



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

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

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

О Т Ч Е Т

по лабораторной работе № 5

Вариант 12

Название: Работа с исключениями и файлами

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

П.А. Мартынюк

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2022

Цель работы:

Получение навыков обработки исключений в Java и навыков работы с файлами в Java.

Выполнение:

Задание 1:

Выполнить задания на основе варианта 1 лабораторной работы 3, контролируя состояние потоков ввода/вывода. При возникновении ошибок, связанных с корректностью выполнения математических операций, генерировать и обрабатывать исключительные ситуации. Предусмотреть обработку исключений, возникающих при нехватке памяти, отсутствии требуемой записи (объекта) в файле, недопустимом значении поля и т.д.

1. 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.
2. 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.

Листинг выполнения подзадачи 1 (файл Vector.java)

```
package task1;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Objects;

public class Vector {

    private ArrayList<Integer> vector;
    private int dimension;

    public Vector() {
        this.vector = new ArrayList<Integer>();
        this.dimension = 0;
    }
}
```

```

public Vector(ArrayList<Integer> vector) {
    this.vector = vector;
    this.dimension = this.vector.size();
}

public Vector(Integer[] vector) {
    this.vector = new ArrayList<>(Arrays.asList(vector));
    this.dimension = this.vector.size();
}

public void addElement(int element){
    this.vector.add(element);
    this.dimension++;
}

public ArrayList<Integer> getVector() {
    return vector;
}

public void setVector(ArrayList<Integer> vector) {
    this.vector = vector;
}

public int getDimension() {
    return dimension;
}

public void setDimension(int dimension) {
    this.dimension = dimension;
}

@Override
public String toString() {
    return "Vector{" +
        "vector=" + vector +
        ", dimension=" + dimension +
        '}';
}

public double calculateModulus(){
    int sumOfSquares = 0;
    for (Integer elem: this.vector) {
        sumOfSquares += Math.pow(elem, 2);
    }
    return Math.sqrt(sumOfSquares);
}

public int scalarProduct(Vector other){
    try {
        int sumOfMuls = 0;
        int dimation = this.dimension > other.dimension ? this.dimension :
other.dimension;
        for (int i = 0; i < dimation; i++) {
            sumOfMuls += this.vector.get(i) * other.vector.get(i);
        }
        System.out.println("Scalar product of vectors: " + sumOfMuls);
        return sumOfMuls;
    } catch (Exception e) {
        System.out.println("scalarProduct exception : Different dimensions!");
        return -1;
    }
}

public Vector vectorSummation(Vector other){
    try {
        int dimation = this.dimension > other.dimension ? this.dimension :
other.dimension;
        Vector temp = new Vector();
        for (int i = 0; i < dimation; i++) {
            temp.addElement(this.vector.get(i) + other.vector.get(i));
        }
        System.out.println("Summation of vectors: " + temp);
    }
}

```

```

        return temp;
    } catch (Exception e) {
        System.out.println("vectorSummation exception : Different dimensions!");
        return new Vector();
    }
}

public Vector vectorSubtraction(Vector other) {
    try {
        int dimation = this.dimension > other.dimension ? this.dimension :
other.dimension;
        Vector temp = new Vector();
        for (int i = 0; i < dimation; i++) {
            temp.addElement(this.vector.get(i) - other.vector.get(i));
        }
        System.out.println("Subtraction of vectors: " + temp);
        return temp;
    } catch (Exception e) {
        System.out.println("vectorSubtraction exception : Different dimensions!");
        return new Vector();
    }
}

public Vector constantMultiplication(int k){
    Vector temp = new Vector();
    for (int i = 0; i < this.dimension; i++) {
        temp.addElement(this.vector.get(i) * k);
    }
    return temp;
}

public boolean isCollinear(Vector other){
    try {
        ArrayList<Float> tempArray = new ArrayList<Float>();
        int dimation = this.dimension > other.dimension ? this.dimension :
other.dimension;
        for (int i = 0; i < dimation; i++) {
            if (this.vector.get(i) == 0) {
                if (other.vector.get(i) == 0) {
                    continue;
                } else {
                    return false;
                }
            } else {
                try {
                    tempArray.add((float) this.vector.get(i) / other.vector.get(i));
                } catch (Exception e) {
                    System.out.println("isCollinear exception : Division by zero!");
                }
            }
        }
        boolean isCollinear = true;
        for (int i = 0; i < tempArray.size() - 1; i++) {
            if (!Objects.equals(tempArray.get(i), tempArray.get(i + 1))) {
                isCollinear = false;
                return isCollinear;
            }
        }
        return isCollinear;
    } catch (Exception e) {
        System.out.println("isCollinear exception : Different dimensions!");
        return false;
    }
}

public boolean isOrthogonal(Vector other){
    try {
        if (scalarProduct(other) == 0) {
            return true;
        } else {
            return false;
        }
    }
}

