывает средний балл и определяет Абитуриентов, зачисленных в учебное заведение.

Листинг выполнения подзадачи 1:

Класс Staff:

```
package laba_3.task_4.hospital;
import java.util.Objects;

public class Staff {
    private int id;
    private String name;
    private String position;

    public Staff() {
    }

    public Staff(int id, String name, String position) {
        this.id = id;
        this.name = name;
        this.position = position;
    }

    public int getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

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

    public String getPosition() {
        return position;
    }

    public void setPosition(String position) {
        this.position = position;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Staff staff = (Staff) o;
        return id == staff.id && Objects.equals(name, staff.name) &&
Objects.equals(position, staff.position);
    }

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

    @Override
    public String toString() {
        return "Staff{" +
```

```

        "id=" + id +
        ", name='" + name + '\'' +
        ", position='" + position + '\'' +
        '}';
    }
}

```

Класс Patient:

```

package laba_3.task_4.hospital;
import java.util.Objects;

public class Patient {
    private String name;
    private int id;
    private boolean inHospital;
    private String reason;

    public Patient() {
    }

    public Patient(String name, int id ) {
        this.name = name;
        this.id = id;
        this.inHospital = true;
        this.reason = "";
    }

    public String getName() {
        return name;
    }

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

    public int getId() {
        return id;
    }

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

    public boolean isInHospital() {
        return inHospital;
    }

    public void setInHospital(boolean inHospital) {
        this.inHospital = inHospital;
    }

    public String getReason() {
        return reason;
    }

    public void setReason(String reason) {
        this.reason = reason;
    }

    public void dismiss(String reason){
        this.inHospital = false;
        this.reason = reason;
    }
}

```

```

    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Patient patient = (Patient) o;
        return id == patient.id && inHospital == patient.inHospital &&
Objects.equals(name, patient.name) && Objects.equals(reason, patient.reason);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name, id, inHospital, reason);
    }

    @Override
    public String toString() {
        return "Patient{" +
            "name='" + name + '\'' +
            ", id=" + id +
            ", inHospital=" + inHospital +
            ", reason='" + reason + '\'' +
            '}';
    }
}

```

Класс Assignment:

```

package laba_3.task_4.hospital;
import java.util.ArrayList;
import java.util.Objects;

public class Assignment {
    private ArrayList<String> procedures;
    private ArrayList<String> drugs;
    private ArrayList<String> surgeries;
    private boolean done;

    public Assignment() {
        this.procedures = new ArrayList<>();
        this.drugs = new ArrayList<>();
        this.surgeries = new ArrayList<>();
        this.done = false;
    }

    public void addProcedure(String proc) {
        this.procedures.add(proc);
    }

    public void addDrug(String drug) {
        this.drugs.add(drug);
    }

    public void addSurgery(String surg) {
        this.surgeries.add(surg);
    }

    public void complete() {
        this.done = true;
    }

    @Override

```

```

    public String toString() {
        return "Assignment{" +
            "procedures=" + procedures +
            ", drugs=" + drugs +
            ", surgeries=" + surgeries +
            ", done=" + done +
            '}';
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Assignment that = (Assignment) o;
        return done == that.done && Objects.equals(procedures, that.procedures) &&
            Objects.equals(drugs, that.drugs) && Objects.equals(surgeries, that.surgeries);
    }

    @Override
    public int hashCode() {
        return Objects.hash(procedures, drugs, surgeries, done);
    }
}

```

Класс Hospital:

```

package laba_3.task_4.hospital;

import java.util.ArrayList;
import java.util.HashMap;

public class Hospital {
    private HashMap<Integer, Integer> client_doctor;
    private HashMap<Integer, Assignment> client_assign;
    private HashMap<Assignment, Integer> assign_staff;
    private ArrayList<Staff> staffArray;
    private ArrayList<Patient> patientArray;

    public Hospital() {
        this.client_doctor = new HashMap<>();
        this.client_assign = new HashMap<>();
        this.assign_staff = new HashMap<>();
        this.staffArray = new ArrayList<>();
        this.patientArray = new ArrayList<>();
    }

    public void addPatient(Patient patient){
        this.patientArray.add(patient);
    }

    public void addStaff(Staff staff){
        this.staffArray.add(staff);
    }

    public void setDoctor(int pat_id, int doc_id){
        this.client_doctor.put(pat_id, doc_id);
    }

    public void setAssignment(int pat_id, Assignment assignment){
        this.client_assign.put(pat_id, assignment);
    }

    public void completeAssignment(int pat_id, int staff_id){

