

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

Вариант 13

Дисциплина: Языки программирования для работы с большими данными

(Подпись, дата)

П.В. Степанов
 (И.О. Фамилия)

Москва, 2022

Цель лабораторной работы: получение первичных навыков работы с файлами и обработки исключений в Java.

Ход работы:

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.

Листинг выполнения подзадачи 1:

Класс MyVectorClass:

```
package laba_5.task_1;

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

    public MyVectorClass() {
    }

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

    public float scalarnost(MyVectorClass other){
        try {
            return (this.x_2 - this.x_1) * (other.x_2 - other.x_1) + (this.y_2 -
this.y_1) * (other.y_2 - other.y_1) + (this.z_2 - this.z_1) * (other.z_2 -
other.z_1);
        } catch (Exception error) {
            System.out.println("Error in Scalarny method");
            return 0;
        }
    }

    public boolean ortogonalnost(MyVectorClass other){
        try {
```

```

        //два вектора являются ортогональными (перпендикулярными), если их
        скалярное
        // произведение равно нулю.
        if(scalarnost(other) == 0.){
            return true;
        } else {
            return false;
        }
    } catch (Exception error){
        System.out.println("Error in Orthogonal method");
        return false;
    }
}

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

        if((left - right) <= eps){
            return true;
        } else {
            return false;
        }
    } catch (Exception error){
        System.out.println("Error in Peresechenie method");
        return false;
    }
}

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

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

```

```

        float this_z = this.z_2 - this.z_1;

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

        float tmpX = this.y * other_z - this.z * other.y;
        float tmpY = -(this.x * other_z - this.z * other.x);
        float tmpZ = this.z * other.y - this.y * other.x;

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

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

        if(sumOfMuls == 0.){
            return true;
        } else {
            return false;
        }
    } catch (Exception error){
        System.out.println("Error in complenarnost method");
        return false;
    }
}

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

```

Листинг выполнения подзадачи 2:

Класс SquareMatrix:

```

package laba_5.task_1;

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

```

```

public SquareMatrix(int[][] mtx) {
    set(mtx);
}

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

public int[][] get() {
    return matrix;
}

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

        for (int i = 0; i < nDim; i++) {
            for (int j = 0; j < nDim; j++) {
                resMtx[i][j] = matrix[i][j] + rightMtx[i][j];
            }
        }
        return new SquareMatrix(resMtx);
    } catch (Exception error) {
        System.out.println("Error in add method");
        return rMtx;
    }
}

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

        for (int i = 0; i < nDim; i++) {
            for (int j = 0; j < nDim; j++) {
                resMtx[i][j] = matrix[i][j] - rightMtx[i][j];
            }
        }
        return new SquareMatrix(resMtx);
    } catch (Exception error) {
        System.out.println("Error in sub method");
        return rMtx;
    }
}

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

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

```

```

        return new SquareMatrix(resMtx);
    } catch (Exception error) {
        System.out.println("Error in mul method");
        return rMtx;
    }
}

public int firstNorma() {
    try {
        int max = 0;
        for (int i = 0; i < nDim; i++) {
            max += matrix[0][i];
        }
        for (int i = 1; i < nDim; i++) {
            int sum = 0;
            for (int j = 0; j < nDim; j++) {
                sum += matrix[i][j];
            }
            max = Math.max(max, sum);
        }
        return max;
    } catch (Exception error) {
        System.out.println("Error in firstNorma method");
        return 0;
    }
}

public int secondNorma() {
    try {
        int max = 0;
        for (int i = 0; i < nDim; i++) {
            max += matrix[i][0];
        }
        for (int i = 0; i < nDim; i++) {
            int sum = 0;
            for (int j = 1; j < nDim; j++) {
                sum += matrix[j][i];
            }
            max = Math.max(max, sum);
        }
        return max;
    } catch (Exception error) {
        System.out.println("Error in secondNorma method");
        return 0;
    }
}

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

```

Задание №2:

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

Листинг выполнения подзадачи 1:

Класс Patient:

```
package laba_5.task_2.patient;
import java.util.regex.Pattern;

public class Patient {
    private String name;
    private String surname;
    private String lastname;
    private String address;
    private String phone;
    private String diagnos;

    public Patient( String name, String surname, String lastname, String address,
String phone, String diagnos)throws Exception {

        if ((name.equals("")) || (surname.equals("")) || (lastname.equals(""))) {
            throw new Exception("NOT FULL NAME");
        }
        if (!Pattern.matches("^8-9\\d{9}", phone)) {
            throw new Exception("Phone is NOT correct");
        }
        if (diagnos.equals("")) {
            throw new Exception("Diagnos is not set");
        }
        this.name = name;
        this.surname = surname;
        this.lastname = lastname;
        this.address = address;
        this.phone = phone;
        this.diagnos = diagnos;
    }

    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 String getDiagnosis() {
        return diagnos;
    }

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

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

```

Листинг выполнения подзадачи 2:

Класс Abiturient:

```

package laba_5.task_2.abitur;
import java.util.ArrayList;
import java.util.regex.*;

public class Abiturient {
    private String name;
    private String surname;
    private String lastname;
    private String ab_address;
    private String ab_phone;
    private ArrayList<Integer> marks;

    public Abiturient(int id, String name, String surname, String lastname, String
address, String phone, ArrayList<Integer> marks) throws Exception {

        if ((name.equals("")) || (surname.equals("")) || (lastname.equals("")))) {
            throw new Exception("NOT FULL NAME");
        }
    }
}

```



```

    }
    if (!Pattern.matches("^8-9\\d{9}", ab_phone)) {
        throw new Exception("Phone is NOT correct");
    }
    for(int x : marks){
        if((x < 1) || (x > 5)){
            throw new Exception("Mark is NOT correct!");
        }
    }
    this.name = name;
    this.surname = surname;
    this.lastname = lastname;
    this.ab_address = address;
    this.ab_phone = phone;
    this.marks = marks;
}

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 ab_address;
}

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

public String getPhone() {
    return ab_phone;
}

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

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

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

```

```

    }

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

```

Задание №3:

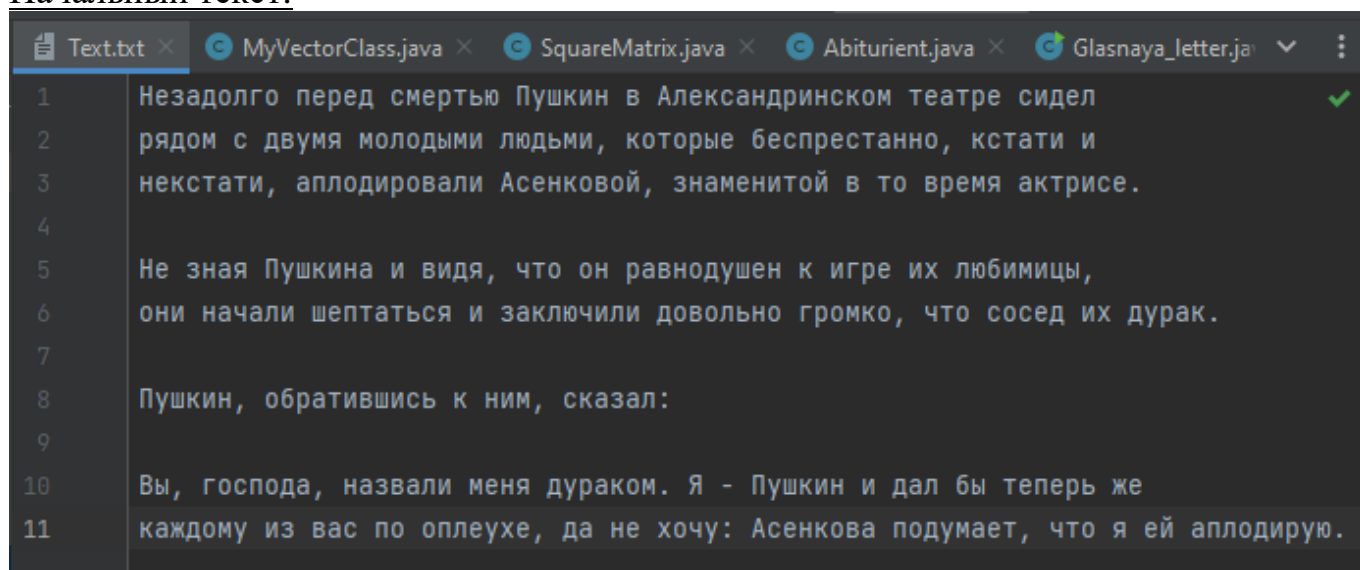
В следующих заданиях требуется ввести последовательность строк из текстового потока и выполнить указанные действия. При этом могут рассматриваться два варианта:

- 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 найти слова, начинающиеся с гласной буквы.
2. Найти и вывести слова текста, для которых последняя буква одного слова совпадает с первой буквой следующего слова.

Начальный текст:



```

1 Незадолго перед смертью Пушкин в Александринском театре сидел
2 рядом с двумя молодыми людьми, которые беспрестанно, кстати и
3 некстати, аплодировали Асенковой, знаменитой в то время актрисе.
4
5 Не зная Пушкина и видя, что он равнодушен к игре их любимицы,
6 они начали шептаться и заключили довольно громко, что сосед их дурак.
7
8 Пушкин, обратившись к ним, сказал:
9
10 Вы, господа, называли меня дураком. Я - Пушкин и дал бы теперь же
11 каждому из вас по оплеухе, да не хочу: Асенкова подумает, что я ей аплодирую.

```

Листинг выполнения подзадачи 1:

```
package laba_5.task_3;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.util.Scanner;
import java.util.Collections;
import java.util.Locale;

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

        File file_1 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Text.txt");
        Path file_2_path =
Paths.get("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Finish_text
_1.txt");
        File file_2 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Finish_text_1.tx
t");

        if(file_2.delete()) {
            try {
                file_2.createNewFile();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        Scanner scanner = null;
        try {
            scanner = new Scanner(file_1);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

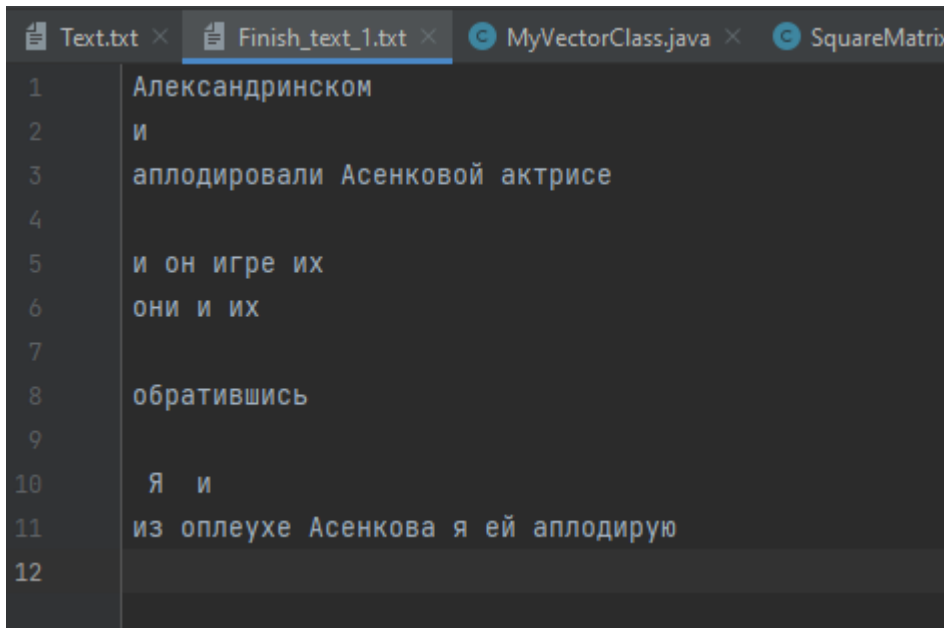
        String start_letters = "аоёеиыуёя";

        while(scanner.hasNextLine()) {
            String file_line = scanner.nextLine();
            file_line = file_line.replaceAll("\\pP", "");
            String[] words = file_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 (start_letters.contains(first_letter)) {
                    out = out.concat(word).concat(" ");
                }
            }

            try {
                Files.write(file_2_path, Collections.singleton(out),
StandardCharsets.UTF_8, StandardOpenOption.APPEND);
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
}  
}  
}
```

