



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

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01 Информатика и вычислительная техника

МАГИСТЕРСКАЯ ПРОГРАММА 09.04.01/12 Интеллектуальный анализ больших
данных в системах поддержки принятия решений.

О Т Ч Е Т

по лабораторной работе № 3

Вариант № 11

Название: Классы, наследование, полиморфизм

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

С.В.Мельников

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2023

Цель работы

Изучить основные принципы ООП языка программирования Java.

Задание 1 (Вариант 1, Задание 1):

Определить класс Вектор размерности n . Реализовать методы сложения, вычитания, умножения, инкремента, декремента, индексирования. Определить массив из m объектов. Каждую из пар векторов передать в методы, возвращающие их скалярное произведение и длины. Вычислить и вывести углы между векторами.

Листинг программы:

Код класса Matix:

```
public class Matrix {
    private final int[][] values;
    private final int rows;
    private final int columns;

    public Matrix(int n) {
        this.rows = n;
        this.columns = n;
        this.values = new int[n][n];
        for (int i = 0; i < n; ++i)
            for (int j = 0; j < n; j++)
                this.values[i][j] = 0;
    }

    public Matrix(int n, int m) {
        this.rows = m;
        this.columns = n;
        this.values = new int[m][n];
    }

    public Matrix(int[][] values) {
        this.values = new int[values.length][];
        this.rows = values.length;
        this.columns = this.rows != 0 ? values[0].length : 0;
        for (int i = 0; i < values.length; i++)
            this.values[i] = values[i].clone();
    }

    public Matrix add(Matrix matrix) {
        int[][] new_values = new int[this.rows][this.columns];
        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                new_values[i][j] = this.values[i][j] + matrix.values[i][j];
        return new Matrix(new_values);
    }

    public Matrix subtract(Matrix matrix) {
        int[][] new_values = new int[this.rows][this.columns];
```

```

        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                new_values[i][j] = this.values[i][j] - matrix.values[i][j];
        return new Matrix(new_values);
    }

    public Matrix multiply(Matrix matrix) {
        int[][] new_values = new int[this.rows][matrix.columns];
        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                for (int k = 0; k < matrix.columns; k++)
                    new_values[i][k] += this.values[i][j] *
matrix.values[j][k];
        return new Matrix(new_values);
    }

    public Matrix multiply(int coefficient) {
        int[][] new_values = new int[this.rows][this.columns];
        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                new_values[i][j] = this.values[i][j] * coefficient;
        return new Matrix(new_values);
    }

    public void increment() {
        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                this.values[i][j] += 1;
    }

    public void decrement() {
        for (int i = 0; i < this.rows; i++)
            for (int j = 0; j < this.columns; j++)
                this.values[i][j] -= 1;
    }

    public int get_element(int row, int column) {
        return this.values[row][column];
    }

    @Override
    public String toString() {
        StringBuilder result = new StringBuilder();
        for (int i = 0; i < this.rows; i++) {
            for (int j = 0; j < this.columns; j++)
                result.append(this.values[i][j]).append(" ");
            result.append("\n");
        }
        return result.toString();
    }

    public static int scalarProduct(Matrix a, Matrix b) {
        int result = 0;
        for (int i = 0; i < a.rows; i++)
            result += a.values[i][0] * b.values[i][0];
        return result;
    }

    public static double angle(Matrix a, Matrix b) {

