



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

О Т Ч Е Т

по лабораторной работе № 5

Название: Исключения. Файлы.

Дисциплина: Языки программирования для работы с большими
данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

Д.В. Авдонин

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

(И.О. Фамилия)

Москва, 2022

Лабораторная работа № 5

Задание:

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.

Ход работы:

Код программы:

```
public class Vector{
    private int n;
    private ArrayList<Double> data;

    public Vector(){
        this.n = 0;
        data = new ArrayList<>();
    }

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

    public Vector(double... data){
        n = data.length;
        this.data = new ArrayList<>(n);
        for (int i = 0; i < n; i++){
            this.data.add(data[i]);
        }
    }

    public int getN() {
        return n;
    }

    public void setN(int n) {
        this.n = n;
    }

    public ArrayList<Double> getData() {
        return data;
    }

    public void setData(ArrayList<Double> data) {
        this.data = data;
    }
}
```

```

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

public double modulus(){ //модуль
    double mod_sq = 0;
    for (double d : this.data){
        mod_sq += d*d;
    }
    return Math.sqrt(mod_sq);
}

public double scalar(Vector v){ //скалярное произведение
    try {
        double sc = 0;
        for (int i = 0; i < n; i++) {
            sc += this.data.get(i) * v.data.get(i);
        }
        return sc;
    }
    catch (Exception e){
        System.out.println("Ошибочные размеры вектора");
        return -1;
    }
}

public Vector sum(Vector v){ // сумма
    try {
        Vector v_output = new Vector(n);
        for (int i = 0; i < n; i++) {
            v_output.data.add(this.data.get(i) + v.data.get(i));
        }
        return v_output;
    }
    catch (Exception e){
        System.out.println("Ошибочный размер вектора");
        return new Vector();
    }
}

public Vector diff(Vector v){ // разность
    try {
        Vector v_output = new Vector(n);
        for (int i = 0; i < n; i++) {
            v_output.data.add(this.data.get(i) - v.data.get(i));
        }
        return v_output;
    }
    catch (Exception e){
        System.out.println("Ошибочный размер вектора");
        return new Vector();
    }
}

public Vector multi(int c){ // умножение на константу
    Vector v_output = new Vector(this.data.size());
    for (int i = 0; i < n; i++){
        v_output.data.add(this.data.get(i) * c);
    }
}

```

```

        return v_output;
    }
}

public static void isCollinear_Orthogonal(Vector v1, Vector v2){
    try {
        boolean isCollinear = true;
        ArrayList<Double> ratio = new ArrayList<>();
        for (int i = 0; i < v1.getN(); i++) {
            if ((v1.getData().get(i) != 0 && v2.getData().get(i) == 0) ||
                (v2.getData().get(i) == 0 && v1.getData().get(i) != 0)) {
                isCollinear = false;
                break;
            }
            if (v1.getData().get(i) == 0 && v2.getData().get(i) == 0) {
                break;
            }
            ratio.add(v1.getData().get(i) / v2.getData().get(i));
        }
        for (int i = 0; i < v1.getN() - 1; i++) {
            if (!Objects.equals(ratio.get(i), ratio.get(i + 1))) {
                isCollinear = false;
                break;
            }
        }
        if (isCollinear) {
            System.out.println("Вектора коллинеарны");
        } else if (v1.scalar(v2) == 0) {
            System.out.println("Вектора ортогональны");
        } else {
            System.out.println("Вектора не коллинеарны и не ортогональны");
        }
    }
    catch (Exception e){
        System.out.println("Ошибка размерности");
    }
}

public static void main(String[] args){
    Scanner scanner = new Scanner(System.in);
    int number, dimension;
    System.out.print("Введите количество векторов: ");
    try {
        number = scanner.nextInt();
    }
    catch (Exception e){
        System.out.println("error");
        number = -1;
    }

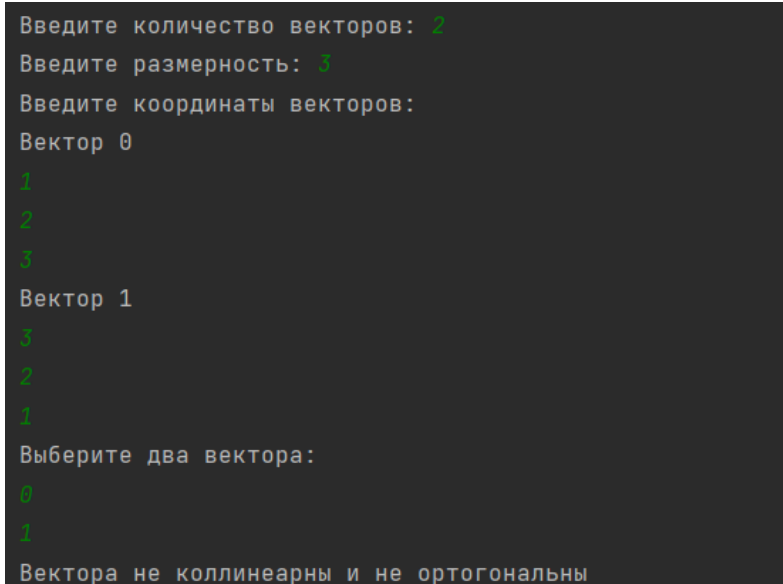
    System.out.print("Введите размерность: ");
    try {
        dimension = scanner.nextInt();
    }
    catch (Exception e){
        System.out.println("error");
        dimension = -1;
    }
}

