



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 № 17

Название: Исключения и файлы

Дисциплина: Языки программирования для работы с большими данными

Студент

ИУ6-23М

(Группа)

(Подпись, дата)

М.О. Усманов

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

П.В. Степанов

(И.О. Фамилия)

Москва, 2022

Цель работы

Получение первичных навыков обработки исключительных ситуаций в языке программирования Java. Изучение основ работы с файлами и файловой системой.

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.

Определить класс Дробь в виде пары (m,n). Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения и деления дробей. Объявить массив из k дробей, ввести/вывести значения для массива дробей. Создать массив объектов и передать его в метод, который изменяет каждый элемент массива с четным индексом путем добавления следующего за ним элемента массива.

Листинг 1 – Код программы класса «Fraction» первого задания

```
public class Fraction {
    int m, n;

    public Fraction() {
        m = 0;
        n = 1;
    }

    public Fraction(int m) {
        this.m = m;
        n = 1;
    }

    public Fraction(int m, int n) {
        this.m = m;
        this.n = n;

        if (n == 0) throw new ArithmeticException("denominator == 0");
    }
}
```

```

    }

    public Fraction(Fraction a) {
        m = a.m;
        n = a.n;

        if (n == 0) throw new ArithmeticException("denominator == 0");
    }

    // Сложение
    public Fraction Add(Fraction addee) {
        // Создаем новый объект класса Fraction
        Fraction sum = new Fraction();
        // Находим общий знаменатель
        int new_n = this.n * addee.n;
        // Присваиваем его новому объекту-сумме
        sum.n = new_n;
        // Получаем сумму числителей
        int new_m = m * addee.n + addee.m * n;
        // Присваиваем её новому объекту-сумме
        sum.m = new_m;

        if (sum.n == 0) throw new ArithmeticException("denominator == 0");

        return sum;
    }

    // Умножение
    public Fraction Multiply(Fraction multiplee) {
        // Создаем новый объект класса Fraction
        // Сразу с умножением
        Fraction product = new Fraction(
            m * multiplee.m,
            n * multiplee.n
        );

        if (product.n == 0) throw new ArithmeticException("denominator == 0");

        return product;
    }

    // Инверсия знака
    public Fraction Invert() {
        // Создаем новый объект класса Fraction
        Fraction inverted = new Fraction(-m, n);

        return inverted;
    }

    // Вычитание
    public Fraction Subtract(Fraction subtractee) {
        // Инвертируем вычитаемое, чтобы провести это как сложение
        Fraction inverted_subtractee = new Fraction(subtractee.Invert());

        // Складываем две дроби
        Fraction sub = this.Add(inverted_subtractee);
    }

```

```

        return sub;
    }

    // Переворачивание дроби
    public Fraction Switch() {
        // Создаем новый объект класса Fraction
        // Переворачиваем дробь
        Fraction switched = new Fraction(n, m);

        if (switched.n == 0) throw new ArithmeticException("denominator == 0");

        return switched;
    }

    // Деление
    public Fraction Divide(Fraction dividee) {
        // Переворачиваем множитель для выполнения деления
        Fraction switched_multiplicator = this.Switch();

        // Умножаем на перевернутую дробь
        Fraction product = this.Multiply(switched_multiplicator);

        return product;
    }

    // Вывод дроби
    public String toString() {
        String fraction = m + "/" + n;
        return fraction;
    }
}

```

Листинг 2 – Код основной программы задания 1 (Подзадача 1)

```

import java.util.InputMismatchException;
import java.util.Scanner;

public class App {
    public static void main(String[] args) throws Exception {
        System.out.println("Введите k:");

        Scanner console = new Scanner(System.in);

        try {

            int k = console.nextInt();
            Fraction[] fractions = new Fraction[k];

            for (int i = 0; i < k; i++) {
                System.out.println("Введите m для дроби " + (i + 1) +
":");
                int m = console.nextInt();
                System.out.println("Введите n для дроби " + (i + 1) +
":");
                int n = console.nextInt();
            }
        }
    }
}