```

```

        this.assign_staff.put(this.client_assign.get(pat_id), staff_id);
        this.client_assign.get(pat_id).complete();
    }

    public void dismissPatient(Patient patient, String reason){
        int pat_id = patient.getId();
        patient.dismiss(reason);
        this.client_doctor.remove(pat_id);
    }

    @Override
    public String toString() {
        return "Hospital{" + "\n" +
            "    ---client_doctor=" + client_doctor + ",\n" +
            "    ---client_assign=" + client_assign + ",\n" +
            "    ---assign_staff=" + assign_staff + ",\n" +
            "    ---staffArray=" + staffArray + ",\n" +
            "    ---patientArray=" + patientArray +
            '}' ;
    }
}

```

Основная программа:

```

package laba_3.task_4.hospital;

public class Main_Hospital {
    public static void main(String[] args) {

        Hospital hospital = new Hospital();

        Patient pat1 = new Patient("Ivanov Petr Semenovich", 1);
        Patient pat2 = new Patient("Smirnov Ivan Andreevich", 2);
        Patient pat3 = new Patient("Semakov Andrey Ivanovich", 3);

        Staff doc1 = new Staff(1, "Troshin Mikhail Dmitrievich", "Doctor");
        Staff doc2 = new Staff(2, "Mikhailova Maria Maximovna", "Doctor");
        Staff nurse = new Staff(3, "Zimina Natalia Andreevna", "Nurse");

        hospital.addPatient(pat1);
        hospital.addPatient(pat2);
        hospital.addPatient(pat3);

        hospital.addStaff(doc1);
        hospital.addStaff(doc2);
        hospital.addStaff(nurse);

        hospital.setDoctor(pat1.getId(), doc1.getId());
        hospital.setDoctor(pat2.getId(), doc2.getId());
        hospital.setDoctor(pat3.getId(), doc2.getId());

        System.out.println();
        System.out.println("Doctors and patients: ");
        System.out.println(hospital);

        Assignment a1 = new Assignment();
        a1.addDrug("Nurofen");
        a1.addDrug("Sinupret");
        a1.addDrug("Coldrex");
        a1.addDrug("Tantum Verde");
        a1.addProcedure("Inhalation");
        hospital.setAssignment(pat1.getId(), a1);
    }
}

```

```

Assignment a2 = new Assignment();
a2.addSurgery("Laser birthmark removal");
a2.addDrug("Panthenol");
hospital.setAssignment(pat2.getId(), a2);

Assignment a3 = new Assignment();
a3.addProcedure("Back massage");
a3.addProcedure("Swimming pool");
a3.addDrug("Vitamin D");
hospital.setAssignment(pat3.getId(), a3);

System.out.println();
System.out.println("Patients with assignments: ");
System.out.println(hospital);

hospital.completeAssignment(pat1.getId(), doc2.getId());
hospital.completeAssignment(pat2.getId(), nurse.getId());
hospital.completeAssignment(pat3.getId(), nurse.getId());

System.out.println();
System.out.println("Patients with done assignments: ");
System.out.println(hospital);

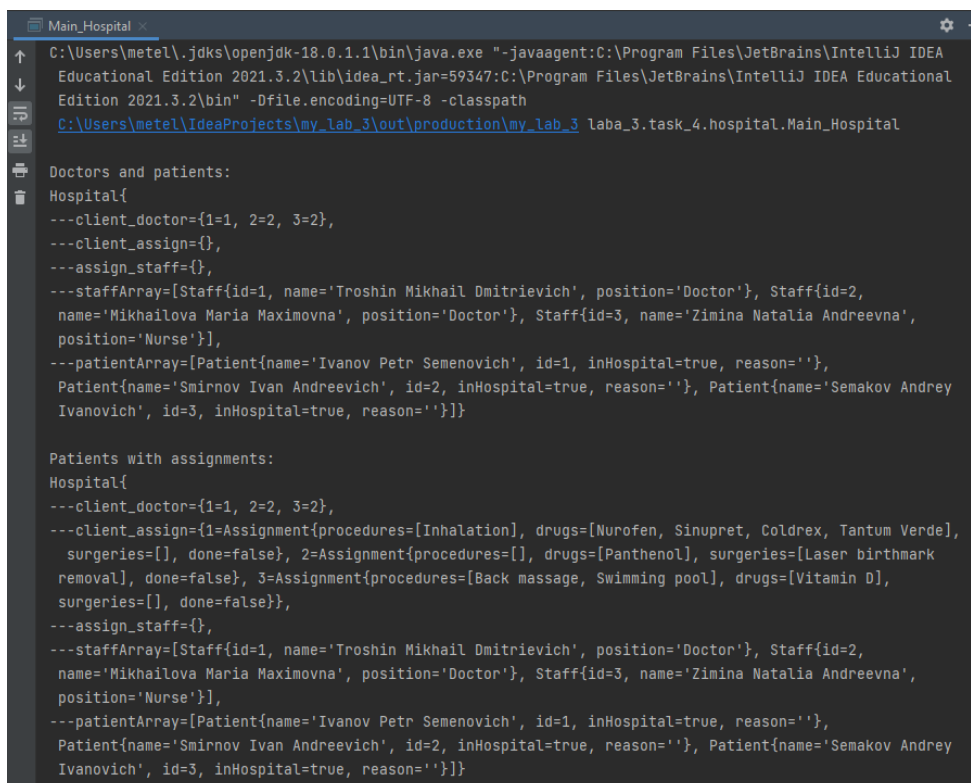
hospital.dismissPatient(pat1, "End of treatment");
hospital.dismissPatient(pat2, "End of treatment");
hospital.dismissPatient(pat3, "Transferred to another department for
treatment");

System.out.println();
System.out.println("Patients are dismissed: ");
System.out.println(hospital);

}
}