```

```

Vector[] vector_array = new Vector[number];
Vector first, second;
double[] coords = new double[dimension];
System.out.println("Введите координаты векторов: ");
for (int i = 0; i < number; i++){
    System.out.println("Вектор " + i);
    for (int j = 0; j < dimension; j++) {
        try {
            coords[j] = scanner.nextDouble();
        } catch (Exception e){
            System.out.println("error");
        }
    }
    vector_array[i] = new Vector(coords);
}
System.out.println("Выберите два вектора:");
try {
    first = vector_array[scanner.nextInt()];
} catch (Exception e){
    first = new Vector(dimension);
    System.out.println("error");
}
try {
    second = vector_array[scanner.nextInt()];
} catch (Exception e){
    second = new Vector(dimension);
    System.out.println("error");
}
isCollinear_Orthogonal(first, second);
}

```



```

Введите количество векторов: 2
Введите размерность: 3
Введите координаты векторов:
Вектор 0
1
2
3
Вектор 1
3
2
1
Выберите два вектора:
0
1
Вектора не коллинеарны и не ортогональны

```

Рисунок 1 – Результат работы программы

Код программы:

```
public class VectorR3 {

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

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

    public VectorR3(){
    }

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

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

    public double getX() {
        return x;
    }

    public double getY() {
        return y;
    }

    public double getZ() {
        return z;
    }

    public double mod(){
        try {
            return Math.sqrt(x * x + y * y + z * z);
        }
        catch (Exception e){
            System.out.println("error");
            return -1;
        }
    }

    public double scalar(VectorR3 v){ //скалярное произведение
        try {
            return this.x * v.x + this.y * v.y + this.z * v.z;
        }
        catch (Exception e){
            System.out.println("error");
        }
    }
}
```

```

        return -1;
    }
}

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

public boolean is_Intersect(VectorR3 v){
    // Система уравнений:
    // this.x1 * t + this.x2 * (1 - t) = v.x1 * s + v.x2 * (1 - s)
    // this.y1 * t + this.y2 * (1 - t) = v.y1 * s + v.y2 * (1 - s)
    // Находим t, s, подставляем в:
    // this.z1 * t + this.z2 * (1 - t) = v.z1 * s + v.z2 * (1 - s)
    // Если получилось равенство, то отрезки (вектора) пересекаются
    try {
        double s = ((this.x2 - v.x2) * this.y + (v.y2 - this.y2) * this.x) / (this.x * v.y - this.y * v.x);
        double t = (s * v.x + this.x2 - v.x2) / this.x;
        double eq1 = this.z1 * t + this.z2 * (1 - t);
        double eq2 = v.z1 * s + v.z2 * (1 - s);

        return eq1 == eq2;
    }
    catch (Exception e){
        System.out.println("error");
        return false;
    }
}

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

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

}

public static void isCoplanar(VectorR3 v1, VectorR3 v2, VectorR3 v3){
    double m = v1.getX() * v2.getY() * v3.getZ() + v1.getY() * v2.getZ() * v3.getX()
        + v1.getZ() * v2.getX() * v3.getY() - v1.getZ() * v2.getY() * v3.getX()
        - v1.getX() * v2.getZ() * v3.getY() - v1.getY() * v2.getX() * v3.getZ();
    if (m == 0){
        System.out.println("Вектора компланарны");
    } else {
        System.out.println("Вектора не компланарны");
    }
}