```

```

        double scalarProduct = 0;
        int a_part = 0;
        int b_part = 0;
        for (int i = 0; i < a.rows; i++) {
            scalarProduct += a.values[i][0] * b.values[i][0];
            a_part += pow(a.values[i][0], 2);
            b_part += pow(b.values[i][0], 2);
        }
        if (a_part == 0 || b_part == 0)
            return 0;
        return Math.acos(scalarProduct / (Math.sqrt(a_part) *
Math.sqrt(b_part))) * 180 / Math.PI;
    }

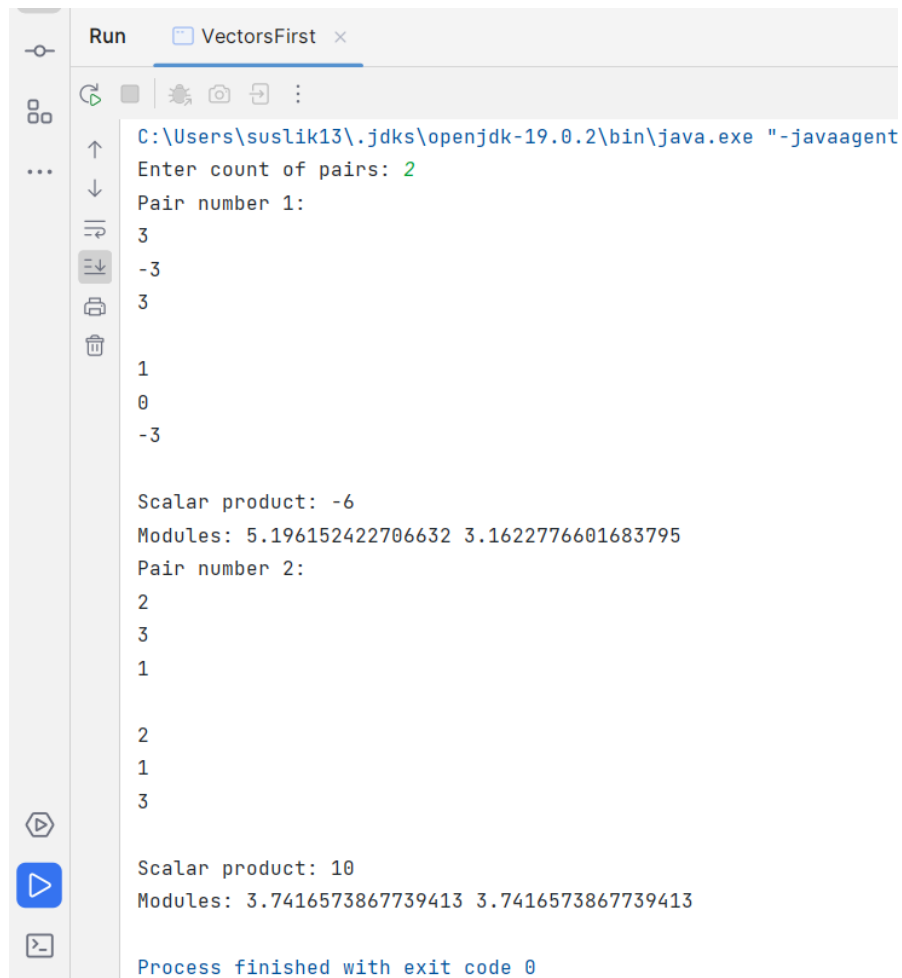
    public static double module(Matrix a) {
        int result = 0;
        for (int i = 0; i < a.rows; i++)
            result += pow(a.values[i][0], 2);
        return Math.sqrt(result);
    }

    public static boolean isCollinear(Matrix a, Matrix b) {
        if (a.rows == 0 || b.rows == 0)
            return true;
        boolean result = true;
        double coefficient = a.values[0][0] * 1.0 / b.values[0][0];
        for (int i = 1; i < a.rows; i++)
            result = result && (Math.abs(a.values[i][0] * 1.0 /
b.values[i][0] - coefficient)) < 0.00001;
        return result;
    }

    public static boolean isOrthogonal(Matrix a, Matrix b) {
        return Matrix.scalarProduct(a, b) == 0;
    }
}

```

Работа программы представлена на рисунке 1.



```
Run VectorsFirst x
C:\Users\suslik13\.jdk\openjdk-19.0.2\bin\java.exe "-javaagent
Enter count of pairs: 2
Pair number 1:
3
-3
3
1
0
-3

Scalar product: -6
Modules: 5.196152422706632 3.1622776601683795
Pair number 2:
2
3
1
2
1
3

Scalar product: 10
Modules: 3.7416573867739413 3.7416573867739413

Process finished with exit code 0
```


Рисунок 1 – Работа программы VectorsFirst.java

Задание 2 (Вариант 1, Задание 2):

Определить класс Вектор размерности n . Определить несколько конструкторов. Реализовать методы для вычисления модуля вектора, скалярного произведения, сложения, вычитания, умножения на константу. Объявить массив объектов. Написать метод, который для заданной пары векторов будет определять, являются ли они коллинеарными или ортогональными.

Листинг программы общий с первым заданием.

Работа программы представлена на рисунке 2.



```
Run VectorsSecond x
C:\Users\suslik13\.jdk\openjdk-19.0.2\bin\java.exe "-javaagent
Enter count of pairs: 2
Pair number 1:
3
3
1
1
3
-2
3
Is collinear? : false
Is orthogonal?: false
Pair number 2:
0
-1
-3
2
3
-3
Is collinear? : false
Is orthogonal?: false
Process finished with exit code 0
```

Рисунок 2 – Работа программы VectorsSecond.java

Задание 3 (Вариант 2, Задание 1):

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.

Student: id, Фамилия, Имя, Отчество, Дата рождения, Адрес, Телефон, Факультет, Курс, Группа. Создать массив объектов. Вывести: а) список студентов заданного факультета; б) списки студентов для каждого факультета и курса; с) список студентов, родившихся после заданного года; d) список учебной группы.

