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_____

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

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

Отчет

по лабораторной работе №3

Дисциплина: Языки программирования для работы с большими данными.

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

Москва, 2022

Лабораторная работа №3

Задание: Вариант 1

3. 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.

Ход работы: Код программы VectorR3

```
import java.util.ArrayList;

public class VectorR3 {

   private double x1;
   private double x2;
   private double y1;
   private double y2;
   private double z1;
   private double z2;

   private double x;
   private double y;
   private double z;

   public VectorR3(){
   }

   public VectorR3(double x1, double x2, double y1, double y2, double z1, double z2){
      this.x1 = x1;
      this.x2 = x2;
      this.y1 = y1;
      this.y2 = y2;
      this.z1 = z1;
      this.z2 = z2;

      this.x = x2 - x1;
      this.y = y2 - x1;
      this.z = z2 - z1;
   }

   public double getX() {
      return x;
   }

   public double getY() {
      return y;
   }

   public double getZ() {
      return z;
   }
```

```
    public double mod(){
        return Math.sqrt(x*x + y*y + z*z);
    }


    public double scalar(VectorR3 v){ //скалярное произведение
        return this.x * v.x + this.y * v.y + this.z * v.z;
    }

    public boolean is_Orthogonal(VectorR3 v){
        return this.scalar(v) == 0;
    }

    public boolean is_Intersect(VectorR3 v){

        double s = ((this.x2 - v.x2) * this.y + (v.y2 - this.y2) * this.x) / (this.x * v.y- this.y * v.x);
        double t = (s * v.x + this.x2 - v.x2) / this.x;
        double eq1 = this.z1 * t + this.z2 * (1 - t);
        double eq2 = v.z1 * s + v.z2 * (1 - s);

        return eq1 == eq2;
    }

    public void compare(VectorR3 v){
        if (this.mod() > v.mod()){
            System.out.println("Первый больше");
        } else if (v.mod() > this.mod()){
            System.out.println("Второй больше");
        } else {
            System.out.println("Равны");
        }
    }

    @Override
    public String toString() {
        return "VectorR3: " +
                "x1 = " + x1 +
                ", x2 = " + x2 +
                ", y1 = " + y1 +
                ", y2 = " + y2 +
                ", z1 = " + z1 +
                ", z2 = " + z2+
                '.';
    }
}
```

Код программы MainVectorR3

```
import java.util.*;
public class MainVectorR3 {
   public static void isCoplanar(VectorR3 v1, VectorR3 v2, VectorR3 v3) {
      double m = v1.getX() * v2.getY() * v3.getZ() + v1.getY() * v2.getZ() * v3.getX()
            + v1.getZ() * v2.getX() * v3.getY() - v1.getZ() * v2.getY() * v3.getX()
            - v1.getX() * v2.getZ() * v3.getY() - v1.getY() * v2.getX() * v3.getZ();
      if (m == 0) {
         System.out.println("Вектора компланарны");
      } else {
         System.out.println("Вектора не компланарны");
      }
   }
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      int number;
      System.out.print("Введите количество векторов: ");
      if (scanner.hasNextInt()) {
         number = scanner.nextInt();
      } else {
         System.out.println("error");
         number = -1;
      }
      VectorR3[] vector_array = new VectorR3[number];
      Random random = new Random();
      double x1, x2, y1, y2, z1, z2;
      System.out.println("Введите координаты векторов: ");
      for (int i = 0; i < number; i++) {
         x1 = random.nextDouble();
         x2 = random.nextFloat();
         y1 = random.nextFloat();
         y2 = random.nextFloat();
         z1 = random.nextFloat();
         z2 = random.nextFloat();
         vector_array[i] = new VectorR3(x1, x2, y1, y2, z1, z2);
      }

      for (int i = 0; i < number; i++) {
         System.out.println("Вектор " + i + ": " + vector_array[i].toString());
      }

      for (int i = 0; i < number - 2; i++) {
         for (int j = i + 1; j < number - 1; j++) {
            for (int k = j + 1; k < number; k++) {
               System.out.print("Вектора " + i + ", " + j + ", " + k + ": ");
               isCoplanar(vector_array[i], vector_array[j], vector_array[k]);
            }
         }
      }
   }
}
```