```

Результат выполнения подзадачи 1:



```

Main_Hospital
C:\Users\metel\jdk\openjdk-18.0.1.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Educational Edition 2021.3.2\lib\idea_rt.jar=59347:C:\Program Files\JetBrains\IntelliJ IDEA Educational Edition 2021.3.2\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\metel\IdeaProjects\my_lab_3\out\production\my_lab_3 laba_3.task_4.hospital.Main_Hospital

Doctors and patients:
Hospital{
  ---client_doctor={1=1, 2=2, 3=2},
  ---client_assign={},
  ---assign_staff={},
  ---staffArray=[Staff{id=1, name='Troshin Mikhail Dmitrievich', position='Doctor'}, Staff{id=2, name='Mikhailova Maria Maximovna', position='Doctor'}, Staff{id=3, name='Zimina Natalia Andreevna', position='Nurse'}],
  ---patientArray=[Patient{name='Ivanov Petr Semenovich', id=1, inHospital=true, reason=''}, Patient{name='Smirnov Ivan Andreevich', id=2, inHospital=true, reason=''}, Patient{name='Semakov Andrey Ivanovich', id=3, inHospital=true, reason=''}]}

Patients with assignments:
Hospital{
  ---client_doctor={1=1, 2=2, 3=2},
  ---client_assign={1=Assignment{procedures=[Inhalation], drugs=[Nurofen, Sinupret, Coldrex, Tantum Verde], surgeries=[], done=false}, 2=Assignment{procedures=[], drugs=[Panthenol], surgeries=[Laser birthmark removal], done=false}, 3=Assignment{procedures=[Back massage, Swimming pool], drugs=[Vitamin D], surgeries=[], done=false}},
  ---assign_staff={},
  ---staffArray=[Staff{id=1, name='Troshin Mikhail Dmitrievich', position='Doctor'}, Staff{id=2, name='Mikhailova Maria Maximovna', position='Doctor'}, Staff{id=3, name='Zimina Natalia Andreevna', position='Nurse'}],
  ---patientArray=[Patient{name='Ivanov Petr Semenovich', id=1, inHospital=true, reason=''}, Patient{name='Smirnov Ivan Andreevich', id=2, inHospital=true, reason=''}, Patient{name='Semakov Andrey Ivanovich', id=3, inHospital=true, reason=''}]}

```

Листинг выполнения подзадачи 2:

Класс Professor:

```
package laba_3.task_4;
import java.util.Random;

public class Professor {
    private String name;
    private Integer id;
    private Integer plus;

    public Professor() {
    }

    Professor(String name, Integer id) {
        this.name = name;
        this.id = id;
    }

    public void addPlus(Integer plus) {
        this.plus = plus;
    }

    public Integer assess() {
        Random random = new Random();
        Integer mark = random.nextInt(50)+1+plus;
        return mark;
    }
    @Override
    public String toString() {
        return "Professor{" +
            "name='" + name + '\'' +
            ", id=" + id +
            '}';
    }
}
```