```

```

    }
    } catch (Exception e) {
        System.out.println("isOrthogonal exception : Different dimensions!");
        return false;
    }
}

public void defineVectorsStatus(Vector other) {
    try {
        if (this.isCollinear(other)) {
            System.out.println("Vectors are collinear");
        } else if (this.isOrthogonal(other)) {
            System.out.println("Vectors are orthogonal");
        } else {
            System.out.println("Vectors are neither orthogonal nor collinear");
        }
    } catch (Exception e) {
        System.out.println("defineVectorsStatus exception : Different dimensions!");
    }
}
}

```

Листинг выполнения подзадачи 1 (файл MainForVector.java)

```

package task1;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;

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

        Scanner in = new Scanner(System.in);
        Vector vec1;
        Vector vec2;

        try {
            System.out.print("Input vector 1 length: ");
            int n1 = in.nextInt();
            ArrayList<Integer> v1 = new ArrayList<>();
            System.out.println("Input elements for vector 1:");
            for (int i = 0; i < n1; i++) {
                int elem = in.nextInt();
                v1.add(elem);
            }
            vec1 = new Vector(v1);
        } catch (Exception e) {
            in.nextLine();
            System.out.println("Wrong input for vector 1!");
            vec1 = new Vector();
        }

        try {
            System.out.print("Input vector 2 length: ");
            int n2 = in.nextInt();
            ArrayList<Integer> v2 = new ArrayList<>();
            System.out.println("Input elements for vector 2:");
            for (int i = 0; i < n2; i++) {
                int elem = in.nextInt();
                v2.add(elem);
            }
            vec2 = new Vector(v2);
        } catch (Exception e) {
            in.nextLine();
            System.out.println("Wrong input for vector 2!");
            vec2 = new Vector();
        }

        System.out.println("Modulus of first vector: " + vec1.calculateModulus());
        System.out.println("Modulus of second vector: " + vec2.calculateModulus());

        vec1.scalarProduct(vec2);
    }
}

```

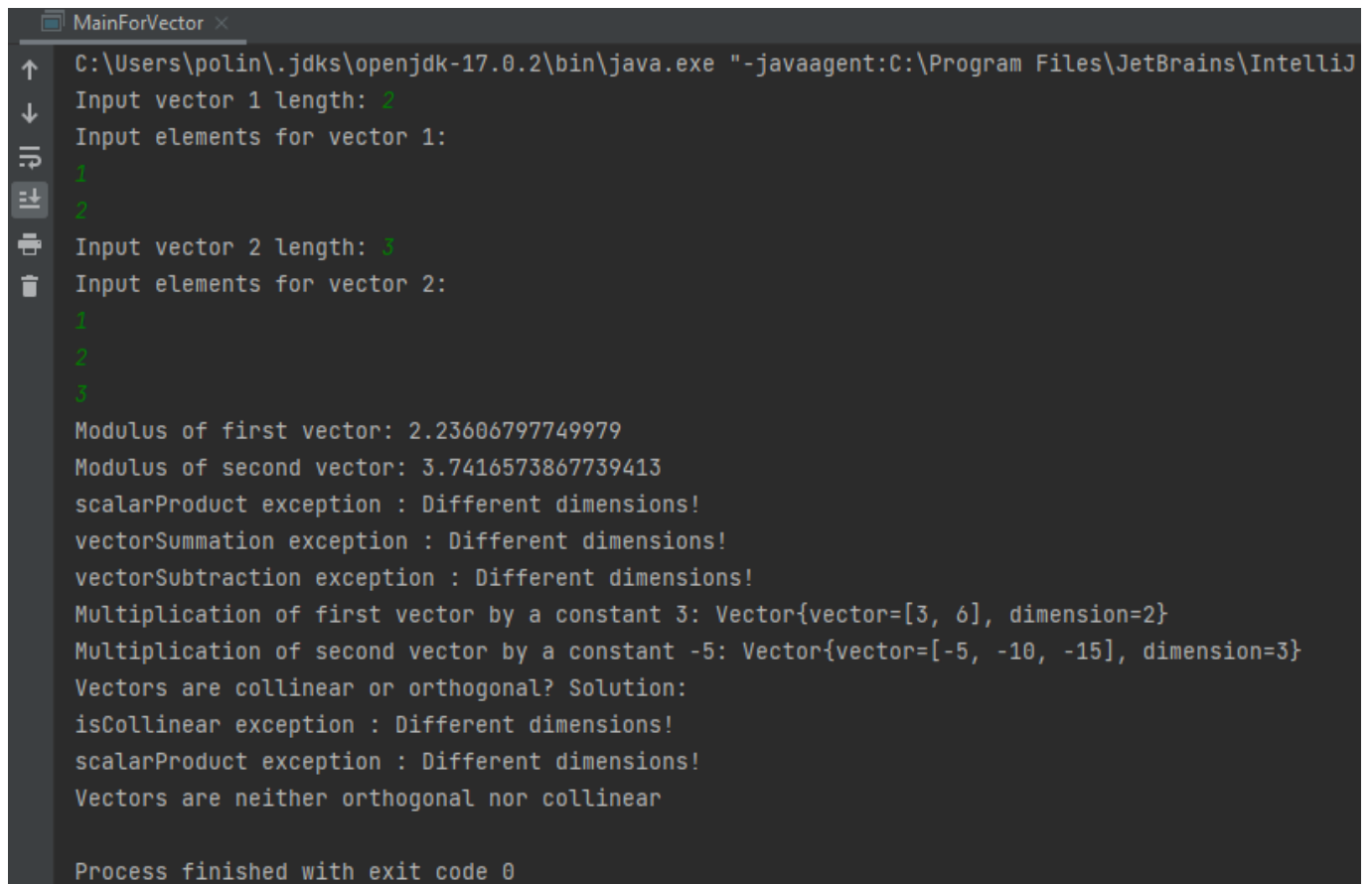
```

        vec1.vectorSummation(vec2);
        vec1.vectorSubtraction(vec2);

        System.out.println("Multiplication of first vector by a constant 3: " +
vec1.constantMultiplication(3));
        System.out.println("Multiplication of second vector by a constant -5: " +
vec2.constantMultiplication(-5));

        System.out.println("Vectors are collinear or orthogonal? Solution: ");
        vec1.defineVectorsStatus(vec2);
    }
}