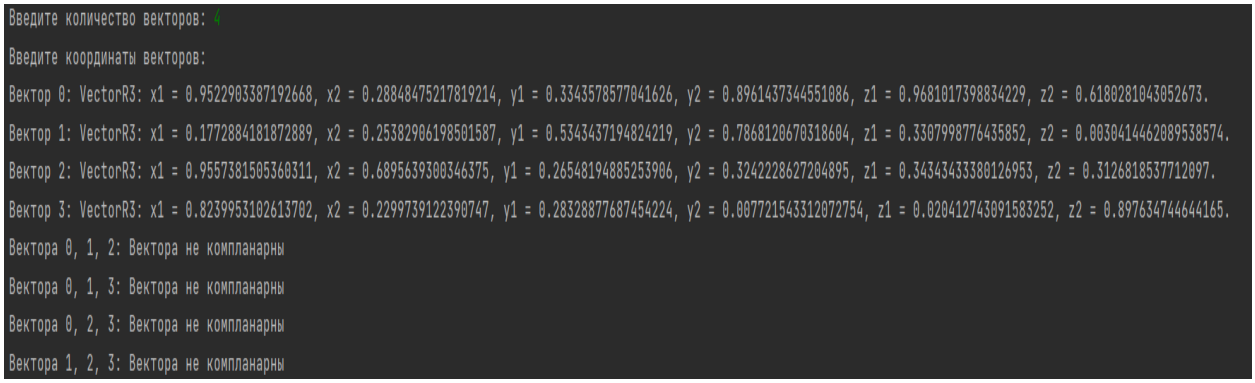

Рисунок 1. Результат работы программы

Задание: Вариант 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

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

Ход работы: Код программы MatrixHandler

```
import java.util.Arrays;
public class MatrixHandler {

   //сложение двух матриц
   public static int[][] add (int[][] a, int[][]b) throws IllegalArgumentException{
      // размерность матриц должна быть одинакова
      if (a.length == b.length && a[0].length == b[0].length){
         int[][] c = new int[a.length][a[0].length];
         for (int i = 0; i < a.length; i++){
            for (int j = 0; j < a[0].length; j++){
               c[i][j] = a[i][j] + b[i][j];
            }
         }

         return c;

      }
      // если матрицы разного размера, бросаем исключение
      else throw new IllegalArgumentException ("Matrix's dimensions should be same");
   }

   //вычитание двух матриц
   public static int[][] substract (int[][] a, int[][]b) throws IllegalArgumentException{
      // размерность матриц должна быть одинакова
      if (a.length == b.length && a[0].length == b[0].length){
```

```
        int[][] c = new int[a.length][a[0].length];
        for (int i = 0; i < a.length; i++){
            for (int j = 0; j < a[0].length; j++){
                c[i][j] = a[i][j] - b[i][j];
            }
        }

        return c;
    }
    // если матрицы разного размера, бросаем исключение
    else throw new IllegalArgumentException ("Matrix's dimensions should be same");
  }

  public static int[][] multiply (int[][] a, int[][]b) throws IllegalArgumentException{
    int l1 = a.length;//m
    int l2 = b[0].length;//n
    int l3 = b.length;//o
    if (a.length == b[0].length){
        int[][] c = new int[a.length][b[0].length];
        for (int i = 0; i < l1; i++) {
            for (int j = 0; j < l2; j++) {
            for (int k = 0; k < l3; k++) {
            c[i][j] += a[i][k] * b[k][j];
                }
            }
        }
        return c;
    }
    else throw new IllegalArgumentException ("Matrix A's columns amount must be same as B's rows
amount");
  }

  public static int[] sumRows(int[][]a)
  {
    int []array = new int[a.length];
    for(int i = 0;i<a.length;i++)
    {
        for(int j = 0;j<a[0].length;j++)
        {
            array[i]+=Math.abs(a[i][j]);
        }
    }
    System.out.println(Arrays.toString(array));
    return array;

  }

  public static int[] sumCols(int [][]a)
  {
    int[]array = new int[a[0].length];
    for(int j = 0;j<a[0].length;j++)
    {
        for(int i =0;i<a.length;i++)
        {
```

```
            array[j]+=Math.abs(a[i][j]);
        }
    }
    System.out.println(Arrays.toString(array));
    return array;

}

public static int maxRows(int []a)
{
    int max = 0;
    for(int val:a) {
        if(val > max)max = val;
    }
    return max;
}

public static int maxCols(int []a)
{
    int min = a[0];
    for(int val:a) {
        if(val > min) min=val;
    }
    return min;
}


//вывод матрицы на экран
public static void print_add(int [][] c){
    System.out.println("Сложение: ");
    for (int i = 0; i < c.length; i++){
        for (int j = 0; j < c[0].length; j++){
            System.out.print(c[i][j] + " ");
        }
        System.out.print("\n");
    }
}

public static void print_substract(int [][] c){
    System.out.println("Вычитание: ");
    for (int i = 0; i < c.length; i++){
        for (int j = 0; j < c[0].length; j++){
            System.out.print(c[i][j] + " ");
        }
        System.out.print("\n");
    }
}
public static void print_multiply(int [][] c){
    System.out.println("Умнажение: ");
    for (int i = 0; i < c.length; i++) {
        for (int j = 0; j < c[0].length; j++) {
            System.out.format(c[i][j] + " ");
        }
        System.out.print("\n");
```

```
    }
  }
  public static void Norm1(int [][] a){
    System.out.println("Norm1 ="+maxRows(sumRows(a)));
  }
  public static void Norm2(int [][] a){
    System.out.println("Norm2 ="+maxCols(sumCols(a)));;
  }

}
```