```

```

}

public static void main(String[] args){
    Scanner scanner = new Scanner(System.in);
    int number;
    System.out.print("Введите количество векторов: ");
    try {
        number = scanner.nextInt();
    } catch (Exception e){
        System.out.println("error");
        number = -1;
    }
    VectorR3[] vector_array = new VectorR3[number];
    Random random = new Random();
    double x1, x2, y1, y2, z1, z2;
    System.out.println("Введите координаты векторов: ");
    for (int i = 0; i < number; i++){
        x1 = random.nextDouble();
        x2 = random.nextFloat();
        y1 = random.nextFloat();
        y2 = random.nextFloat();
        z1 = random.nextFloat();
        z2 = random.nextFloat();
        vector_array[i] = new VectorR3(x1, x2, y1, y2, z1, z2);
    }

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

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

```

```

Введите количество векторов: 5
Введите координаты векторов:
Вектор 0: VectorR3: x1 = 0.6842532121120312, x2 = 0.28175705671310425, y1 = 0.9221724271774292, y2 = 0.38307541608810425, z1 = 0.8355540037155151, z2 = 0.1076766848564148.
Вектор 1: VectorR3: x1 = 0.5125834076631008, x2 = 0.3880281448364258, y1 = 0.9146167039871216, y2 = 0.5121161937713623, z1 = 0.1874597668647766, z2 = 0.2771315574645996.
Вектор 2: VectorR3: x1 = 0.8124511998153472, x2 = 0.13411003351211548, y1 = 0.6970949172973633, y2 = 0.2632228136062622, z1 = 0.9216728210449219, z2 = 0.15629583597183228.
Вектор 3: VectorR3: x1 = 0.14608450263892447, x2 = 0.20827674865722656, y1 = 0.39262497425079346, y2 = 0.2320818305015564, z1 = 0.8808506727218628, z2 = 0.6486178636550903.
Вектор 4: VectorR3: x1 = 0.14079631673634385, x2 = 0.26780402660369873, y1 = 0.15552836656570435, y2 = 0.06202572584152222, z1 = 0.960381269454956, z2 = 0.7789296507835388.
Вектора 0, 1, 2: Вектора не компланарны
Вектора 0, 1, 3: Вектора не компланарны
Вектора 0, 1, 4: Вектора не компланарны
Вектора 0, 2, 3: Вектора не компланарны
Вектора 0, 2, 4: Вектора не компланарны
Вектора 0, 3, 4: Вектора не компланарны
Вектора 1, 2, 3: Вектора не компланарны
Вектора 1, 2, 4: Вектора не компланарны
Вектора 1, 3, 4: Вектора не компланарны
Вектора 2, 3, 4: Вектора не компланарны
Process finished with exit code 0

```

Рисунок 2 – Результат работы программы

Задание:

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

Ход работы:

Код программы:

```
Введите количество покупателей: 3
Покупатель 0:
Введите фамилию: Аветисян
Введите имя: Арсений
Введите отчество: Андреевич
Введите номер карты: 12345
Введите номер счета: 111111111
Покупатель 1:
Введите фамилию: Петров
Введите имя: Василий
Введите отчество: Петрович
Введите номер карты: 111111111
Введите номер счета: 111111111
Покупатель 2:
Введите фамилию: Аннушкин
Введите имя: Антон
Введите отчество: Антонович
Введите номер карты: 1111111
Введите номер счета: 1111111111
Сортировка:
Customer{id=1, surname='Аветисян', name='Арсений', lastname='Андреевич', credit_card=12345, bank_acc=111111111}
Customer{id=3, surname='Аннушкин', name='Антон', lastname='Антонович', credit_card=1111111, bank_acc=111111111}
Customer{id=2, surname='Петров', name='Василий', lastname='Петрович', credit_card=111111111, bank_acc=111111111}
Карты из диапазона:
Customer{id=1, surname='Аветисян', name='Арсений', lastname='Андреевич', credit_card=12345, bank_acc=111111111}
Customer{id=3, surname='Аннушкин', name='Антон', lastname='Антонович', credit_card=1111111, bank_acc=111111111}
```

Рисунок 3 – Результат работы программы

Код программы:

```
Введите количество элементов: 3
Введите элементы: 1
2
3
Введите второй массив: 4
5
6
One_Dim_Array{data=[1, 2, 3]} One_Dim_Array{data=[4, 5, 6]}
Сумма: One_Dim_Array{data=[5, 7, 9]}
Разность: One_Dim_Array{data=[-3, -3, -3]}
Произведение: One_Dim_Array{data=[4, 10, 18]}
```

Рисунок 4 – Результат работы программы

Задание:

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.