```



```

MainForVector x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ
Input vector 1 length: 2
Input elements for vector 1:
1
2
Input vector 2 length: 3
Input elements for vector 2:
1
2
3
Modulus of first vector: 2.23606797749979
Modulus of second vector: 3.7416573867739413
scalarProduct exception : Different dimensions!
vectorSummation exception : Different dimensions!
vectorSubtraction exception : Different dimensions!
Multiplication of first vector by a constant 3: Vector{vector=[3, 6], dimension=2}
Multiplication of second vector by a constant -5: Vector{vector=[-5, -10, -15], dimension=3}
Vectors are collinear or orthogonal? Solution:
isCollinear exception : Different dimensions!
scalarProduct exception : Different dimensions!
Vectors are neither orthogonal nor collinear

Process finished with exit code 0

```

Рисунки 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл VectorDim3.java)

```

package task1;

public class VectorDim3 {

    private float x_start;
    private float x_end;
    private float y_start;
    private float y_end;
    private float z_start;
    private float z_end;

    public VectorDim3() {
    }

    public VectorDim3(float x_start, float x_end, float y_start, float y_end, float z_start,
float z_end) {
        this.x_start = x_start;
        this.x_end = x_end;
        this.y_start = y_start;
        this.y_end = y_end;
        this.z_start = z_start;
    }
}

```

```

        this.z_end = z_end;
    }

    public float scalarProduct(VectorDim3 other) {
        try {
            return (this.x_end - this.x_start) * (other.x_end - other.x_start) + (this.y_end -
this.y_start) * (other.y_end - other.y_start) + (this.z_end - this.z_start) * (other.z_end -
other.z_start);
        } catch (Exception e) {
            System.out.println("scalarProduct exception : calculation error!");
            return -1;
        }
    }

    public boolean isOrthogonal(VectorDim3 other){
        try {
            if (scalarProduct(other) == 0.) {
                System.out.println("Vectors are orthogonal");
                return true;
            } else {
                System.out.println("Vectors are not orthogonal");
                return false;
            }
        } catch (Exception e) {
            System.out.println("isOrthogonal exception : calculation error!");
            return false;
        }
    }

    public void isCrossed(VectorDim3 other, float eps){
        /*
        x = x_start*t + x_end*(1 - t)
        y = y_start*t + y_end*(1 - t)
        z = z_start*t + z_end*(1 - t)

        this.x_start*t + this.x_end*(1 - t) = other.x_start*s + other.x_end*(1 - s)
        this.y_start*t + this.y_end*(1 - t) = other.y_start*s + other.y_end*(1 - s)
        */

        try {
            double s = ((other.y_end - this.y_end) / (this.y_start - this.y_end) -
(other.x_end - this.x_end) / (this.x_start - this.x_end)) / ((other.x_start - other.x_end) /
(this.x_start - this.x_end) - (other.y_start - other.y_end) / (this.y_start - this.y_end));
            double t = s * ((other.x_start - other.x_end) / (this.x_start - this.x_end)) +
(other.x_end - this.x_end) / (this.x_start - this.x_end);

            double left = this.z_start * t + this.z_end * (1 - t);
            double right = other.z_start * s + other.z_end * (1 - s);

            if ((left - right) <= eps) {
                System.out.println("Vectors are crossed");
            } else {
                System.out.println("Vectors are not crossed");
            }
        } catch (Exception e) {
            System.out.println("isCrossed exception : calculation error!");
        }
    }

    public void compareVectors(VectorDim3 other){
        try {
            double firstModulus = Math.sqrt(Math.pow((this.x_end - this.x_start), 2) +
Math.pow((this.y_end - this.y_start), 2) + Math.pow((this.z_end - this.z_start), 2));
            double secondModulus = Math.sqrt(Math.pow((other.x_end - other.x_start), 2) +
Math.pow((other.y_end - other.y_start), 2) + Math.pow((other.z_end - other.z_start), 2));
            if (firstModulus > secondModulus) {
                System.out.println("First vector is greater");
            } else if (firstModulus < secondModulus) {
                System.out.println("Second vector is greater");
            } else {
                System.out.println("Vector have same length");
            }
        } catch (Exception e) {