Код программы Matrix

```
import java.util.Scanner;

public class Matrix {
  public static void main(String[] args) {
    System.out.print("Введите диапозон n: ");
    Scanner scan = new Scanner(System.in);
    int n = Integer.parseInt(scan.nextLine());
    int[][] a = new int[n][n];
    int[][] b = new int[n][n];

    for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
        a[i][j] = (int) (Math.random() * n );
        b[i][j] = (int) (Math.random() * n );
      }
    }

    System.out.println("Матрица A: ");
    for (int i = 0; i < n; i++) {//вывод матрицы
      for (int j = 0; j < n; j++) {
        System.out.print(a[i][j] + " ");
      }
      System.out.print("\n");//переход на новую строчку
    }

    System.out.println("Матрица B: ");
    for (int i = 0; i < n; i++) {//вывод матрицы
      for (int j = 0; j < n; j++) {
        System.out.print(b[i][j] + " ");
      }
      System.out.print("\n");//переход на новую строчку
    }
    int [][] c = new int[n][n];
    MatrixHandler.print_add(MatrixHandler.add(a, b));
    MatrixHandler.print_substract(MatrixHandler.substract(a, b));
    MatrixHandler.print_multiply(MatrixHandler.multiply(a, b));
    MatrixHandler.Norm1(a);
    MatrixHandler.Norm2(a);
    // теперь проверим выбрасвыние исключений
```

```
        try {
            MatrixHandler.add(c, b);
        }
        catch (IllegalArgumentException e) {
            System.out.println("Wrong arguments put into add method");
        }

        try {
            MatrixHandler.multiply(a, b);
        }
        catch (IllegalArgumentException e) {
            System.out.println("Wrong arguments put into multiply method");
        }
    }
}
```

```
Введите диапозон n: 3
Матрица A:
2 2 2
1 2 1
2 2 0
Матрица B:
2 1 0
2 2 1
2 1 0
Сложение:
4 3 2
3 4 2
4 3 0
Вычитание:
0 1 2
-1 0 0
0 1 0
Умнажение:
12 8 2
8 6 2
8 6 2
[6, 4, 4]
Norm1 =6
[5, 6, 3]
Norm2 =6
```

Рисунок 2. Результат работы программы

Задание: Вариант 2. Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

3. Patient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Номер медицинской карты, Диагноз. Создать массив объектов. Вывести: a) список пациентов, имеющих данный диагноз; b) список пациентов, номер медицинской карты у которых находится в заданном интервале.

Ход работы: Код программы 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 "Пациент{" +
```

```
            "id=" + id +
            ", Имя='" + name + '\'' +
            ", Фамилия='" + surname + '\'' +
            ", Отчество='" + lastname + '\'' +
            ", Аддрес='" + address + '\'' +
            ", Телефон='" + phone + '\'' +
            ", № карты=" + cardNumber +
            ", Диагнес='" + diagnosis + '\'' +
            '}';
    }
}
```

Код программы Patient

```
import java.util.ArrayList;

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

        Patient[] patientsArray = createPatientsArray();
        System.out.println("Пациенты:");
        for (Patient p: patientsArray) {
            System.out.println(p);
        }

        Patient[] patientsWithCOVID = chooseByDiagnosis(patientsArray, "COVID-19");
        System.out.println();
        System.out.println("Пациент с COVID-19:");
        for (Patient p: patientsWithCOVID) {
            System.out.println(p);
        }
        Patient[] patientsInRange = chooseByCardNumber(patientsArray, 130, 140);
        System.out.println();
        System.out.println("Пациенты с № карты в диапозоне 130...140:");
        for (Patient p: patientsInRange) {
            System.out.println(p);
        }

    }

    private static Patient[] createPatientsArray(){
        Patient p1 = new Patient(1,"Сергей", "Иванов", "Игоревич", "Дом №12", "8-968-374-26-47", 132,
"Астма");
        Patient p2 = new Patient(2,"Григорий", "Кайдмен", "Петрович", "Дом № 125", "8-969-375-27-
74", 148, "Глоукома");
        Patient p3 = new Patient(3,"Владимир", "Костромин", "Олегович", "Дом № 13/2", "8-977-234-86-
07", 119, "Диабет");
        Patient p4 = new Patient(4,"Алексей", "Миронов", "Александрович", "Дом 5", "8-978-306-36-43",
135, "COVID-19");
        Patient p5 = new Patient(5,"Анна", "Миронова", "Генадевна", "Дом 5", "8-961-333-28-17", 138,
"COVID-19");
        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()]);
    }
}
```

```
Пациенты:
Пациент{id=1, Имя='Сергей', Фамилия='Иванов', Отчество='Игоревич', Аддрес='Дом №12', Телефон='8-968-374-26-47', № карты=132, Диагнес='Астма'}
Пациент{id=2, Имя='Григорий', Фамилия='Кайдмен', Отчество='Петрович', Аддрес='Дом № 125', Телефон='8-969-375-27-74', № карты=148, Диагнес='Глоукома'}
Пациент{id=3, Имя='Владимир', Фамилия='Костромин', Отчество='Олегович', Аддрес='Дом № 13/2', Телефон='8-977-234-86-07', № карты=119, Диагнес='Диабет'}
Пациент{id=4, Имя='Алексей', Фамилия='Миронов', Отчество='Александрович', Аддрес='Дом 5', Телефон='8-978-306-36-43', № карты=135, Диагнес='COVID-19'}
Пациент{id=5, Имя='Анна', Фамилия='Миронова', Отчество='Генадевна', Аддрес='Дом 5', Телефон='8-961-333-28-17', № карты=138, Диагнес='COVID-19'}

Пациент с COVID-19:
Пациент{id=4, Имя='Алексей', Фамилия='Миронов', Отчество='Александрович', Аддрес='Дом 5', Телефон='8-978-306-36-43', № карты=135, Диагнес='COVID-19'}
Пациент{id=5, Имя='Анна', Фамилия='Миронова', Отчество='Генадевна', Аддрес='Дом 5', Телефон='8-961-333-28-17', № карты=138, Диагнес='COVID-19'}