Класс Student:

```
package laba_3.task_4;

public class Student {
    private String name;
    private int id;
    public int avg_ex;

    public Student() {
    }

    public Student(String name, int id) {
        this.name = name;
        this.id = id;
    }

    @Override
    public String toString() {
        return "Student{" +
            "name='" + name + '\'' +
            ", id=" + id +
            '}';
    }
}
```

Класс University:

```
package laba_3.task_4;

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

import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

public class University {
    private ArrayList<Faculty> facultyArrayList;
    private Iterator itr;

    public University(){
        this.facultyArrayList = new ArrayList<>();
    }

    public void addFaculty(Faculty faculty){
        this.facultyArrayList.add(faculty);
    }

    public void addProhodnoiUniv(){
        for (int i=0;i<this.facultyArrayList.size();i++){
            this.facultyArrayList.get(i).addProhodnoi();
        }
    }

    public void enteranced_students(){
        for (int i=0;i<this.facultyArrayList.size();i++) {
            this.facultyArrayList.get(i).entered_faculty();
        }
    }

    @Override
    public String toString() {
        return "University:" + "\n" +
            "----All_faculties=" + facultyArrayList + ",\n" +
            '.';
    }
}
```

Класс Faculty:

```
package laba_3.task_4;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Random;

public class Faculty {
    private String name;
    private int id;
    private int prohodnoi;
    private ArrayList<Professor> professorArrayList;
    private ArrayList<Student> studentArrayList;
    public HashMap<Student, Integer> student_mark;
    public ArrayList<Student> real_students;

    public Faculty() {
    }

    public Faculty(String name, Integer id) {
```



```

        this.name = name;
        this.id = id;
        this.professorArrayList = new ArrayList<>();
        this.studentArrayList = new ArrayList<>();
        this.student_mark = new HashMap<>();
        this.real_students = new ArrayList<>();
    }

    public void addProfessor(Professor professor) {
        professorArrayList.add(professor);
    }

    public void addStudent(Student student) {
        this.studentArrayList.add(student);
    }

    public void pass_exams(Student student, Integer mark) {
        System.out.println(student + " " + mark + "\n");
        this.student_mark.put(student, mark);
    }

    public Professor examiner() {
        Random random = new Random();
        return
this.professorArrayList.get(random.nextInt(this.professorArrayList.size()));
    }

    public void addProhodnoi() {
        Integer mark_sum = 0;
        for (int mark : student_mark.values()) mark_sum += mark;
        this.prohodnoi = mark_sum / student_mark.size();
    }

    public Integer prohod_ball() {
        return this.prohodnoi;
    }

    public Integer get_id() {
        return this.id;
    }

    public void entered_faculty() {
        for (Map.Entry entry_sm : student_mark.entrySet()) {
            if (prohodnoi <= (Integer) entry_sm.getValue()) {
                real_students.add((Student) entry_sm.getKey());
            }
        }
    }

    /* public void addSubjects(Integer acc_id, Integer cardNum) {
        account_card.put(acc_id, cardNum);
        card_active.put(cardNum, true);
    }
*/

    @Override
    public String toString() {
        return "Faculty " + name +
            "\n Abiturients = " + studentArrayList +
            "\n Professors =" + professorArrayList +
            "\n Real Students =" + real_students +
            "\n Prohodnoi = " + prohodnoi + "\n ";
    }
}