2. В каждой строке стихотворения Александра Блока найти и заменить заданную подстроку на подстроку иной длины.

Ход работы:

Код программы:

```
public static void main(String[] args) throws IOException {
    Path in_path = Paths.get("./in_file.txt");
    Path out_path = Paths.get("./out_file.txt");

    Files.deleteIfExists(in_path);
    Files.deleteIfExists(out_path);

    Files.createFile(in_path);
    Files.createFile(out_path);

    String str_1 = "привет";
    String str_2 = ",";

    Scanner scanner = new Scanner(System.in);
    while(true) {
        String line = scanner.nextLine();
        if(line.isEmpty()){
            break;
        }
        int index = 0;
        try {
            Files.write(in_path, Collections.singleton(line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
        while (index != -1) {
            index = line.indexOf(str_1);
            if (index != -1){
                line = line.replace(str_1, str_2);
            }
        }
        try {
            Files.write(out_path, Collections.singleton(line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```

}
}
}

```

```

Всё это было, было, былопривет
Свершился дней круговорот.
Какая ложь, какая сила
Тебя, прошедшее, вернет?
В час утра, чистый и хрустальныйпривет
У стен Московского Кремляпривет
Восторг души первоначальный
Вернет ли мне моя земля?
Иль в ночь на Пасху, над Невойпривет
Под ветром, в стужу, в ледоход –
Старуха нищая клюкою
Мой труп спокойный шевельнет?
Иль на возлюбленной поляне
Под шелест осени седой
Мне тело в дождевом тумане
Расклевает коршун молодой?
Иль просто в час тоски беззвезднойпривет
В каких-то четырех стенахпривет
С необходимостью железной
Усну на белых простынях?
И в новой жизни, непохожейпривет
Забуду прежнюю мечтупривет
И буду так же помнить дождейпривет
Как нынче помню Калиту?
Но верю – не пройдет бесследно
Всё, что так страстно я любилпривет
Весь трепет этой жизни беднойпривет
Весь этот непонятный пыл!
1989 г.

```

Рисунок 5 – Входной файл

	English.txt	Russian.txt	English.txt	English.txt
1			Всё это было, было, было,	
2			Свершился дней круговорот.	
3			Какая ложь, какая сила	
4			Тебя, прошедшее, вернет?	
5			В час утра, чистый и хрустальный,	
6			У стен Московского Кремля,	
7			Восторг души первоначальный	
8			Вернет ли мне моя земля?	
9			Иль в ночь на Пасху, над Невой,	
10			Под ветром, в стужу, в ледоход –	
11			Старуха нищая клюкою	
12			Мой труп спокойный шевельнет?	
13			Иль на возлюбленной поляне	
14			Под шелест осени седой	
15			Мне тело в дождевом тумане	
16			Расклевает коршун молодой?	
17			Иль просто в час тоски беззвездной,	
18			В каких-то четырех стенах,	
19			С необходимостью железной	
20			Усну на белых простынях?	
21			И в новой жизни, непохожей,	
22			Забуду прежнюю мечту,	
23			И буду так же помнить дождей,	
24			Как нынче помню Калиту?	
25			Но верю – не пройдет бесследно	
26			Всё, что так страстно я любил,	
27			Весь трепет этой жизни бедной,	
28			Весь этот непонятный пыл!	
29			1989 г.	

Рисунок 6 – Выходной файл

3. В каждой строке найти слова, начинающиеся с гласной буквы.