Пациенты с № карты в диапозоне 130...140:
Пациент{id=1, Имя='Сергей', Фамилия='Иванов', Отчество='Игоревич', Аддрес='Дом №12', Телефон='8-968-374-26-47', № карты=132, Диагнес='Астма'}
Пациент{id=4, Имя='Алексей', Фамилия='Миронов', Отчество='Александрович', Аддрес='Дом 5', Телефон='8-978-306-36-43', № карты=135, Диагнес='COVID-19'}
Пациент{id=5, Имя='Анна', Фамилия='Миронова', Отчество='Генадевна', Аддрес='Дом 5', Телефон='8-961-333-28-17', № карты=138, Диагнес='COVID-19'}
```

Рисунок 3. Результат работы программы

Задание: Вариант 2. Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

4. Abiturient: id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки. Создать массив объектов. Вывести: a) список абитуриентов, имеющих неудовлетворительные оценки; b) список абитуриентов, средний балл у которых выше заданного; c) выбрать заданное число n абитуриентов, имеющих самый высокий средний балл (вывести также полный список абитуриентов, имеющих полупроходной балл).

Ход работы: Код программы Abiturient

```
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 +
                '}';
    }
}
```

Код программы MainForAbiturient

```
import java.util.*;

public class MainForAbiturient {

    public static void main(String[] args) {
```

```
        Abiturient a1 = null;
        try {
            a1 = new Abiturient(1, "Иван", "Иванов", "Иванович", "Дом № 5", "8968-374-26-47", new
ArrayList<Integer>(Arrays.asList(3, 2, 5)));
        } catch (Exception e) {
            e.printStackTrace();
        }
        Abiturient a2 = null;
        try {
            a2 = new Abiturient(2, "Petr", "Petrov", "", "House 3", "8-969-375-27-74", new
ArrayList<Integer>(Arrays.asList(4, 4, 5)));
        } catch (Exception e) {
            e.printStackTrace();
        }
        Abiturient a3 = null;
        try {
            a3 = new Abiturient(3, "Dmitry", "Smirnov", "Ivanovich", "House 9", "8-977-234-86-07", new
ArrayList<Integer>(Arrays.asList(5, 0, 5)));
        } catch (Exception e) {
            e.printStackTrace();
        }
        Abiturient a4 = null;
        try {
            a4 = new Abiturient(4, "Ivan", "Smirnov", "Andreevich", "House 5", "8-978-306-36-43", new
ArrayList<Integer>(Arrays.asList(3, 2, 4)));
        } catch (Exception e) {
            e.printStackTrace();
        }
        Abiturient a5 = null;
        try {
            a5 = new Abiturient(5, "Alexander", "Ivanov", "Ilich", "House 11", "8-961-333-28-17", new
ArrayList<Integer>(Arrays.asList(5, 5, 5)));
        } catch (Exception e) {
            e.printStackTrace();
        }

        Abiturient[] abiturientsArray = new Abiturient[]{a1, a2, a3, a4, a5};

        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[] chooseWithNeuds(Abiturient[] abiturientsArray) {
    ArrayList<Abiturient> newAbiturientsArray = new ArrayList<>();
    for (int i = 0; i < abiturientsArray.length; i++) {
        try {
            if (abiturientsArray[i].getMarks().contains(2)) {
                newAbiturientsArray.add(abiturientsArray[i]);
            }
        } catch (Exception e) {
        }
    }
    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++) {
        try {
            float avg = 0;
            for (Integer m : abiturientsArray[i].getMarks()) {
                avg += m;
            }
            avg = avg / abiturientsArray[i].getMarks().size();
            if (avg > mark) {
                newAbiturientsArray.add(abiturientsArray[i]);
            }
        } catch (Exception e) {
        }
    }
    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++) {
        try {
            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]);
            }
        } catch (Exception e) {
        }
    }
    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()]);
  }
}
```

```
Abiturients with neuds:
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=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]}], 5.0=[Abiturient{id=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=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5', phone='8-978-306-36-43', marks=[3, 2, 4]}
```

Рисунок 4. Результат работы программы

Задание: Вариант 3. Создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

3. Создать объект класса Одномерный массив, используя класс Массив. Методы: создать, вывести на консоль, выполнить операции (сложить, вычесть, перемножить).

Ход работы: Код программы One_Dim_Array

```
import java.util.ArrayList;
import java.util.Objects;

public class One_Dim_Array extends Array{

   private ArrayList<Integer> data;

   public One_Dim_Array(){
   }

   public One_Dim_Array(int n){
      this.data = new ArrayList<>(n);
   }

   public One_Dim_Array(ArrayList<Integer> data){
      this.data = data;
   }

   @Override
   public void print() {
      System.out.println("Массив: " + this);
   }

   @Override
   public One_Dim_Array sum(Object obj) {
      One_Dim_Array array = new One_Dim_Array(this.data.size());
      if (obj.getClass().equals(this.getClass())){
         if (this.data.size() != ((One_Dim_Array) obj).data.size()){
            System.out.println("ошибка, несоответствие размеров");
         } else {
            for (int i = 0; i < this.data.size(); i++) {
               array.data.add(this.data.get(i) + ((One_Dim_Array) obj).data.get(i));
            }
         }
      } else {
         System.out.println("ошибка, несоответствие классов");
      }
      return array;
   }