Листинг программы:

Код класса Student:

```
public class Student {
    private int id;
    private String surname;
    private String name;
    private String patronymic;
    private Date birthday;
    private String address;
    private String phone;
    private String faculty;
    private int course;
    private int group;

    public Student(int id, String surname, String name, String patronymic, Date
birthday, String address, String phone, String faculty, int course, int group) {
        this.id = id;
        this.surname = surname;
        this.name = name;
        this.patronymic = patronymic;
        this.birthday = birthday;
        this.address = address;
        this.phone = phone;
        this.faculty = faculty;
        this.course = course;
        this.group = group;
    }

    public int getId() {
        return id;
    }

    public String getSurname() {
        return surname;
    }

    public String getName() {
        return name;
    }

    public String getPatronymic() {
        return patronymic;
    }

    public Date getBirthday() {
        return birthday;
    }

    public String getAddress() {
        return address;
    }

    public String getPhone() {
        return phone;
    }

    public String getFaculty() {
        return faculty;
    }

    public int getCourse() {
        return course;
    }

    public int getGroup() {
```

```

        return group;
    }

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

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

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

    public void setPatronymic(String patronymic) {
        this.patronymic = patronymic;
    }

    public void setBirthday(Date birthday) {
        this.birthday = birthday;
    }

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

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

    public void setFaculty(String faculty) {
        this.faculty = faculty;
    }

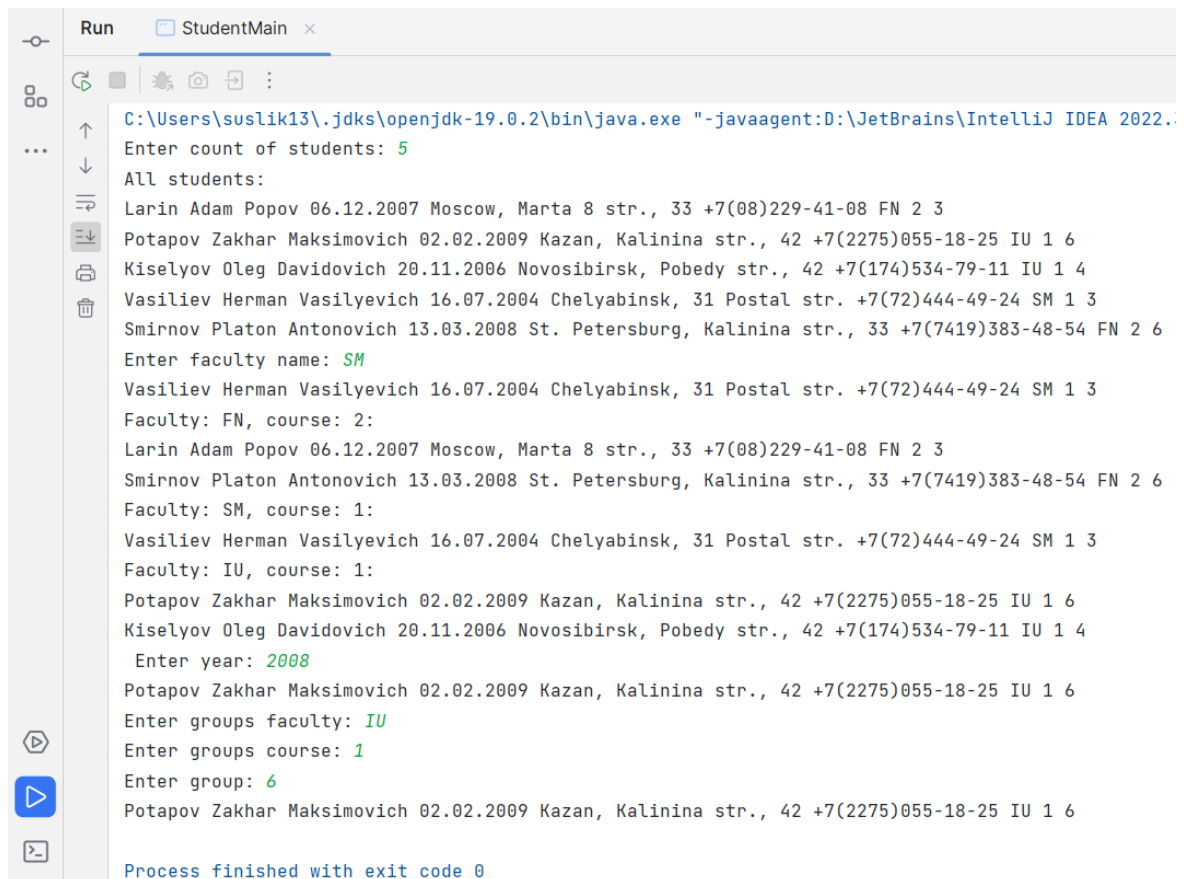
    public void setCourse(int course) {
        this.course = course;
    }

    public void setGroup(int group) {
        this.group = group;
    }

    @Override
    public String toString() {
        SimpleDateFormat format = new SimpleDateFormat("dd.MM.yyyy");
        return surname + " " + name + " " + patronymic + " " +
format.format(birthday) + " " +
        address + " " + phone + " " + faculty + " " + course + " " +
group;
    }
}