```

```

        fractions[i] = new Fraction(m, n);
    }

    PrintArray(fractions);

    for (int i = 0; i < k - 1; i += 2) {
        fractions[i] = fractions[i].Add(fractions[i + 1]);
    }

    PrintArray(fractions);

    } catch (InputMismatchException e) {

        System.out.println("Неверный формат введенных данных:
допустимы только целые числа");

    } catch (NegativeArraySizeException e) {

        System.out.println("Попытка создать массив отрицательной
размерности");

    } catch (OutOfMemoryError e) {

        System.out.println("Не хватает памяти для массива");

    } catch (ArithmeticException e) {

        System.out.println(e);

    } catch (Exception e) {

        System.out.println("Что-то пошло не так");

    } finally {

        console.close();

    }

}

private static void PrintArray(Fraction[] array) {
    System.out.print("[");
    for (int i = 0; i < array.length; i++) {
        System.out.print(array[i].toString());
        if (i != array.length - 1) System.out.print(", ");
    }
    System.out.print("]\n");
}
}

```

Приведем результаты обработки нескольких исключительных ситуаций.

```
CodeDetailsInExceptionMessages' -cp 'E:\Project
Введите k:
5
Введите m для дроби 1:
4
Введите n для дроби 1:
0
java.lang.ArithmeticException: denominator == 0
PS E:\Projects\Java\Repo\lab_5_var_1_7>
java.lang.ArithmeticException: denominator == 0
PS E:\Projects\Java\Repo\lab_5_var_1_7> e;; cd 'e:\Projects\Java\Repo\lab_5_va
CodeDetailsInExceptionMessages' -cp 'E:\Projects\Java\Repo\lab_5_var_1_7\bin
Введите k:
ф
Неверный формат введенных данных: допустимы только целые числа
PS E:\Projects\Java\Repo\lab_5_var_1_7>
```

Рисунок 1 – Результаты выполнения варианта задания 1

Определить класс Комплекс. Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения, деления, присваивания комплексных чисел. Создать два вектора размерности n из комплексных координат. Передать их в метод, который выполнит их сложение.

Листинг 3 – Код основной программы

```
import java.util.InputMismatchException;
import java.util.Scanner;
import java.util.Vector;

public class App {
    public static void main(String[] args) throws Exception {

        System.out.println("Введите n:");
        Scanner console = new Scanner(System.in);

        try {

            int n = console.nextInt();

            Vector<Complex> complex_1 = new Vector<>();

            for (int i = 0; i < n; i++) {
                System.out.println("Введите re для числа " + (i + 1) + "в
векторе 1:");
                int re = console.nextInt();
                System.out.println("Введите im для числа " + (i + 1) + "в
векторе 1:");
                int im = console.nextInt();

                complex_1.add(new Complex(re, im));
            }

            Vector<Complex> complex_2 = new Vector<>();
```

```

        for (int i = 0; i < n; i++) {
            System.out.println("Введите re для числа " + (i + 1) + " в
векторе 2:");
            int re = console.nextInt();
            System.out.println("Введите im для числа " + (i + 1) + " в
векторе 2:");
            int im = console.nextInt();

            complex_2.add(new Complex(re, im));
        }

        PrintVector(complex_1);
        PrintVector(complex_2);

        Vector<Complex> result = SumVectors(complex_1, complex_2);

        PrintVector(result);

    } catch (InputMismatchException e) {

        System.out.println("Неверный формат введенных данных:
допустимы только целые числа");

    } catch (OutOfMemoryError e) {

        System.out.println("Не хватает памяти для списка");

    } catch (ArithmeticException e) {

        System.out.println(e);

    } catch (Exception e) {

        System.out.println("Что-то пошло не так");

    } finally {

        console.close();

    }

}

private static void PrintVector(Vector<Complex> vector) {
    System.out.print("[");
    for (int i = 0; i < vector.size(); i++) {
        System.out.print(vector.get(i).toString());
        if (i != vector.size() - 1) System.out.print(", ");
    }
    System.out.print("]\n");
}

private static Vector<Complex> SumVectors(Vector<Complex> a,
Vector<Complex> b) {
    Vector<Complex> result = new Vector<>();

```

```

        for (int i = 0; i < a.size(); i++) {
            result.add(a.get(i).Add(b.get(i)));
        }
        return result;
    }
}

```

Листинг 4 – Код класса «Complex»

```

public class Complex {
    int re, im;

    public Complex() {
        this.re = 0;
        this.im = 0;
    }

    public Complex(int re) {
        this.re = re;
        this.im = 0;
    }

    public Complex(int re, int im) {
        this.re = re;
        this.im = im;
    }

    public Complex(Complex complex) {
        this.re = complex.re;
        this.im = complex.im;
    }

    // Сложение
    public Complex Add(Complex addee) {
        // Создаем новый объект класса Complex
        Complex sum = new Complex(
            this.re + addee.re,
            this.im + addee.im
        );

        return sum;
    }

    // Вычитание
    public Complex Subtract(Complex subtractee) {
        Complex sub = new Complex(
            this.re - subtractee.re,
            this.im - subtractee.im
        );

        return sub;
    }

    // Умножение
    public Complex Multiply(Complex multee) {