   @Override
   public One_Dim_Array multi(Object obj) {
      One_Dim_Array array = new One_Dim_Array(this.data.size());
      if (obj.getClass().equals(this.getClass())){
         if (this.data.size() != ((One_Dim_Array) obj).data.size()){
            System.out.println("ошибка, несоответствие размеров");
         } else {
            for (int i = 0; i < this.data.size(); i++) {
               array.data.add(this.data.get(i)*((One_Dim_Array) obj).data.get(i));
            }
         }
      } else {
```

```
            System.out.println("ошибка, несоответствие классов");
        }
        return array;
    }

    @Override
    public One_Dim_Array diff(Object obj) {
        One_Dim_Array array = new One_Dim_Array(this.data.size());
        if (obj.getClass().equals(this.getClass())){
            if (this.data.size() != ((One_Dim_Array) obj).data.size()){
                System.out.println("ошибка, несоответствие размеров");
            } else {
                for (int i = 0; i < this.data.size(); i++) {
                    array.data.add(this.data.get(i) - ((One_Dim_Array) obj).data.get(i));
                }
            }
        } else {
            System.out.println("ошибка, несоответствие классов");
        }
        return array;
    }

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

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

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

Код программы Main_For_One_Dim_Array

```
import java.util.*;
public class Main_For_One_Dim_Array {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        int number;
        System.out.print("Введите количество элементов: ");
        if (scanner.hasNextInt()) {
            number = scanner.nextInt();
```

```
        } else {
            System.out.println("error");
            number = -1;
        }

        ArrayList<Integer> input_data = new ArrayList<>(number);
        System.out.print("Введите элементы: ");
        for (int i = 0; i < number; i++){
            if (scanner.hasNextInt()) {
                input_data.add(scanner.nextInt());
            } else {
                System.out.println("error");
            }
        }
        One_Dim_Array array = new One_Dim_Array(input_data);

        ArrayList<Integer> input_data_second = new ArrayList<>(number);
        System.out.print("Введите второй массив: ");
        for (int i = 0; i < number; i++){
            if (scanner.hasNextInt()) {
                input_data_second.add(scanner.nextInt());
            } else {
                System.out.println("error");
            }
        }
        One_Dim_Array second_array = new One_Dim_Array(input_data_second);
        System.out.println(array + " " + second_array);

        System.out.println("Сумма: " + array.sum(second_array));
        System.out.println("Разность: " + array.diff(second_array));
        System.out.println("Произведение: " + array.multi(second_array));
    }
}
```

```
Введите количество элементов: 3
Введите элементы: 1 2 3
Введите второй массив: 1 2 3
One_Dim_Array{data=[1, 2, 3]} One_Dim_Array{data=[1, 2, 3]}
Сумма: One_Dim_Array{data=[2, 4, 6]}
Разность: One_Dim_Array{data=[0, 0, 0]}
Произведение: One_Dim_Array{data=[1, 4, 9]}
```

Рисунок 5. Результат работы программы

Задание: Вариант 3. Создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

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

Ход работы: Код программы Number

```
import java.util.Objects;

public class Number {

   private final int number;

   public Number(){
      this.number = 0;
   }

   public Number(int number){
      this.number = number;
   }

   public int getNumber() {
      return number;
   }

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

   public Object sum(Object obj){
      if (obj.getClass().equals(this.getClass())){
         Number number = new Number(((Number) obj).number + this.number);
      } else {
         System.out.println("ошибка, несоответствие классов");
      }
      return number;
   }

   public Object diff(Object obj){
      if (obj.getClass().equals(this.getClass())){
         Number number = new Number(this.number - ((Number) obj).number);
      } else {
         System.out.println("ошибка, несоответствие классов");
      }
      return number;
   }
   public Object multi(Object obj){
      if (obj.getClass().equals(this.getClass())){
         Number number = new Number(((Number) obj).number * this.number);
      } else {
         System.out.println("ошибка, несоответствие классов");
      }
      return number;
   }
```

```
    public Object div(Object obj){
        if (obj.getClass().equals(this.getClass())){
            Number number = new Number(this.number/((Number) obj).number);
        } else {
            System.out.println("ошибка, несоответствие классов");
        }
        return number;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Number number1 = (Number) o;
        return number == number1.number;
    }

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

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

Код программы Number

```
import java.util.*;
public class Main_For_Simple_Fraction {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        int numerator, denominator;
        System.out.print("Введите числитель: ");
        if (scanner.hasNextInt()) {
            numerator = scanner.nextInt();
        } else {
            System.out.println("error");
            numerator = -1;
        }
        System.out.print("Введите знаменатель: ");
        if (scanner.hasNextInt()) {
            denominator = scanner.nextInt();
        } else {
            System.out.println("error");
            denominator = -1;
        }
```

```
        Simple_Fraction first_fraction = new Simple_Fraction(numerator, denominator);