```

Работа программы представлена на рисунке 3.



```
Run StudentMain x
C:\Users\suslik13\jdk\openjdk-19.0.2\bin\java.exe "-javaagent:D:\JetBrains\IntelliJ IDEA 2022.1\lib\idea_rt.jar=1213:D:\JetBrains\IntelliJ IDEA 2022.1\bin" -Dfile.encoding=UTF-8
Enter count of students: 5
All students:
Larin Adam Popov 06.12.2007 Moscow, Marta 8 str., 33 +7(08)229-41-08 FN 2 3
Potapov Zakhar Maksimovich 02.02.2009 Kazan, Kalinina str., 42 +7(2275)055-18-25 IU 1 6
Kiselyov Oleg Davidovich 20.11.2006 Novosibirsk, Pobedy str., 42 +7(174)534-79-11 IU 1 4
Vasiliev Herman Vasilyevich 16.07.2004 Chelyabinsk, 31 Postal str. +7(72)444-49-24 SM 1 3
Smirnov Platon Antonovich 13.03.2008 St. Petersburg, Kalinina str., 33 +7(7419)383-48-54 FN 2 6
Enter faculty name: SM
Vasiliev Herman Vasilyevich 16.07.2004 Chelyabinsk, 31 Postal str. +7(72)444-49-24 SM 1 3
Faculty: FN, course: 2:
Larin Adam Popov 06.12.2007 Moscow, Marta 8 str., 33 +7(08)229-41-08 FN 2 3
Smirnov Platon Antonovich 13.03.2008 St. Petersburg, Kalinina str., 33 +7(7419)383-48-54 FN 2 6
Faculty: SM, course: 1:
Vasiliev Herman Vasilyevich 16.07.2004 Chelyabinsk, 31 Postal str. +7(72)444-49-24 SM 1 3
Faculty: IU, course: 1:
Potapov Zakhar Maksimovich 02.02.2009 Kazan, Kalinina str., 42 +7(2275)055-18-25 IU 1 6
Kiselyov Oleg Davidovich 20.11.2006 Novosibirsk, Pobedy str., 42 +7(174)534-79-11 IU 1 4
Enter year: 2008
Potapov Zakhar Maksimovich 02.02.2009 Kazan, Kalinina str., 42 +7(2275)055-18-25 IU 1 6
Enter groups faculty: IU
Enter groups course: 1
Enter group: 6
Potapov Zakhar Maksimovich 02.02.2009 Kazan, Kalinina str., 42 +7(2275)055-18-25 IU 1 6
Process finished with exit code 0
```

Рисунок 3 – Работа программы StudentMain.java

Задание 4 (Вариант 2, Задание 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.

Customer: id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Номер банковского счета. Создать массив объектов. Вывести: а) список покупателей в алфавитном порядке; б) список покупателей, у которых номер кредитной карточки находится в заданном интервале.

Листинг программы:

Код класса Customer:

```
public class Customer implements Comparable<Customer> {
    private int id;
    private String name;
    private String surname;
```

```

private String patronymic;
private String address;
private String card_number;
private String account_number;

public int getId() {
    return id;
}

public Customer(int id, String name, String surname, String patronymic,
String address, String card_number, String account_number) {
    this.id = id;
    this.name = name;
    this.surname = surname;
    this.patronymic = patronymic;
    this.address = address;
    this.card_number = card_number;
    this.account_number = account_number;
}

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 getPatronymic() {
    return patronymic;
}

public void setPatronymic(String patronymic) {
    this.patronymic = patronymic;
}

@Override
public String toString() {
    return "Customer " + id + ": " +
        surname + " " + name + " " + patronymic +
        ", card: " + card_number + ", account: " + account_number +
        ", " + address;
}

public String getAddress() {
    return address;
}

```

```

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

public String getCard_number() {
    return card_number;
}

public void setCard_number(String card_number) {
    this.card_number = card_number;
}

public String getAccount_number() {
    return account_number;
}

public void setAccount_number(String account_number) {
    this.account_number = account_number;
}

@Override
public int compareTo(Customer o) {
    return (this.surname + this.name +
this.patronymic).compareTo(o.surname + o.name + o.patronymic);
}
}