Результат выполнения подзадачи 1:



The screenshot shows an IDE with four tabs: 'Text.txt', 'Finish_text_1.txt', 'MyVectorClass.java', and 'SquareMatrix'. The 'Finish_text_1.txt' tab is active, displaying the following text:

```
1    Александринском  
2    и  
3    аплодировали Асенковой актрисе  
4  
5    и он игре их  
6    они и их  
7  
8    обратившись  
9  
10   Я и  
11   из оплеухе Асенкова я ей аплодирую  
12
```

Листинг выполнения подзадачи 2:

```
package laba_5.task_3;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.nio.file.StandardOpenOption;  
import java.io.File;  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.nio.charset.StandardCharsets;  
import java.nio.file.Files;  
import java.util.Scanner;  
import java.util.Collections;  
import java.util.Locale;  
  
public class First_last_word_letter {  
    public static void main(String[] args) {  
  
        File file_1 = new  
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Text.txt");  
        Path file_2_path =  
Paths.get("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Finish_text  
_2.txt");  
        File file_2 = new  
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\Finish_text_2.tx  
t");  
  
        if(file_2.delete()) {  
            try {  
                file_2.createNewFile();  
            } catch (IOException e) {  
                e.printStackTrace();  
            }  
        }  
    }  
}
```

```

    }

    Scanner scanner = null;
    try {
        scanner = new Scanner(file_1);
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }

    while(scanner.hasNextLine()) {
        String file_line = scanner.nextLine();
        file_line = file_line.replaceAll("\\pP", "");
        String[] words = file_line.split(" ");
        String out = "";
        for (int i=0; i < words.length -1 ; ++i){
            String last_letter = words[i].length() > 1 ?
words[i].substring(words[i].length()-1, words[i].length()-0) : words[i];
            String first_letter = (words[i + 1].length() > 1) ? words[i +
1].substring(0, 1) : words[i + 1];
            if (last_letter.equals(first_letter)){
                out = words[i] + " " + words[i+1];
            }
        }

        try {
            Files.write(file_2_path, Collections.singleton(out),
StandardCharsets.UTF_8, StandardOpenOption.APPEND);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

Результат выполнения подзадачи 2:

1	
2	кстати и
3	
4	
5	что он
6	
7	
8	
9	
10	
11	по оплеухе
12	

Задание №4:

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

1. Прочитать текст Java-программы и в каждом слове длиннее двух символов все строчные символы заменить прописными.
2. В файле, содержащем фамилии студентов и их оценки, записать прописными буквами фамилии тех студентов, которые имеют средний балл более “7”.

Листинг выполнения подзадачи 1:

```
package laba_5.task_4;
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.StandardOpenOption;
import java.util.Collections;
import java.util.Locale;
import java.util.Scanner;

public class Java_text {

    public static void main(String[] args) {

        File file_1 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_3\\First_last_word_
letter.java");

        File file_2 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_4\\Java_program_cha
nged.java");

        file_2.delete();

        try {
            file_2.createNewFile();
        } catch (IOException e) {
            e.printStackTrace();
        }

        Path out_file_path = Path.of(file_2.getPath());

        Scanner scanner = null;
        try {
            scanner = new Scanner(file_1);
        } 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();
        }
    }
}
```

Результат выполнения подзадачи 1:

```

h_text_1.txt × First_last_word_letter.java × Java_program_changed.java × Java_text.java ×
PACKAGE LABA_5.TASK_3;
IMPORT JAVA.NIO.FILE.PATH;
IMPORT JAVA.NIO.FILE.PATHS;
IMPORT JAVA.NIO.FILE.STANDARDOPENOPTION;
IMPORT JAVA.io.FILE;
IMPORT JAVA.io.FILENOTFOUNDEXCEPTION;
IMPORT JAVA.io.IOEXCEPTION;
IMPORT JAVA.NIO.CHARSET.STANDARDCHARSETS;
IMPORT JAVA.NIO.FILE.FILES;
IMPORT JAVA.UTIL.SCANNER;
IMPORT JAVA.UTIL.COLLECTIONS;
IMPORT JAVA.UTIL.LOCALE;

PUBLIC CLASS FIRST_LAST_WORD_LETTER {
    PUBLIC STATIC VOID MAIN(STRING[] ARGS) {

        FILE FILE_1 = NEW FILE("C:\\USERS\\METEL\\IDEAPROJECTS\\MY_LAB_5\\
        PATH FILE_2_PATH = PATHS.GET("C:\\USERS\\METEL\\IDEAPROJECTS\\MY_L
        FILE FILE_2 = NEW FILE("C:\\USERS\\METEL\\IDEAPROJECTS\\MY_LAB_5\\

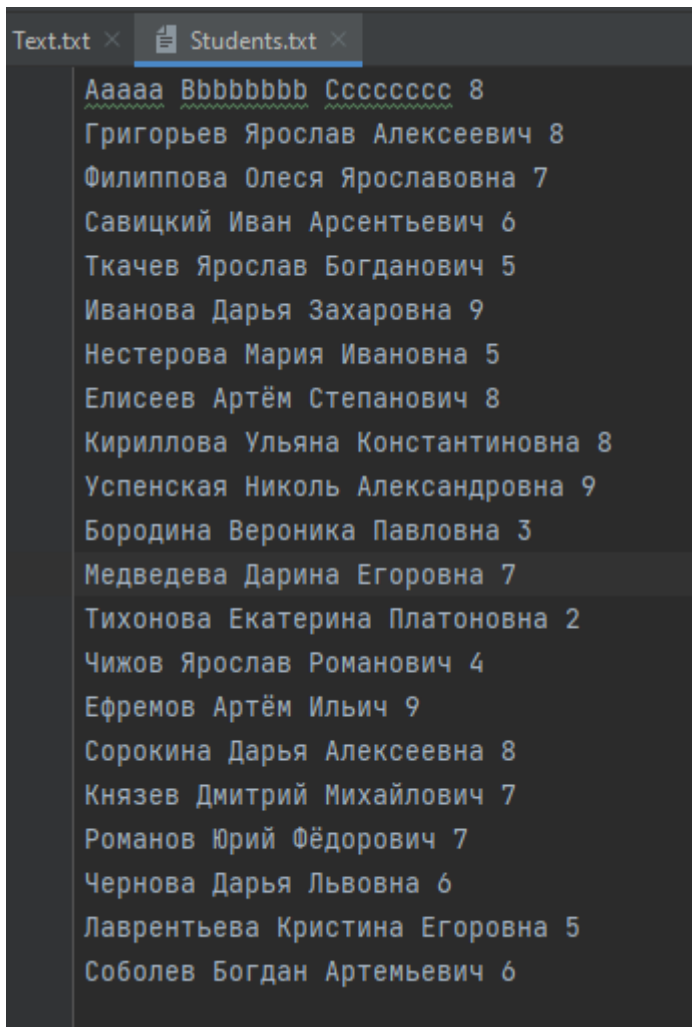
        if(FILE_2.DELETE()) {
            TRY {
                FILE_2.CREATENEWFILE();
            } CATCH (IOEXCEPTION e) {
                e.PRINTSTACKTRACE();
            }
        }

        SCANNER SCANNER = NULL;
        TRY {
            SCANNER = NEW SCANNER(FILE_1);
        } CATCH (FILENOTFOUNDEXCEPTION e) {
            e.PRINTSTACKTRACE();
        }