        System.out.print("Введите числитель: ");
        if (scanner.hasNextInt()) {
            numerator = scanner.nextInt();
        } else {
            System.out.println("error");
            numerator = -1;
        }
        System.out.print("Введите знаменатель: ");
        if (scanner.hasNextInt()) {
            denominator = scanner.nextInt();
        } else {
            System.out.println("error");
            denominator = -1;
        }
        Simple_Fraction second_fraction = new Simple_Fraction(numerator, denominator);

        first_fraction.print();
        second_fraction.print();
        System.out.println("Сложение:");
        ((Simple_Fraction) first_fraction.sum(second_fraction)).print();
        System.out.println("Вычетание:");
        ((Simple_Fraction) first_fraction.diff(second_fraction)).print();
        System.out.println("Умножение:");
        ((Simple_Fraction) first_fraction.multi(second_fraction)).print();
        System.out.println("Деление:");
        ((Simple_Fraction) first_fraction.div(second_fraction)).print();


    }
}
```

```
Введите числитель: 1
Введите знаменатель: 2
Введите числитель: 1
Введите знаменатель: 5
Дробь{числитель=1, знаменатель=2}
Дробь{числитель=1, знаменатель=5}
Сложение:
Дробь{числитель=7, знаменатель=10}
Вычетание:
Дробь{числитель=3, знаменатель=10}
Умножение:
Дробь{числитель=1, знаменатель=10}
Деление:
Дробь{числитель=5, знаменатель=2}
```

Рисунок 6. Результат работы программы

Задание: Вариант 4.Построить модель программной системы.

3. 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.

Ход работы: Код программы 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<Patient1> 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 addPatient1(Patient1 patient1){
      this.patientArray.add(patient1);
   }

   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 dismissPatient1(Patient1 patient1, String reason){
      int pat_id = patient1.getId();
      patient1.dismiss(reason);
```

```
        this.client_doctor.remove(pat_id);
    }

    @Override
          public String toString() {
        return "Больница{" + "\n" +
                "---Доктора=" + client_doctor + ",\n" +
                "---Назначения=" + client_assign + ",\n" +
                "---Назначения персоналу=" + assign_staff + ",\n" +
                "---Персонал=" + staffArray + ",\n" +
                "---Пациенты=" + patientArray +
                '}';
    }
}
```

Код программы Patient1

```
import java.util.Objects;
public class Patient1 {
    private String name;
    private int id;
    private boolean inHospital;
    private String reason;

    public Patient1() {
    }

    public Patient1(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;
        Patient1 patient1 = (Patient1) o;
        return id == patient1.id && inHospital == patient1.inHospital && Objects.equals(name,
patient1.name) && Objects.equals(reason, patient1.reason);
    }

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

    @Override
    public String toString() {
        return "Пациент{" +
                "Имя='" + name + '\'' +
                ", id=" + id +
                ", В Больнице=" + inHospital +
                ", причина='" + reason + '\'' +
                '}';
    }
}
```

Код программы Staff

```
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 "Персонал{" +
                "id=" + id +
                ", Имя='" + name + '\'' +
```

```
            ", Должность='" + position + '\'' +
            '}';
    }
}
```

Код программы MainForHospital

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

        Hospital hospital = new Hospital();

        Patient1 pat1 = new Patient1("Сергей Иванов Игоревич", 1);
        Patient1 pat2 = new Patient1("Григорий Кайдмен Петрович", 2);
        Patient1 pat3 = new Patient1("Владимир Костромин Олегович", 3);


        Staff doc1 = new Staff(1, "Семёнов Михаил Александрович", "Доктор");
        Staff doc2 = new Staff(2, "Воронова Мария Юрьевна", "Доктор");
        Staff nurse = new Staff(3, "Самойлова Ирина Виталевна", "Медсестра");

        hospital.addPatient1(pat1);
        hospital.addPatient1(pat2);
        hospital.addPatient1(pat3);


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

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


        System.out.println();
        System.out.println("Доктор и пациент: ");
        System.out.println(hospital);

        Assignment a1 = new Assignment();
        a1.addDrug("Нурофен");
        a1.addDrug("Ибуклин");
        a1.addDrug("Ситоцыл");
        a1.addDrug("Колдакт");
        a1.addProcedure("Ингаляция");
        hospital.setAssignment(pat1.getId(), a1);

        Assignment a2 = new Assignment();
        a2.addSurgery("Заменя швов");
        a2.addDrug("Пантенол");
        hospital.setAssignment(pat2.getId(), a2);

        Assignment a3 = new Assignment();
```

```
        a3.addProcedure("Массаж спины");
        a3.addProcedure("Плавательный бассейн");
        a3.addDrug("Витамин Д");
        hospital.setAssignment(pat3.getId(), a3);

        System.out.println();
        System.out.println("Пациенты с назначениями: ");
        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("Пациенты с выполненными назначениями: ");
        System.out.println(hospital);

        hospital.dismissPatient1(pat1, "Окончание лечения");
        hospital.dismissPatient1(pat2, "Окончание лечения");
        hospital.dismissPatient1(pat3, "Переведен на лечение в другое отделение");


        System.out.println();
        System.out.println("Пациенты выписаны: ");
        System.out.println(hospital);

    }
}
```

```
Доктор и пациент:
Больница{
---Доктора={1=1, 2=2},
---Назначения={},
---Назначения персоналу={},
---Персонал=[Персонал{id=1, Имя='Семёнов Михаил Александрович', Должность='Доктор'}, Персонал{id=2, Имя='Воронова Мария Юрьевна', Должность='Доктор'}, Персонал{id=3, Имя='Самойло
---Пациенты=[Пациент{Имя='Сергей Иванов Игоревич', id=1, В Больнице=true, причина=''}, Пациент{Имя='Григорий Кайдмен Петрович', id=2, В Больнице=true, причина=''}, Пациент{Имя='