```

Основная программа Main_Exams:

```
package laba_3.task_4;
//Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает
Экзамены.
// Преподаватель выставляет Оценку. Система подсчитывает средний балл и
// определяет Абитуриентов, зачисленных в учебное заведение.

public class Main_Exams {
    public static void main(String[] args) {

        Student st1 = new Student("Ivanov Ivan Ivanovich", 1);
        Student st2 = new Student("Petrov Petr Petrovich", 2);
        Student st3 = new Student("Smirnov Igor Igorevich", 3);
        Student st4 = new Student("Ivanova Alena Petrovna", 4);
        Student st5 = new Student("Matvienko Lena Petrovovna", 5);
        Student st6 = new Student("Semenov Egor Igorevich", 6);

        Faculty informatics = new Faculty("informatics",1);
        Faculty physics = new Faculty("physics",2);
        Faculty english = new Faculty("english",3);
        Faculty biology = new Faculty("Biology",4);

        University My_university = new University();
        My_university.addFaculty(informatics);
        My_university.addFaculty(physics);
        My_university.addFaculty(english);
        My_university.addFaculty(biology);

        Professor pr1 = new Professor("Vasiliev Kassian Alvianovich", 1);
        Professor pr2 = new Professor("Denisov Kliment Rubenovich", 2);
        Professor pr3 = new Professor("Nikolaev Arsen Artemovich", 3);
        Professor pr4 = new Professor("Davydov Bogdan Germanovich", 4);

        pr1.addPlus(40);
        pr2.addPlus(30);
        pr3.addPlus(20);
        pr4.addPlus(50);

        //prof -> facul
        informatics.addProfessor(pr1);
        informatics.addProfessor(pr2);
        physics.addProfessor(pr2);
        physics.addProfessor(pr3);
        physics.addProfessor(pr4);
        english.addProfessor(pr3);
        english.addProfessor(pr1);
        biology.addProfessor(pr4);

        //st -> facul - facultet.add_st (запись на факультет)

        informatics.addStudent(st1);
        informatics.addStudent(st2);
        informatics.addStudent(st3);
        physics.addStudent(st3);
        physics.addStudent(st4);
        physics.addStudent(st5);
        english.addStudent(st5);
        english.addStudent(st6);
        english.addStudent(st1);
        biology.addStudent(st1);
        biology.addStudent(st2);
        biology.addStudent(st3);
    }
}
```

```

        //st - pass.exams - informatics.(st.get_id, fac.proff_array(1(rand))) -
>return st-mark

informatics.pass_exams(st1, informatics.examenator().assess());
informatics.pass_exams(st2, informatics.examenator().assess());
informatics.pass_exams(st3, informatics.examenator().assess());
physics.pass_exams(st3, physics.examenator().assess());
physics.pass_exams(st4, physics.examenator().assess());
physics.pass_exams(st5, physics.examenator().assess());
english.pass_exams(st5, english.examenator().assess());
english.pass_exams(st6, english.examenator().assess());
english.pass_exams(st1, english.examenator().assess());
biology.pass_exams(st1, biology.examenator().assess());
biology.pass_exams(st2, biology.examenator().assess());
biology.pass_exams(st3, biology.examenator().assess());

My_university.addProhodnoiUniv();

My_university.enteranced_students();

System.out.println(My_university);
}
}

```

Результат выполнения подзадачи 2:

```

University:
---All_faculties=[Faculty informatics
Abiturients = [Student{name='Ivanov Ivan Ivanovich', id=1}, Student{name='Petrov Petr Petrovich', id=2},
Student{name='Smirnov Igor Igorevich', id=3}]
Professors = [Professor{name='Vasiliev Kassian Alvianovich', id=1}, Professor{name='Denisov Kliment
Rubenovich', id=2}]
Real Students = [Student{name='Ivanov Ivan Ivanovich', id=1}, Student{name='Petrov Petr Petrovich', id=2}]
Prohodnoi = 50
, Faculty physics
Abiturients = [Student{name='Smirnov Igor Igorevich', id=3}, Student{name='Ivanova Alena Petrovna',
id=4}, Student{name='Matvienko Lena Petrovovna', id=5}]
Professors = [Professor{name='Denisov Kliment Rubenovich', id=2}, Professor{name='Nikolaev Arsen
Artemovich', id=3}, Professor{name='Davydov Bogdan Germanovich', id=4}]
Real Students = [Student{name='Matvienko Lena Petrovovna', id=5}, Student{name='Smirnov Igor Igorevich',
id=3}]
Prohodnoi = 58
, Faculty english
Abiturients = [Student{name='Matvienko Lena Petrovovna', id=5}, Student{name='Semenov Egor Igorevich',
id=6}, Student{name='Ivanov Ivan Ivanovich', id=1}]
Professors = [Professor{name='Nikolaev Arsen Artemovich', id=3}, Professor{name='Vasiliev Kassian
Alvianovich', id=1}]
Real Students = [Student{name='Semenov Egor Igorevich', id=6}, Student{name='Matvienko Lena Petrovovna',
id=5}]
Prohodnoi = 63
, Faculty Biology
Abiturients = [Student{name='Ivanov Ivan Ivanovich', id=1}, Student{name='Petrov Petr Petrovich', id=2},
Student{name='Smirnov Igor Igorevich', id=3}]
Professors = [Professor{name='Davydov Bogdan Germanovich', id=4}]
Real Students = [Student{name='Ivanov Ivan Ivanovich', id=1}]
Prohodnoi = 84

```

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