```

```

        System.out.println("compareVectors exception : calculation error!");
    }
}

public void areComplanar(VectorDim3 other, VectorDim3 another) {
    try {
        float this_x = this.x_end - this.x_start;
        float this_y = this.y_end - this.y_start;
        float this_z = this.z_end - this.z_start;

        float other_x = other.x_end - other.x_start;
        float other_y = other.y_end - other.y_start;
        float other_z = other.z_end - other.z_start;

        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_end - another.x_start;
        float another_y = another.y_end - another.y_start;
        float another_z = another.z_end - another.z_start;

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

        if (sumOfMuls == 0.) {
            System.out.println("Vectors are complanar");
        } else {
            System.out.println("Vectors are not complanar");
        }
    } catch (Exception e) {
        System.out.println("areComplanar exception : calculation error!");
    }
}

@Override
public String toString() {
    return "VectorDim3{" +
        "x_start=" + x_start +
        ", x_end=" + x_end +
        ", y_start=" + y_start +
        ", y_end=" + y_end +
        ", z_start=" + z_start +
        ", z_end=" + z_end +
        '}';
}
}

```

Листинг выполнения подзадачи 2 (файл MainForVectorDim3.java)

```

package task1;

import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;

public class MainForVectorDim3 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        VectorDim3 vec1;
        VectorDim3 vec2;
        VectorDim3 vec3;

        try {
            ArrayList<Float> v1 = new ArrayList<>();
            System.out.println("Input parameters for vector 1:");
            for (int i = 0; i < 6; i++) {
                float elem = in.nextFloat();
                v1.add(elem);
            }
            vec1 = new VectorDim3(v1.get(0), v1.get(1), v1.get(2), v1.get(3), v1.get(4),
v1.get(5));
        } catch (Exception e) {
            in.nextLine();
        }
    }
}

```



```

        System.out.println("Wrong input for vector 1!");
        vec1 = new VectorDim3();
    }

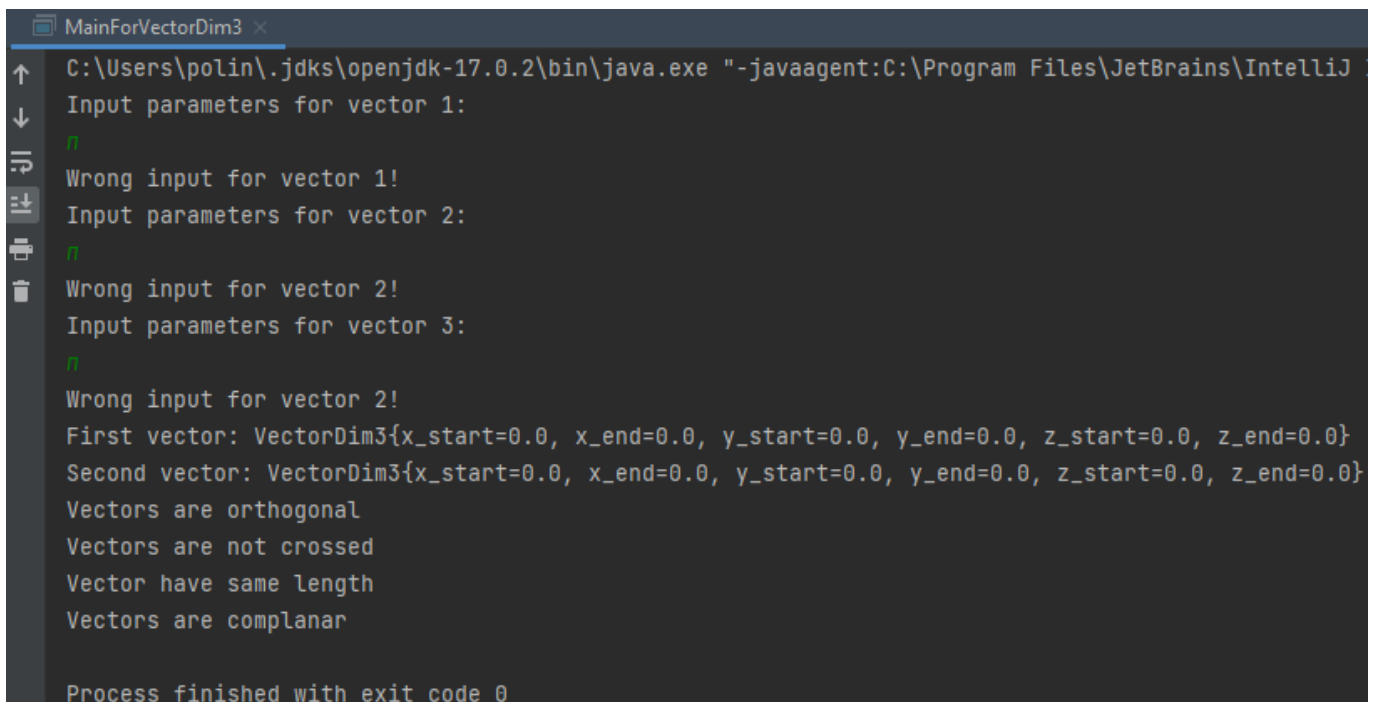
    try {
        ArrayList<Float> v2 = new ArrayList<>();
        System.out.println("Input parameters for vector 2:");
        for (int i = 0; i < 6; i++) {
            float elem = in.nextFloat();
            v2.add(elem);
        }
        vec2 = new VectorDim3(v2.get(0), v2.get(1), v2.get(2), v2.get(3), v2.get(4),
v2.get(5));
    } catch (Exception e) {
        in.nextLine();
        System.out.println("Wrong input for vector 2!");
        vec2 = new VectorDim3();
    }

    try {
        ArrayList<Float> v3 = new ArrayList<>();
        System.out.println("Input parameters for vector 3:");
        for (int i = 0; i < 6; i++) {
            float elem = in.nextFloat();
            v3.add(elem);
        }
        vec3 = new VectorDim3(v3.get(0), v3.get(1), v3.get(2), v3.get(3), v3.get(4),
v3.get(5));
    } catch (Exception e) {
        in.nextLine();
        System.out.println("Wrong input for vector 2!");
        vec3 = new VectorDim3();
    }

    System.out.println("First vector: "+vec1);
    System.out.println("Second vector: "+vec2);

    vec1.isOrthogonal(vec2);
    vec1.isCrossed(vec2, 0.5f);
    vec1.compareVectors(vec2);
    vec1.areComplanar(vec2, vec3);
}
}