```

Работа программы представлена на рисунке 4.

```

Run CustomerMain x
C:\Users\suslik13\jdk\openjdk-19.0.2\bin\java.exe "-javaagent:D:\JetBrains\IntelliJ IDEA 2022.3.2\lib\idea_rt.jar=61366:D:\J...
Enter count of customers: 4
Generated customers:
Customer 0: Demyanov Ali Alekseevich, card: 2892876383870426, account: 27676212043255799569, Togliatti, Shosseynaya str., 12
Customer 1: Krasnov Arsen Maksimovich, card: 9366486159341312, account: 37280393607540326010, Togliatti, Shosseynaya str., 12
Customer 2: Sharov Marcel Mironovich, card: 5447467837577684, account: 69766346007286625032, Volgograd, Truda street, 43
Customer 3: Sokolo Demid Ibrahimovich, card: 8754570215590259, account: 50877088488899389231, Krasnoyarsk, Michurina str., 49
Sorted customers:
Customer 0: Demyanov Ali Alekseevich, card: 2892876383870426, account: 27676212043255799569, Togliatti, Shosseynaya str., 12
Customer 1: Krasnov Arsen Maksimovich, card: 9366486159341312, account: 37280393607540326010, Togliatti, Shosseynaya str., 12
Customer 2: Sharov Marcel Mironovich, card: 5447467837577684, account: 69766346007286625032, Volgograd, Truda street, 43
Customer 3: Sokolo Demid Ibrahimovich, card: 8754570215590259, account: 50877088488899389231, Krasnoyarsk, Michurina str., 49
Enter lower card number: 4444444444444444
Enter upper card number: 9999999999999999
Customer 1: Krasnov Arsen Maksimovich, card: 9366486159341312, account: 37280393607540326010, Togliatti, Shosseynaya str., 12
Customer 2: Sharov Marcel Mironovich, card: 5447467837577684, account: 69766346007286625032, Volgograd, Truda street, 43
Customer 3: Sokolo Demid Ibrahimovich, card: 8754570215590259, account: 50877088488899389231, Krasnoyarsk, Michurina str., 49
Process finished with exit code 0

```

Рисунок 4 – Работа программы CustomerMain.java

Задание 5 (Вариант 3, Задание 11):

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

Создать объект класса Сутки, используя классы Час, Минута. Методы: вывести на консоль текущее время, рассчитать время суток (утро, день, вечер, ночь).

Листинг программы:

Код класса Minute:

```
public class Minute {
    private final int minute;

    public Minute(int minute) {
        this.minute = minute % 60;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Minute minute1 = (Minute) o;
        return minute == minute1.minute;
    }

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

    @Override
    public String toString() {
        return "" + minute;
    }

    public int getMinute() {
        return minute;
    }
}
```

Код класса Hour:

```
public class Hour {
    private final int hour;

    public Hour(int hour) {
        this.hour = hour % 24;
    }
}
```

```

    }

    public boolean equals(Object object) {
        if (this == object) return true;
        if (object == null || getClass() != object.getClass()) return
false;
        if (!super.equals(object)) return false;
        Hour hour1 = (Hour) object;
        return hour == hour1.hour;
    }

    public int hashCode() {
        return Objects.hash(super.hashCode(), hour);
    }

    @Override
    public String toString() {
        return "" + hour;
    }

    public int getHour() {
        return hour;
    }
}

```

Код класса Day:

```

public class Day {
    private final Hour hours;
    private final Minute minute;

    public Day(int hours, int minute) {
        this.hours = new Hour(hours);
        this.minute = new Minute(minute);
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Day day = (Day) o;
        return Objects.equals(hours, day.hours) && Objects.equals(minute,
day.minute);
    }

    @Override
    public int hashCode() {
        return Objects.hash(hours, minute);
    }

    @Override
    public String toString() {
        return "Time: " + hours + ":" + minute;
    }

    public static String now() {
        Date dateNow = new Date();
        SimpleDateFormat formatForDateNow = new SimpleDateFormat("hh:mm");
        return formatForDateNow.format(dateNow);
    }
}

```

```

    }

    public String timeOfDay() {
        int time = hours.getHour() * 60 + minute.getMinute();
        if (time < 6 * 60) {
            return "night";
        } else if (time < 12 * 60) {
            return "morning";
        } else if (time < 18 * 60) {
            return "afternoon";
        } else {
            return "evening";
        }
    }
}

```

Работа программы представлена на рисунке 5.



Рисунок 5 – Работа программы DayMain.java

Задание 6 (Вариант 3, Задание 1):

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

Создать объект класса Текстовый файл, используя класс Файл. Методы: создать, переименовать, вывести на консоль содержимое, дополнить, удалить.