Пациенты с назначениями:
Больница{
---Доктора={1=1, 2=2},
---Назначения={1=Назначение{Процедуры=[Ингаляция], Лекарства=[Нурофен, Ибуклин, Ситоцыл, Колдакт], Операции=[], Выполнение=false}, 2=Назначение{Процедуры=[], Лекарства=[Пантенол]
---Назначения персоналу={},
---Персонал=[Персонал{id=1, Имя='Семёнов Михаил Александрович', Должность='Доктор'}, Персонал{id=2, Имя='Воронова Мария Юрьевна', Должность='Доктор'}, Персонал{id=3, Имя='Самойло
---Пациенты=[Пациент{Имя='Сергей Иванов Игоревич', id=1, В Больнице=true, причина=''}, Пациент{Имя='Григорий Кайдмен Петрович', id=2, В Больнице=true, причина=''}, Пациент{Имя='

Пациенты с выполненными назначениями:
Больница{
---Доктора={1=1, 2=2},
---Назначения={1=Назначение{Процедуры=[Ингаляция], Лекарства=[Нурофен, Ибуклин, Ситоцыл, Колдакт], Операции=[], Выполнение=true}, 2=Назначение{Процедуры=[], Лекарства=[Пантенол],
---Назначения персоналу={Назначение{Процедуры=[Массаж спины, Плавательный бассейн], Лекарства=[Витамин Д], Операции=[], Выполнение=true}=3, Назначение{Процедуры=[Ингаляция], Лека
---Персонал=[Персонал{id=1, Имя='Семёнов Михаил Александрович', Должность='Доктор'}, Персонал{id=2, Имя='Воронова Мария Юрьевна', Должность='Доктор'}, Персонал{id=3, Имя='Самойло
---Пациенты=[Пациент{Имя='Сергей Иванов Игоревич', id=1, В Больнице=true, причина=''}, Пациент{Имя='Григорий Кайдмен Петрович', id=2, В Больнице=true, причина=''}, Пациент{Имя='
```

Рисунок 7. Результат работы программы

Задание: Вариант 4.Построить модель программной системы.

4. Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает Экзамены. Преподаватель выставляет Оценку. Система

подсчитывает средний балл и определяет Абитуриентов, зачисленных в учебное заведение.

Ход работы: Код программы Abiturient

```
import java.util.Objects;

public class Abiturient {
   private int id;
   private String name;
   private String status;
   // private boolean inHospital;

   //private String Evaluation;

   public Abiturient() {
   }

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


   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 getStatus() {return status;}
   public void setStatus(String status) {
      this.status = status;
   }



   @Override
   public boolean equals(Object o) {
```

```
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Abiturient abiturient = (Abiturient) o;
        return id == abiturient.id  &&  Objects.equals(name, abiturient.name) &&  Objects.equals(status,
abiturient.status) ;
    }


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

    @Override
    public String toString() {
        return "\n"+"Абитуриент: " +
                ", id=" + id + '\''  + ' '+
                "Имя='" + name + '\'' + ' '+
                "Статус='" + status + '\'' + ' '
                ;

    }


}
```

Код программы Assignment

```
import java.util.ArrayList;
import java.util.Objects;

public class Assignment {
    private ArrayList<String> Evaluation;
    private ArrayList<String> SR;
    private ArrayList<String> Result;
    private boolean done;

    public Assignment() {
        this.Evaluation = new ArrayList<>();
        this.SR = new ArrayList<>();
        this.Result = new ArrayList<>();
        this.done = false;
    }

    public void addEvaluation(String oz){
        this.Evaluation.add(oz);
    }

    public void addSR(String sr){
        this.SR.add(sr);
    }
```

```
    public void addResult(String res){
        this.Result.add(res);
    }

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

    @Override
    public String toString() {
        return  ",\n" +
                "Оценки " + Evaluation +
                ", Средний бал =" + SR +
                ", Решение: " + Result

                ;
    }

    @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(Evaluation, that.Evaluation) && Objects.equals(SR,
that.SR) && Objects.equals(Result, that.Result);
    }

    @Override
    public int hashCode() {
        return Objects.hash(Evaluation,SR, Result, done);
        //drugs, surgeries, done
    }
}
```

Код программы Faculty

```
import java.util.ArrayList;
import java.util.HashMap;
public class Faculty {
        private HashMap<Integer, Integer> client_doctor;
        private HashMap<Integer, Assignment> client_assign;
        private HashMap<Assignment, Integer> assign_teachers;
        private ArrayList<Teachers> teachersArray;
        private ArrayList<Abiturient> abiturientArray;

        public Faculty() {
            this.client_doctor = new HashMap<>();
            this.client_assign = new HashMap<>();
            this.assign_teachers = new HashMap<>();
            this.teachersArray = new ArrayList<>();
            this.abiturientArray = new ArrayList<>();
        }
```