```



```

MainForVectorDim3 x
C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ
Input parameters for vector 1:
n
Wrong input for vector 1!
Input parameters for vector 2:
n
Wrong input for vector 2!
Input parameters for vector 3:
n
Wrong input for vector 2!
First vector: VectorDim3{x_start=0.0, x_end=0.0, y_start=0.0, y_end=0.0, z_start=0.0, z_end=0.0}
Second vector: VectorDim3{x_start=0.0, x_end=0.0, y_start=0.0, y_end=0.0, z_start=0.0, z_end=0.0}
Vectors are orthogonal
Vectors are not crossed
Vector have same length
Vectors are complanar

Process finished with exit code 0

```

Рисунок 2 - Результат выполнения кода решения подзадачи 2

Задание 2:

Выполнить задания из варианта 2 лабораторной работы 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

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; в) выбрать заданное число n абитуриентов, имеющих самый высокий средний балл (вывести также полный список абитуриентов, имеющих полупроходной балл).

Листинг выполнения подзадачи 1 (файл Patient.java)

```
package task2;

import java.util.regex.Pattern;

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) throws Exception {
        if (id < 0) {
            throw new Exception("Patient exception : wrong id!");
        }
        if ((cardNumber < 100) || (cardNumber > 999)) {
            throw new Exception("Patient exception : wrong card number!");
        }
        if ((name.equals("")) || (surname.equals("")) || (lastname.equals(""))) {
            throw new Exception("Patient exception : empty name/surname/lastname!");
        }
        if (!Pattern.matches("^8-9\\d{2}-\\d{3}-\\d{2}-\\d{2}", phone)) {
            throw new Exception("Patient exception : wrong phone number!");
        }
        if (diagnosis.equals("")) {
            throw new Exception("Patient exception : no 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 + '\'' +
            '}';
    }
}

```

Листинг выполнения подзадачи 1 (файл MainForPatient.java)

```

package task2;

import java.util.ArrayList;

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

        Patient[] patientsArray = new Patient[0];
        try {
            patientsArray = createPatientsArray();
        } catch (Exception e) {
            e.printStackTrace();
        }
        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() throws Exception {
        Patient p1 = new Patient(1, "Ivan", "Ivanov", "Ivanovich", "House 5", "89683742647",
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()]);
}
}