```



```

    int k1 = multee.re * (this.re + this.im);
    int k2 = this.re * (multee.im - multee.re);
    int k3 = this.im * (multee.re + multee.im);

    int real_part = k1 - k3;
    int imaginary_part = k1 + k2;

    Complex mul = new Complex(real_part, imaginary_part);

    return mul;
}

// Деление
public Complex Divide(Complex dividee) {

    int re_num = (this.re * dividee.re) + (this.im * dividee.im);
    int re_den = (dividee.re * dividee.re + dividee.im * dividee.im);

    int im_num = (this.im * dividee.re) - (this.re * dividee.im);
    int im_den = (dividee.re * dividee.re + dividee.im * dividee.im);

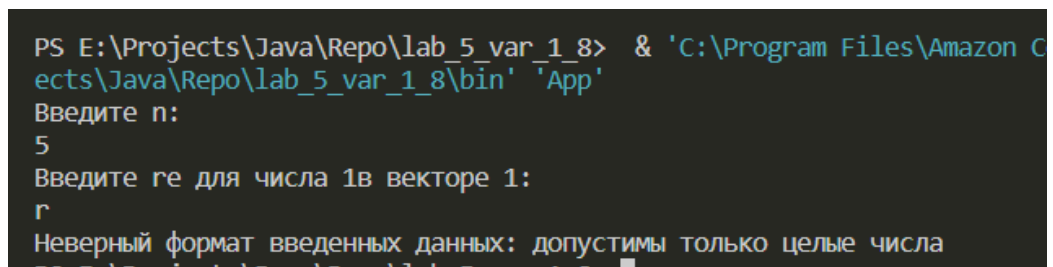
    Complex div = new Complex(
        re_num / re_den,
        im_num / im_den
    );

    return div;
}

// Присвоение
public void SetComplex(int re, int im) {
    this.re = re;
    this.im = im;
}

// Вывод числа
public String toString() {
    String complex = re + " + i" + im;
    return complex;
}
}

```



```

PS E:\Projects\Java\Repo\lab_5_var_1_8> & 'C:\Program Files\Amazon C
ects\Java\Repo\lab_5_var_1_8\bin' 'App'
Введите n:
5
Введите re для числа 1в векторе 1:
г
Неверный формат введенных данных: допустимы только целые числа
PS E:\Projects\Java\Repo\lab_5_var_1_8>

```

Рисунок 2 – Результаты обработки исключений

Задание 2.

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

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

Phone: id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Дебет, Кредит, Время городских и междугородных разговоров. Создать массив объектов. Вывести: а) сведения об абонентах, у которых время внутригородских разговоров превышает заданное; б) сведения об абонентах, которые пользовались междугородной связью; в) сведения об абонентах в алфавитном порядке.

Листинг 5 – Код измененного класса «Phone» с добавленными вызовами исключений

```
public class Phone {
    int id;
    String first_name, last_name, patronymic;
    String address;
    long card_number;
    double debit, credit;
    int intercity_calls, intracity_calls; // Междугородние звонки и
    внутригородские соотв.

    public int getId() {
        return id;
    }

    public String getLastName() {
        return last_name;
    }

    public String getFirstName() {
        return first_name;
    }

    public String getPatronymic() {
        return patronymic;
    }

    public String getAddress() {
        return address;
    }

    public long getCardNumber() {
        return card_number;
    }
}
```

```

public double getDebit() {
    return debit;
}

public double getCredit() {
    return credit;
}

public int getIntercityCalls() {
    return intercity_calls;
}

public int getIntracityCalls() {
    return intracity_calls;
}

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

public void setCardNumber(long card_number) throws Exception {
    if (card_number == 0) throw new Exception("Card Number can't be
zero");

    int length = (int) (Math.log10(card_number) + 1);
    if (length != 16) throw new Exception("Invalid card number format");

    this.card_number = card_number;
}

public void setCredit(double credit) {
    this.credit = credit;
}

public void setDebit(double debit) {
    this.debit = debit;
}

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

public void setIntercityCalls(int intercity_calls) {
    this.intercity_calls = intercity_calls;
}

public void setIntracityCalls(int intracity_calls) {
    this.intracity_calls = intracity_calls;
}

public void setLastName(String last_name) throws Exception {
    checkStringCorrectness(last_name);
    this.last_name = last_name;
}