```
    public void addAbiturient(Abiturient abiturient){
       this.abiturientArray.add(abiturient);
    }

    public void addTeachers(Teachers teachers){
       this.teachersArray.add(teachers);
    }
    public void setTS(int ab_id, int ts_id){
       this.client_doctor.put(ab_id, ts_id);
    }

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


    public void completeAssignment(int ab_id, int ts_id){
       this.assign_teachers.put(this.client_assign.get(ab_id), ts_id);
      this.client_assign.get(ab_id).complete();
    }



    @Override
    public String toString() {
       return  "\n"+"Экзамен{" + "\n" +
             "---Прикрипление=" + client_doctor + ",\n" +
             "---Проверяющие " + teachersArray + ",\n" +
             "---Абитуриенты " + abiturientArray + ",\n"+
             "---Результаты  " + client_assign + ",\n"
             ;
    }
  }
```

Код программы Teachers

```
import java.util.Objects;

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

    public Teachers() {
    }

    public Teachers(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;
      Teachers teachers = (Teachers) o;
      return id == teachers.id && Objects.equals(name, teachers.name) && Objects.equals(position,
teachers.position);
    }

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

    @Override
    public String toString() {
      return "\n"+ "Проверяющий: " + ' '+
            "id=" + id + '\''+' '+
            ", Имя='" + name + '\'' + ' '+
            ", Должность='" + position + '\'' +' '
            ;
    }
  }
```

Код программы Main

```
public class Main {

  public static void main(String[] args) {

    Faculty faculty = new Faculty();

    Abiturient ab1 = new Abiturient(1, "Сергей Иванов Игоревич", "Зарегестрирован" );
    Abiturient ab2 = new Abiturient(2,"Григорий Кайдмен Петрович","Зарегестрирован");
    Abiturient ab3 = new Abiturient(3, "Владимир Костромин Олегович", "Зарегестрирован");


    Teachers ts1 = new Teachers(1, "Семёнов Михаил Александрович", "Преподователь");
    Teachers ts2 = new Teachers(2, "Воронова Мария Юрьевна", "Преподователь");
    Teachers ts3 = new Teachers(3, "Воронова Мария Юрьевна", "Преподователь");
```

```
        faculty.addAbiturient(ab1);
        faculty.addAbiturient(ab2);
        faculty.addAbiturient(ab3);


        faculty.addTeachers(ts1);
        faculty.addTeachers(ts2);
        faculty.addTeachers(ts3);

        faculty.setTS(ab1.getId(), ts1.getId());
        faculty.setTS(ab2.getId(), ts2.getId());
        faculty.setTS(ab3.getId(), ts3.getId());

        Assignment a1 = new Assignment();
        a1.addEvaluation("Экзамен 1 - 4");
        a1.addEvaluation("Экзамен 2 - 4");
        a1.addEvaluation("Экзамен 3 - 4");
        a1.addSR("4");
        a1.addResult("Зачислен");
        faculty.setAssignment(ab1.getId(), a1);

        Assignment a2 = new Assignment();
        a2.addEvaluation("Экзамен 1 - 3");
        a2.addEvaluation("Экзамен 2 - 4");
        a2.addEvaluation("Экзамен 3 - 3");
        a2.addSR("3,3");
        a2.addResult("Незачислен");
        faculty.setAssignment(ab2.getId(), a2);

        Assignment a3 = new Assignment();
        a3.addEvaluation("Экзамен 1 - 5");
        a3.addEvaluation("Экзамен 2 - 5");
        a3.addEvaluation("Экзамен 3 - 5");
        a3.addSR("5");
        a3.addResult("Зачислен");
        faculty.setAssignment(ab3.getId(), a3);

        System.out.println();
        System.out.println(faculty);

        faculty.completeAssignment(ab1.getId(), ts1.getId());
        faculty.completeAssignment(ab2.getId(), ts2.getId());
        faculty.completeAssignment(ab3.getId(), ts3.getId());

    }

}
```

```
Экзамен{
---Прикрипление={1=1, 2=2, 3=3},
---Проверяющие [
Проверяющий:  id=1' , Имя='Семёнов Михаил Александрович' , Должность='Преподователь' ,
Проверяющий:  id=2' , Имя='Воронова Мария Юрьевна' , Должность='Преподователь' ,
Проверяющий:  id=3' , Имя='Воронова Мария Юрьевна' , Должность='Преподователь' ],
---Абитуриенты [
Абитуриент: , id=1' Имя='Сергей Иванов Игоревич' Статус='Зарегестрирован' ,
Абитуриент: , id=2' Имя='Григорий Кайдмен Петрович' Статус='Зарегестрирован' ,
Абитуриент: , id=3' Имя='Владимир Костромин Олегович' Статус='Зарегестрирован' ],
---Результаты  {1=,
Оценки [Экзамен 1 - 4, Экзамен 2 - 4, Экзамен 3 - 4], Средний бал =[4], Решение: [Зачислен], 2=,
Оценки [Экзамен 1 - 3, Экзамен 2 - 4, Экзамен 3 - 3], Средний бал =[3,3], Решение: [Незачислен], 3=,
Оценки [Экзамен 1 - 5, Экзамен 2 - 5, Экзамен 3 - 5], Средний бал =[5], Решение: [Зачислен]},
```

Рисунок 8. Результат работы программы

Вывод: лабораторная работа была выполнена в соответствие с заданием и полученные верные результаты работ программ