```

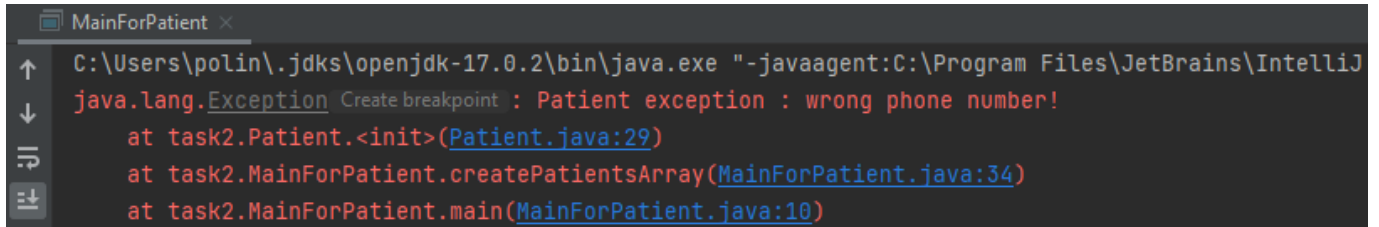


Рисунок 3 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл Abiturient.java)

```

package task2;

import java.util.ArrayList;
import java.util.regex.*;

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) throws Exception {
        if (id < 0) {
            throw new Exception("Abiturient exception : wrong id!");
        }
        if ((name.equals("")) || (surname.equals("")) || (lastname.equals(""))) {
            throw new Exception("Abiturient exception : empty name/surname/lastname!");
        }
        if (!Pattern.matches("^8-9\\d{2}-\\d{3}-\\d{2}-\\d{2}", phone)) {
            throw new Exception("Abiturient exception : wrong phone number!");
        }
        for(int x : marks){
            if((x < 2) || (x > 5)){
                throw new Exception("Abiturient exception : wrong mark number!");
            }
        }
        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 +
            '}';
    }
}

```

Листинг выполнения подзадачи 2 (файл MainForAbiturient.java)

```

package task2;

import java.util.*;

public class MainForAbiturient {

```

```

public static void main(String[] args) {

    Abiturient a1 = null;
    try {
        a1 = new Abiturient(1, "Ivan", "Ivanov", "Ivanovich", "House 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<>();

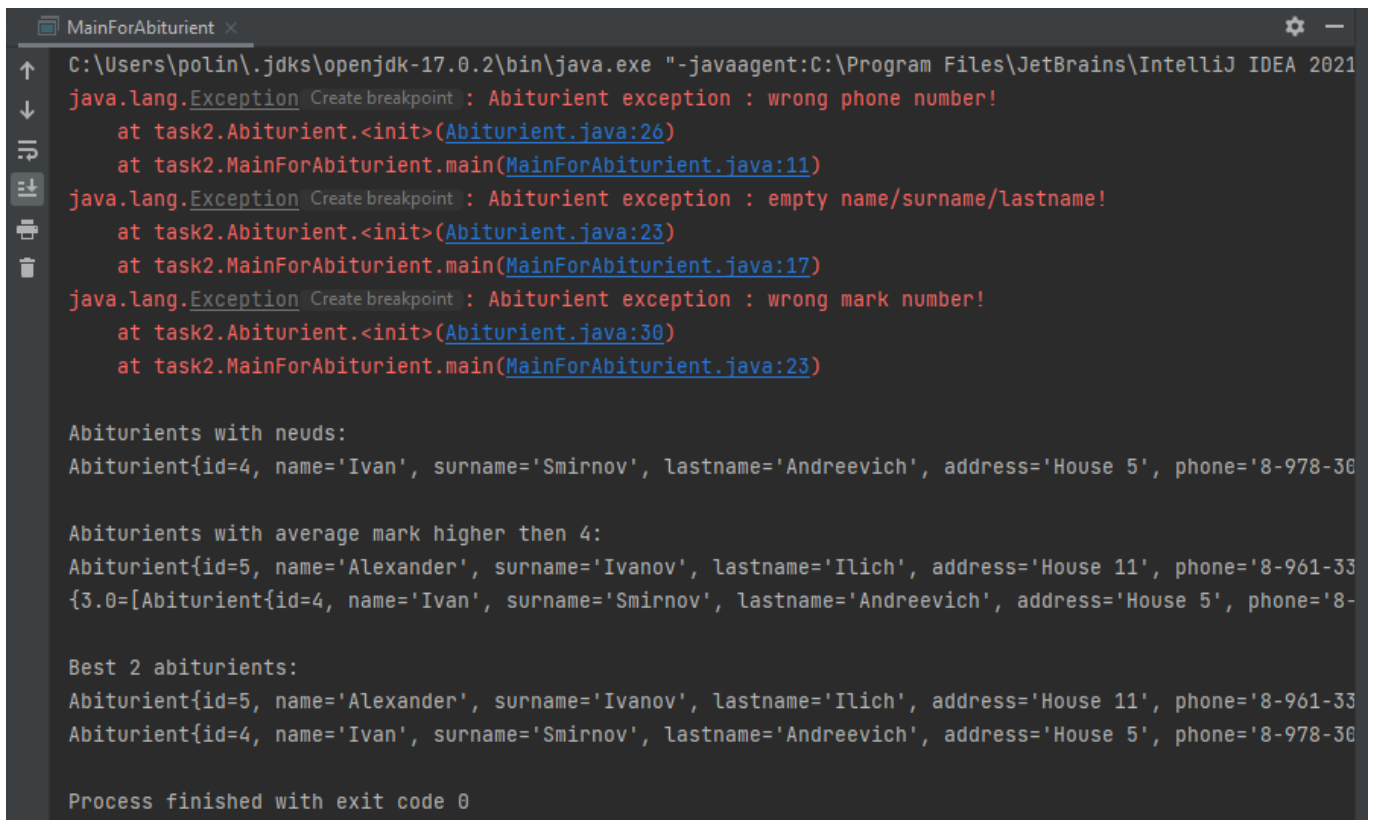
        SortedMap<Float, ArrayList<Abiturient>> map = new TreeMap<>();
        for (int i = 0; i < abiturientsArray.length; i++){
            try {
                float avg = 0;
                for (Integer m : abiturientsArray[i].getMarks()) {
                    avg += m;
                }
              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