Листинг программы:

Код класса MyFile:

```

public class MyFile {
    protected String name;
    protected final String format;

    public MyFile(String file) {
        String[] parse = file.split("\\.");
        this.name = parse[0];
        this.format = parse[1];
    }

    public void create() {
        try {
            new File(this.name + "." + this.format).createNewFile();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    public void rename(String new_name) {
        File f1 = new File(this.name + "." + this.format);
        File f2 = new File(new_name + "." + this.format);
        f1.renameTo(f2);
        this.name = new_name;
    }

    public void delete() {
        new File(this.name + "." + this.format).delete();
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        MyFile myFile = (MyFile) o;
        return Objects.equals(name, myFile.name) && Objects.equals(format,
myFile.format);
    }

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

    @Override
    public String toString() {
        return "File: " + name + '.' + format;
    }
}

```

Код класса TextFile:

```

public class TextFile extends MyFile{
    public TextFile(String name) {
        super(name + ".txt");
    }

    public void printFile() {
        try (BufferedReader br = new BufferedReader(new
FileReader(this.name + "." + this.format))) {
            String line = br.readLine();

```

```

        while (line != null) {
            System.out.println(line);
            line = br.readLine();
        }
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

public void addText(String text) {
    try(FileWriter writer = new FileWriter(this.name + "." +
this.format, true)) {
        writer.write(text);
        writer.append('\n');
        writer.flush();
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

@Override
public boolean equals(Object o) {
    return super.equals(o);
}

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

@Override
public String toString() {
    return super.toString();
}
}

```

Работа программы представлена на рисунке 6.



Рисунок 6 – Работа программы FileMain.java

Задание 7 (Вариант 4, Задание 1):

Построить модель программной системы.

Система Факультатив. Преподаватель объявляет запись на Курс. Студент записывается на Курс, обучается и по окончании Преподаватель выставляет Оценку, которая сохраняется в Архиве. Студентов, Преподавателей и Курсов при обучении может быть несколько.

Листинг программы:

Код класса Teacher:

```
public class Teacher {
    private final String name;

    public Teacher(String name) {
        this.name = name;
    }

    public Course announce_course(String course_title) {
        System.out.println(this.name + " announced course: " +
course_title);
        return new Course(course_title, this);
    }

    public void rate(Student student, Course course, int mark, Archive
archive) {
        archive.writeRecord(course, student, mark);
    }

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

Код класса Course:

```
public class Course {

    private final String title;
    private final Teacher teacher;
    private final ArrayList<Student> students;

    public Course(String title, Teacher teacher) {
        this.title = title;
        this.teacher = teacher;
        this.students = new ArrayList<>();
    }

    public String getTitle() {
        return title;
    }
}
```

```

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

    public void study() {
        System.out.println(this.teacher.getName() + " teaches subject " +
this.title);
        for (Student student: students) {
            System.out.println(student.getName() + ": done");
        }
    }
}

```

Код класса Student:

```

public class Student {
    private final String name;

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

    public void goToCourse(Course course) {
        System.out.println("Student " + name + " enrolled in a course " +
course.getTitle());
        course.addStudent(this);
    }

    public String getName() {
        return name;
    }
}

```

Код класса Archive:

```

public class Archive {
    private final HashMap<Course, HashMap<Student, Integer>> store;

    public Archive() {
        this.store = new HashMap<>();
    }

    public void writeRecord(Course course, Student student, int mark) {
        if (store.containsKey(course)) {
            if (store.get(course).containsKey(student)) {
                store.get(course).replace(student, mark);
            } else {
                store.get(course).put(student, mark);
            }
        } else {
            HashMap<Student, Integer> students = new HashMap<>();
            students.put(student, mark);
            store.put(course, students);
        }
    }

    public void printTable() {
        for (Map.Entry<Course, HashMap<Student, Integer>> entry:

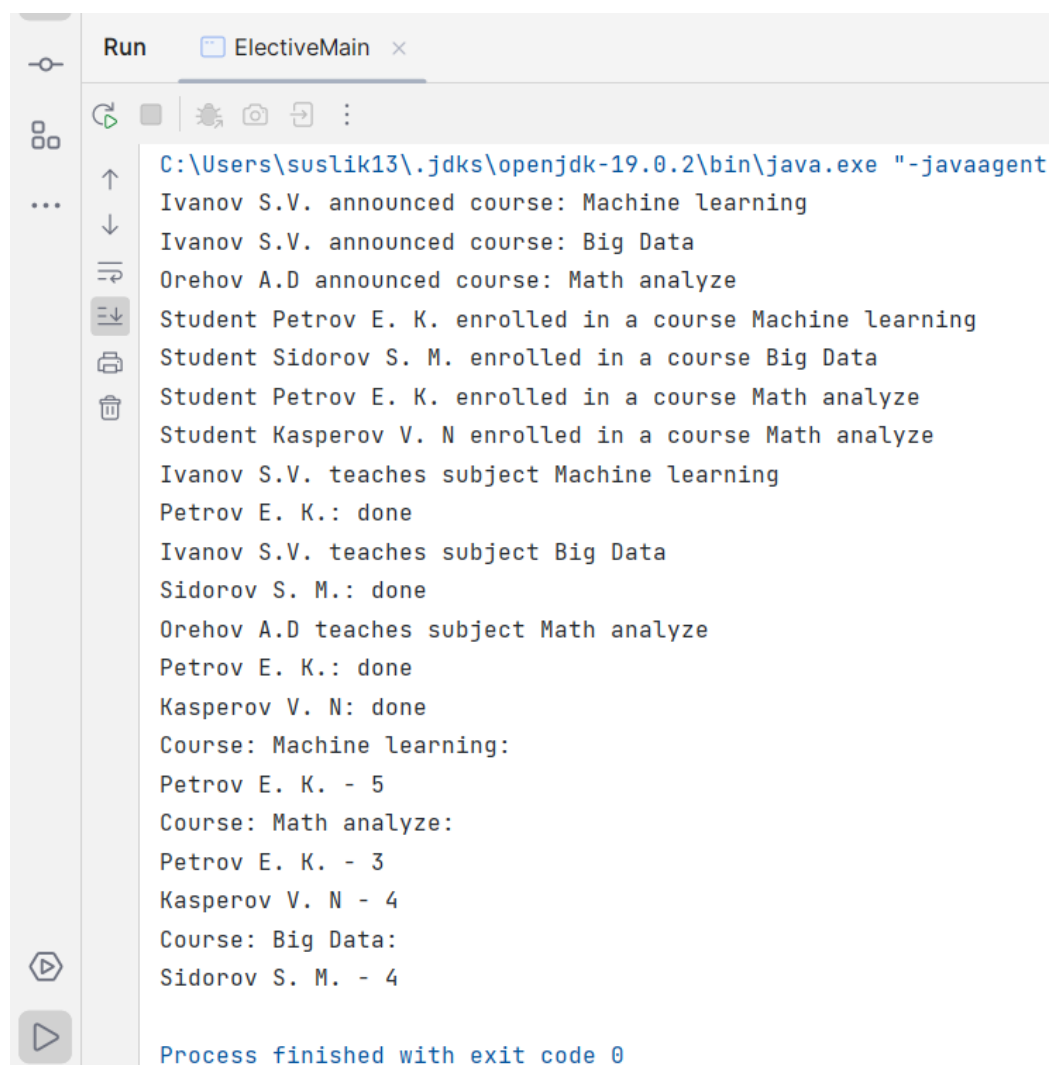
```

```

store.entrySet()) {
    String course_name = entry.getKey().getTitle();
    HashMap<Student, Integer> students_map = entry.getValue();
    System.out.println("Course: " + course_name + ":");
    for (Map.Entry<Student, Integer> students:
students_map.entrySet()) {
        String student = students.getKey().getName();
        int mark = students.getValue();
        System.out.println(student + " - " + mark);
    }
}
}
}

```

Работа программы представлена на рисунке 7.



```

Run ElectiveMain x
C:\Users\suslik13\jdk\openjdk-19.0.2\bin\java.exe "-javaagent
Ivanov S.V. announced course: Machine learning
Ivanov S.V. announced course: Big Data
Orehov A.D announced course: Math analyze
Student Petrov E. K. enrolled in a course Machine learning
Student Sidorov S. M. enrolled in a course Big Data
Student Petrov E. K. enrolled in a course Math analyze
Student Kasperov V. N enrolled in a course Math analyze
Ivanov S.V. teaches subject Machine learning
Petrov E. K.: done
Ivanov S.V. teaches subject Big Data
Sidorov S. M.: done
Orehov A.D teaches subject Math analyze
Petrov E. K.: done
Kasperov V. N: done
Course: Machine learning:
Petrov E. K. - 5
Course: Math analyze:
Petrov E. K. - 3
Kasperov V. N - 4
Course: Big Data:
Sidorov S. M. - 4
Process finished with exit code 0

```

Рисунок 7 – Работа программы ElectiveMain.java

Задание 8 (Вариант 4, Задание 2):

Построить модель программной системы.

Система Платежи. Клиент имеет Счет в банке и Кредитную Карту (КК). Клиент может оплатить Заказ, сделать платеж на другой Счет, заблокировать КК и аннулировать Счет. Администратор может заблокировать КК за превышение кредита.

Листинг программы:

Код класса Account:

```
public class Account {
    private final String account_number;
    private int funds;

    public Account(String account_number, int funds) {
        this.account_number = account_number;
        this.funds = funds;
    }

    public int getFunds() {
        return funds;
    }

    public void setFunds(int funds) {
        int prev = this.funds;
        this.funds = funds;
        int curr = this.funds;
        if (prev >= curr) {
            System.out.println(this.account_number + ": Payment has been
made. current funds: " + this.funds);
        } else {
            System.out.println(this.account_number + ": Payment received.
current funds: " + this.funds);
        }
    }

    public void cancel() {
        this.funds = 0;
        System.out.println(this.account_number + ": Account canceled");
    }
}
```

Код класса Admin:

```
public class Admin {
    public void blockCreditCard(CreditCard card) {
        System.out.println("Admin blocked card " + card.getCardNumber());
        card.setIsBlocked(true);
    }
}
```

Код класса Client:

```
public class Client {
    private final String name;
    private final Account account;
    private final CreditCard credit_card;

    public Client(String name, Account account, CreditCard credit_card) {
        this.name = name;
        this.account = account;
        this.credit_card = credit_card;
    }

    public void payOrder(Order order) {
        if (this.credit_card.getIsBlocked()) {
            System.out.println(this.name + ": Your account is blocked!");
        } else {
            System.out.println("Order " + order.getName() + " payed");
            this.account.setFunds(account.getFunds() - order.getCost());
        }
    }

    public void payAnotherAccount(Account account, int amount) {
        if (this.credit_card.getIsBlocked()) {
            System.out.println(this.name + ": Your account is blocked!");
        } else {
            this.account.setFunds(this.account.getFunds() - amount);
            account.setFunds(account.getFunds() + amount);
        }
    }

    public void blockCard() {
        this.credit_card.setIsBlocked(true);
    }

    public void cancelAccount() {
        this.account.cancel();
    }
}
```

Код класса CreditCard:

```
public class CreditCard {
    private final String card_number;
    private boolean is_blocked;

    public CreditCard(String card_number) {
        this.card_number = card_number;
        this.is_blocked = false;
    }

    public boolean getIsBlocked() {
        return is_blocked;
    }

    public void setIsBlocked(boolean is_blocked) {
        this.is_blocked = is_blocked;
        if (this.is_blocked) {
            System.out.println(this.card_number + ": now blocked");
        } else {
            System.out.println(this.card_number + ": now unblocked");
        }
    }
}
```

```

    }

    public String getCardNumber() {
        return card_number;
    }
}

```

Код класса Order:

```

public class Order {
    private final String name;
    private final int cost;

    public Order(String name, int cost) {
        this.name = name;
        this.cost = cost;
    }

    public int getCost() {
        return cost;
    }

    public String getName() {
        return name;
    }
}

```

Работа программы представлена на рисунке 8.

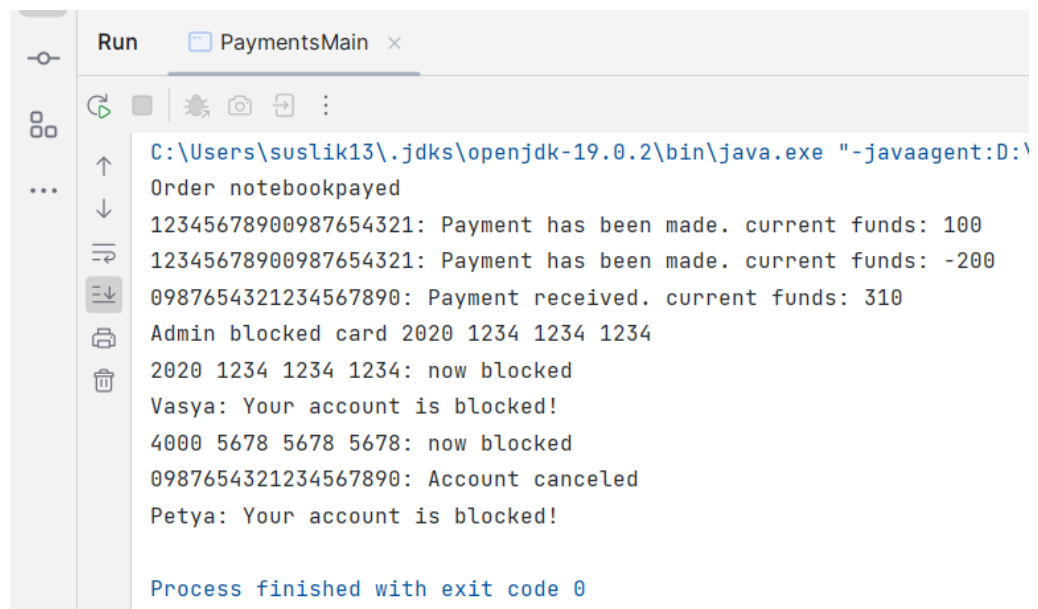


Рисунок 8 – Работа программы PaymentsMain.java

Вывод

В ходе выполнения лабораторной работы были изучены и освоены основные принципы ООП языка программирования Java.