        WHILE(SCANNER.HASNEXTLINE()) {
            STRING FILE_LINE = SCANNER.NEXTLINE();
            FILE_LINE = FILE_LINE.REPLACEALL("\\PP", "");
            STRING[] WORDS = FILE_LINE.SPLIT(" ");
            STRING OUT = " ":

```


Начальный текст:



The screenshot shows a text editor with two tabs: 'Text.txt' and 'Students.txt'. The 'Students.txt' tab is active and displays a list of 20 students, each followed by a mark. The text is as follows:

Student Name	Mark
Ааааа Bbbbbbbb Cccccccs	8
Григорьев Ярослав Алексеевич	8
Филиппова Олеся Ярославовна	7
Савицкий Иван Арсентьевич	6
Ткачев Ярослав Богданович	5
Иванова Дарья Захаровна	9
Нестерова Мария Ивановна	5
Елисеев Артём Степанович	8
Кириллова Ульяна Константиновна	8
Успенская Николь Александровна	9
Бородина Вероника Павловна	3
Медведева Дарина Егоровна	7
Тихонова Екатерина Платоновна	2
Чижов Ярослав Романович	4
Ефремов Артём Ильич	9
Сорокина Дарья Алексеевна	8
Князев Дмитрий Михайлович	7
Романов Юрий Фёдорович	7
Чернова Дарья Львовна	6
Лаврентьева Кристина Егоровна	5
Соболев Богдан Артемьевич	6

Листинг выполнения подзадачи 2:

```
package laba_5.task_4.students;
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.StandardOpenOption;
import java.util.Collections;
import java.util.Locale;
import java.util.Scanner;

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

        File file_2_1 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_4\\students\\Studen
ts.txt");

        File file_2_2 = new
File("C:\\Users\\metel\\IdeaProjects\\my_lab_5\\src\\laba_5\\task_4\\students\\Studen
ts_changed.txt");
```

```

file_2_2.delete();

try {
    file_2_2.createNewFile();
} catch (IOException e) {
    e.printStackTrace();
}

Path out_file_path = Path.of(file_2_2.getPath());

Scanner scanner = null;
try {
    scanner = new Scanner(file_2_1);
} catch (FileNotFoundException e) {
    e.printStackTrace();
}
while(scanner.hasNextLine()) {
    String line = scanner.nextLine();

    String[] words = line.split(" ");
    System.out.println(words[3]);
    Integer mark = Integer.parseInt(words[3]);
    if (mark > 7){
        String word_upper = words[0].toUpperCase(Locale.ROOT);
        line = line.replace(words[0], word_upper);
    }

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

```

Результат выполнения подзадачи 2:

```
AAAAA Bbbbbbbb Cccccccc 8
ГРИГОРЬЕВ Ярослав Алексеевич 8
Филиппова Олеся Ярославовна 7
Савицкий Иван Арсентьевич 6
Ткачев Ярослав Богданович 5
ИВАНОВА Дарья Захаровна 9
Нестерова Мария Ивановна 5
ЕЛИСЕЕВ Артём Степанович 8
КИРИЛЛОВА Ульяна Константиновна 8
УСПЕНСКАЯ Николь Александровна 9
Бородина Вероника Павловна 3
Медведева Дарина Егоровна 7
Тихонова Екатерина Платоновна 2
Чижов Ярослав Романович 4
ЕФРЕМОВ Артём Ильич 9
СОРОКИНА Дарья Алексеевна 8
Князев Дмитрий Михайлович 7
Романов Юрий Фёдорович 7
Чернова Дарья Львовна 6
Лаврентьева Кристина Егоровна 5
Соболев Богдан Артемьевич 6
|
```

Программное решение представлено в репозитории распределённой системы управления версиями Git:

https://github.com/matvilen/BigDataLanguages/tree/main/my_lab_5/src/laba_5

Вывод: при выполнении лабораторной работы были получены навыки обработки исключений в Java и навыки работы с файлами в Java.