```

```

C:\Users\polin\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2021
java.lang.Exception Create breakpoint : Abiturient exception : wrong phone number!
    at task2.Abiturient.<init>(Abiturient.java:26)
    at task2.MainForAbiturient.main(MainForAbiturient.java:11)
java.lang.Exception Create breakpoint : Abiturient exception : empty name/surname/lastname!
    at task2.Abiturient.<init>(Abiturient.java:23)
    at task2.MainForAbiturient.main(MainForAbiturient.java:17)
java.lang.Exception Create breakpoint : Abiturient exception : wrong mark number!
    at task2.Abiturient.<init>(Abiturient.java:30)
    at task2.MainForAbiturient.main(MainForAbiturient.java:23)

Abiturients with neuds:
Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5', phone='8-978-36

Abiturients with average mark higher then 4:
Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11', phone='8-961-33
{3.0=[Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5', phone='8-

Best 2 abiturients:
Abiturient{id=5, name='Alexander', surname='Ivanov', lastname='Ilich', address='House 11', phone='8-961-33
Abiturient{id=4, name='Ivan', surname='Smirnov', lastname='Andreevich', address='House 5', phone='8-978-36

Process finished with exit code 0

```

Рисунок 4 - Результат выполнения кода решения подзадачи 2

Задание 3:

В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. При этом могут рассматриваться два варианта:

- каждая строка состоит из одного слова;
- каждая строка состоит из нескольких слов.

Имена входного и выходного файлов, а также абсолютный путь к ним могут быть введены как параметры командной строки или храниться в файле.

1. В каждой строке стихотворения Александра Блока найти и заменить заданную подстроку на подстроку иной длины.
2. В каждой строке найти слова, начинающиеся с гласной буквы.

Текстовый файл для выполнения подзадач 1 и 2 (файл OriginalText_ABlock.txt)

```

Ночь, улица, фонарь, аптека,
Бессмысленный и тусклый свет.
Живи еще хоть четверть века -
Все будет так. Исхода нет.
Умрешь - начнешь опять сначала
И повторится все, как встарь:
Ночь, ледяная рябь канала,
Аптека, улица, фонарь.
```

Листинг выполнения подзадачи 1 (файл Main1.java)

```
package task3;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import java.util.Collections;
import java.util.Scanner;

public class Main1 {
    public static void main(String[] args) {

        File inp_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\OriginalText_ABloc
k.txt");
        Path out_file_path =
Paths.get("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\ModifiedText_
ABlock.txt");
        File out_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\ModifiedText_ABloc
k.txt");

        String substring = " улица, ";
        String replacement = " тротуар, ";

        if(out_file.delete()) {
            try {
                out_file.createNewFile();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        Scanner scanner = null;
        try {
            scanner = new Scanner(inp_file);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        while(scanner.hasNextLine()) {
            String line = scanner.nextLine();
            int index = 0;
            while (index != -1) {
                index = line.indexOf(substring);
                if (index != -1){
                    line = line.replace(substring, replacement);
                }
            }
            try {
                Files.write(out_file_path, Collections.singleton(line),
StandardCharsets.UTF_8, StandardOpenOption.APPEND);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Результирующий текстовый файл после выполнения подзадачи 1 (файл ModifiedText_ABlock.txt)

```
Ночь, тротуар, фонарь, аптека,  
Бессмысленный и тусклый свет.  
Живи еще хоть четверть века -  
Все будет так. Исхода нет.  
Умрешь - начнешь опять сначала  
И повторится все, как встарь:  
Ночь, ледяная рябь канала,  
Аптека, тротуар, фонарь.
```

Листинг выполнения подзадачи 2 (файл Main2.java)

```
package task3;  
  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.nio.charset.StandardCharsets;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.nio.file.StandardOpenOption;  
import java.util.Collections;  
import java.util.List;  
import java.util.Locale;  
import java.util.Scanner;  
  
public class Main2 {  
    public static void main(String[] args) {  
  
        File inp_file = new  
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\OriginalText_ABloc  
k.txt");  
        Path out_file_path =  
Paths.get("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\ModifiedText_  
ABlock.txt");  
        File out_file = new  
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task3\\ModifiedText_ABloc  
k.txt");  
  
        String vowels = "аоэеиуыёя";  
  
        if(out_file.delete()) {  
            try {  
                out_file.createNewFile();  
            } catch (IOException e) {  
                e.printStackTrace();  
            }  
        }  
  
        Scanner scanner = null;  
        try {  
            scanner = new Scanner(inp_file);  
        } catch (FileNotFoundException e) {  
            e.printStackTrace();  
        }  
        while(scanner.hasNextLine()) {  
            String line = scanner.nextLine();  
  
            line = line.replaceAll("\\pP", "");  
            String[] words = line.split(" ");  
            String out = "";  
            for(String word : words){  
                String first_letter = word.length() > 1 ? word.substring(0, 1) : word;  
                first_letter = first_letter.toLowerCase(Locale.ROOT);  
                if (vowels.contains(first_letter)) {  
                    out = out.concat(word).concat(" ");  
                }  
            }  
  
            try {
```

```

        Files.write(out_file_path, Collections.singleton(out), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}
}

```

Результирующий текстовый файл после выполнения подзадачи 2 (файл ModifiedText_ABlock.txt)