Ход работы:

Код программы:

```
public static void main(String[] args) throws IOException {
    Path in_path = Paths.get("./in1_file.txt");
    Path out_path = Paths.get("./out1_file.txt");

    Files.deleteIfExists(in_path);
    Files.deleteIfExists(out_path);

    Files.createFile(in_path);
    Files.createFile(out_path);

    String str = "аеёиоуыэюя";

    Scanner scanner = new Scanner(System.in);
    while(true) {
        String line = scanner.nextLine();
        if(line.isEmpty()){
            break;
        }

        try {
            Files.write(in_path, Collections.singleton(line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }

        StringBuilder out_line = new StringBuilder();
        String[] words;
        words = line.split(" ");

        for (String word : words){
            for (Character letter : str.toCharArray()){
                if (word.toLowerCase(Locale.ROOT).startsWith(letter.toString())){
                    out_line.append(" ").append(word);
                }
            }
        }

        try {
            Files.write(out_path, Collections.singleton(out_line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```

1  Всё это было, было, было,
2  Свершился дней круговорот.
3  Какая ложь, какая сила
4  Тебя, прошедшее, вернет?
5  В час утра, чистый и хрустальный,
6  У стен Московского Кремля,
7  Восторг души первоначальный
8  Вернет ли мне моя земля?
9  Иль в ночь на Пасху, над Невой,
10 Под ветром, в стужу, в ледоход –
11 Старуха нищая клюкою
12 Мой труп спокойный шевельнет?
13 Иль на возлюбленной поляне
14 Под шелест осени седой
15 Мне тело в дождевом тумане
16 Расклетет коршун молодой?
17 Иль просто в час тоски беззвездной,
18 В каких-то четырех стенах,
19 С необходимостью железной
20 Усну на белых простынях?
21 И в новой жизни, непохожей,
22 Забуду прежнюю мечту,
23 И буду так же помнить дождей,
24 Как нынче помню Калиту?
25 Но верю – не пройдет бесследно
26 Всё, что так страстно я любил,
27 Весь трепет этой жизни бедной,
28 Весь этот непонятный пыл!

```

Рисунок 7 – Входной файл

```

это

утра, и
у

Иль

Иль
осени

Иль

Усну
и

и

я
этой
этот

```

Рисунок 8 – Выходной файл

1. Прочитать текст Java-программы и записать в другой файл в обратном порядке символы каждой строки.

Ход работы:

Код программы:

```
public static void main(String[] args) throws IOException {
    Path in_path = Paths.get("./in_java.txt");
    Path out_path = Paths.get("./out_java.txt");

    Files.deleteIfExists(in_path);
    Files.deleteIfExists(out_path);

    Files.createFile(in_path);
    Files.createFile(out_path);

    Scanner scanner = new Scanner(System.in);
    while(true) {
        String line = scanner.nextLine();
        if(line.isEmpty()){
            break;
        }

        try {
            Files.write(in_path, Collections.singleton(line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }

        StringBuilder out_line = new StringBuilder(line).reverse();

        try {
            Files.write(out_path, Collections.singleton(out_line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```
1 public static void main(String[] args){
2 // Var 2.2, 2.3 Ввести с консоли n целых чисел и поместить их в массив.
3 // На консоль вывести: наибольшее и наименьшее число / числа, которые делятся на 3 или на 9
4 int n;
5 Scanner scanner = new Scanner(System.in);
6 System.out.print("Введите количество чисел: ");
7 if (scanner.hasNextInt()){
8     n = scanner.nextInt();
9 } else {
10     System.out.println("error");
11     n = - 1;
12 }
13 }
```

Рисунок 9 – Входной файл

```

1  [Osgra ][gnirtS(niam diov citats cilbup
2  .виссам в хи ьтитсепоп и лесич хылец п илоснок с итсевВ 3.2 ,2.2 raV //
3  9 ан или 3 ан ястялед еыроток ,алсич / олсич еешьнемиан и еешьлобиан :итсевыв ьлоснок aH //
4  ;n tni
5  ;)ni.metsyS(rennacS wen = rennacS rennacS
6  ;)" :лесич овтсечилок етидевВ"(tnirp.tuo.metsyS
7  {})(tnItxeNsah.rennacs( fi
8  ;)(tnItxen.rennacs = n
9  { esle }
10 ;)"porre"(nltirp.tuo.metsyS
11 ;1 - = n
12 }

```

Рисунок 10 – Выходной файл

2. Прочитать текст Java-программы и в каждом слове длиннее двух символов все строчные символы заменить прописными.