```

улица аптека
и
еще
Исхода
Умрешь  опять
И
Аптека улица

```

Задание 4:

При выполнении следующих заданий для вывода результатов создавать новую директорию и файл средствами класса File.

1. Прочитать текст Java-программы и записать в другой файл в обратном порядке символы каждой строки.
2. Прочитать текст Java-программы и в каждом слове длиннее двух символов все строчные символы заменить прописными.

Листинг исходной Java-программы для выполнения подзадач 1 и 2 (файл Java_program.java)

```

package task4;

import java.util.Scanner;

public class Java_program {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("What is your name? ");
        String name = in.nextLine();

        System.out.printf("Hello, %s! \n", name);
        in.close();
    }
}

```

Листинг выполнения подзадачи 1 (файл Main1.java)

```
package task4;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import java.util.Collections;
import java.util.Locale;
import java.util.Scanner;

public class Main1 {
    public static void main(String[] args) {

        File inp_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task4\\Java_program.java"
);

        File out_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task4\\Java_program_resul
t.java");

        out_file.delete();

        try {
            out_file.createNewFile();
        } catch (IOException e) {
            e.printStackTrace();
        }

        Path out_file_path = Path.of(out_file.getPath());

        Scanner scanner = null;
        try {
            scanner = new Scanner(inp_file);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        while(scanner.hasNextLine()) {
            String line = scanner.nextLine();

            char[] c = line.toCharArray();
            int len = c.length/2;
            for (int i = 0; i < len; i++) {
                char temp = c[i];
                c[i] = c[c.length - 1 - i];
                c[c.length - 1 - i] = temp;
            }
            String out = new String(c);

            try {
                Files.write(out_file_path, Collections.singleton(out), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
            } catch (IOException e) {
                e.printStackTrace();
            }

        }
    }
}
```

Листинг модифицированной Java-программы после выполнения подзадачи 1
(файл Java_program_result.java)

```
;4ksat egakcap

;rennacS.litu.avaj tropmi

{ margorp_avaJ ssalc cilbup

{ )sgra ][gnirtS(niam dioV citats cilbup
;)ni.metsyS(rennacS wen = ni rennacS
;) " ?eman ruoy si tahW"(tnirp.tuo.metsyS
;) (eniLtxen.ni = eman gnirtS

;)eman , "n\ !s% ,olleH"(ftnirp.tuo.metsyS
;) (esolc.ni
}
}
```

Листинг выполнения подзадачи 2 (файл Main2.java)

```
package task4;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import java.util.Collections;
import java.util.Locale;
import java.util.Scanner;

public class Main2 {
    public static void main(String[] args) {

        File inp_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task4\\Java_program.java"
);

        File out_file = new
File("C:\\Users\\polin\\IdeaProjects\\JavaMagisterCourse\\Lab5\\src\\task4\\Java_program_resul
t.java");

        out_file.delete();

        try {
            out_file.createNewFile();
        } catch (IOException e) {
            e.printStackTrace();
        }

        Path out_file_path = Path.of(out_file.getPath());

        Scanner scanner = null;
        try {
            scanner = new Scanner(inp_file);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        while(scanner.hasNextLine()) {
            String line = scanner.nextLine();

            //line = line.replaceAll("\\pP", " ");
            String[] words = line.split("[!(<\\ \\#\\.\\:\\\\"]");

            for(String word : words){
                if (word.length() > 2) {
```

```

        String word_upper = word.toUpperCase(Locale.ROOT);
        line = line.replace(word, word_upper);
    };
}

    try {
        Files.write(out_file_path, Collections.singleton(line),
StandardCharsets.UTF_8, StandardOpenOption.APPEND);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
}
}
}
}

```

Листинг модифицированной Java-программы после выполнения подзадачи 2
(файл Java_program_result.java)

```

PACKAGE TASK4;

IMPORT JAVA.UTIL.SCANNER;

PUBLIC CLASS JAVA_PROGRAM {

    PUBLIC STATIC VOID MAIN(STRING[] ARGS) {
        SCANNER in = NEW SCANNER(SYSTEM.in);
        SYSTEM.OUT.PRINT("WHAT is YOUR NAME? ");
        STRING NAME = in.NEXTLINE();

        SYSTEM.OUT.PRINTF("HELLO, %s! \n", NAME);
        in.CLOSE();
    }
}
}

```

Ссылка на программное решение:

Программное решение представлено в репозитории распределённой системы управления версиями Git:

<https://github.com/Owlfeather/JavaMagisterCourse/tree/main/Lab5/src>

Вывод:

При выполнении лабораторной работы были получены навыки обработки исключений в Java и навыки работы с файлами в Java.