Ход работы:

Код программы:

```

public static void main(String[] args) throws IOException {
    Path in_path = Paths.get("./in1_java.txt");
    Path out_path = Paths.get("./out1_java.txt");

    Files.deleteIfExists(in_path);
    Files.deleteIfExists(out_path);

    Files.createFile(in_path);
    Files.createFile(out_path);

    Scanner scanner = new Scanner(System.in);
    while(true) {
        String line = scanner.nextLine();
        if(line.isEmpty()){
            break;
        }

        try {
            Files.write(in_path, Collections.singleton(line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }

        StringBuilder out_line = new StringBuilder();
        String[] words;
        words = line.split(" ");

        for (String word : words){
            if (word.length() > 2){
                word = word.toUpperCase(Locale.ROOT);
            }
            out_line.append(" ").append(word);
        }

        try {
            Files.write(out_path, Collections.singleton(out_line), StandardCharsets.UTF_8,
StandardOpenOption.APPEND);

```

```

    } catch (IOException e) {
        System.out.println(e.getMessage());
    }
}
}

```

```

1 | public static void main(String[] args){
2 |     // Var 2.2, 2.3 Ввести с консоли n целых чисел и поместить их в массив.
3 |     // На консоль вывести: наибольшее и наименьшее число / числа, которые делятся на 3 или на 9
4 |     int n;
5 |     Scanner scanner = new Scanner(System.in);
6 |     System.out.print("Введите количество чисел: ");
7 |     if (scanner.hasNextInt()){
8 |         n = scanner.nextInt();
9 |     } else {
10 |         System.out.println("error");
11 |         n = - 1;
12 |     }
13 |     ArrayList<Integer> list = new ArrayList<>();
14 |     for (int i = 1; i <= n; i++){
15 |         System.out.print("Введите число " + i + ": ");
16 |         if (scanner.hasNextInt()){
17 |             list.add(scanner.nextInt());
18 |         } else {
19 |             System.out.print("error");
20 |         }
21 |     }
22 |     System.out.println("max: " + Collections.max(list));
23 |     System.out.println("min: " + Collections.min(list));
24 |     for (int number : list){
25 |         if (number % 3 == 0){
26 |             System.out.print(number + " ");
27 |         }
28 |     }
29 | }
30 |

```

Рисунок 11 – Входной файл

```

PUBLIC STATIC VOID MAIN(STRING[] ARGS){
    // VAR 2.2, 2.3 ВВЕСТИ с КОНСОЛИ n ЦЕЛЫХ ЧИСЕЛ и ПОМЕСТИТЬ их в МАССИВ.
    // На КОНСОЛЬ ВЫВЕСТИ: НАИБОЛЬШЕЕ и НАИМЕНЬШЕЕ ЧИСЛО / ЧИСЛА, КОТОРЫЕ ДЕЛЯТСЯ на 3 ИЛИ на 9
    INT n;
    SCANNER SCANNER = NEW SCANNER(SYSTEM.IN);
    SYSTEM.OUT.PRINT("ВВЕДИТЕ КОЛИЧЕСТВО ЧИСЕЛ: ");
    IF (SCANNER.HASNEXTINT()){
        n = SCANNER.NEXTINT();
    } ELSE {
        SYSTEM.OUT.PRINTLN("ERROR");
        n = - 1;
    }
    ARRAYLIST<INTEGER> LIST = NEW ARRAYLIST<>();
    FOR (INT i = 1; i <= n; I++){
        SYSTEM.OUT.PRINT("ВВЕДИТЕ ЧИСЛО " + i + ": ");
        IF (SCANNER.HASNEXTINT()){
            LIST.ADD(SCANNER.NEXTINT());
        } ELSE {
            SYSTEM.OUT.PRINT("ERROR");
        }
    }
    SYSTEM.OUT.PRINTLN("MAX: " + COLLECTIONS.MAX(LIST));
    SYSTEM.OUT.PRINTLN("MIN: " + COLLECTIONS.MIN(LIST));
    FOR (INT NUMBER : LIST){
        IF (NUMBER % 3 == 0){
            SYSTEM.OUT.PRINT(NUMBER + " ");
        }
    }
}

```

Рисунок 12 – Выходной файл

Вывод: лабораторная работа выполнена в соответствии с заданием и вариантом.