```

```

public void setFirstName(String name) throws Exception {
    checkStringCorrectness(name);
    this.first_name = name;
}

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

@Override
public String toString() {
    String result = "{ " +
        "ID: " + this.id + "\n" +
        "Last Name: " + this.last_name + "\n" +
        "First Name: " + this.first_name + "\n" +
        "Patronymic: " + this.patronymic + "\n" +
        "Address: " + this.address + "\n" +
        "Card Number: " + this.card_number + "\n" +
        "Debit: " + this.debit + "\n" +
        "Credit: " + this.credit + "\n" +
        "Intercity Calls: " + this.intercity_calls + " min. \n" +
        "Intracity Calls: " + this.intracity_calls + " min. " +
        "}";
    return result;
}

private void checkStringCorrectness(String value) throws Exception {
    if (value.equals("")) throw new Exception("Invalid value");
}
}

```

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

Car: id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер. Создать массив объектов. Вывести: а) список автомобилей заданной марки; б) список автомобилей заданной модели, которые эксплуатируются больше n лет; в) список автомобилей заданного года выпуска, цена которых больше указанной.

Листинг 6 – Код измененного класса «Car» с добавленными вызовами исключений

```

import java.util.regex.Pattern;

public class Car {

```

```

int id;
String brand;
String model;
int year;
String color;
int price;
String number;

public void setId(int id) throws Exception {
    if (id <= 0) throw new Exception("ID can't be 0 or negative");
    this.id = id;
}

public void setBrand(String brand) throws Exception {
    checkStringCorrectness(brand);
    this.brand = brand;
}

public void setModel(String model) throws Exception {
    checkStringCorrectness(model);
    this.model = model;
}

public void setColor(String color) throws Exception {
    checkStringCorrectness(color);
    this.color = color;
}

public void setPrice(int price) throws Exception {
    if (price < 0) throw new Exception("Price can't be negative");
    this.price = price;
}

public void setNumber(String number) throws Exception {
    if (!Pattern.matches("\\D{1}\\d{3}\\D{2}", number)) throw new
Exception("Invalid car number format");
    this.number = number;
}

public void setYear(int year) {
    this.year = year;
}

public int getId() {
    return id;
}

public String getBrand() {
    return brand;
}

public String getModel() {
    return model;
}

public String getColor() {
    return color;
}

```

```

    }

    public String getNumber() {
        return number;
    }

    public int getPrice() {
        return price;
    }

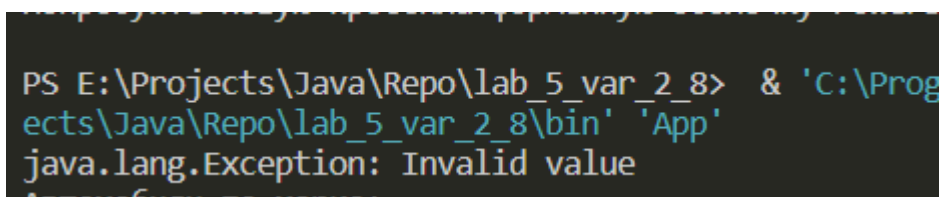
    public int getYear() {
        return year;
    }

    @Override
    public String toString() {
        String result = "{ " +
            "ID: " + this.id + "\n" +
            "Brand: " + this.brand + "\n" +
            "Model: " + this.model + "\n" +
            "Year: " + this.year + "\n" +
            "Color: " + this.color + "\n" +
            "Price: " + this.price + "\n" +
            "Number: " + this.number + "\n" +
            "}";
        return result;
    }

    private void checkStringCorrectness(String value) throws Exception {
        if (value.equals("")) throw new Exception("Invalid value");
    }
}

```

Приведем результаты выполнения работы программы при введенных некорректных данных.



```

PS E:\Projects\Java\Repo\lab_5_var_2_8> & 'C:\Projects\Java\Repo\lab_5_var_2_8\bin' 'App'
java.lang.Exception: Invalid value

```

Рисунок 3 – Обработка исключения

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.

– в каждом слове стихотворения Николая Заболоцкого заменить первую букву слова на прописную.

– определить частоту повторяемости букв и слов в стихотворении Александра Пушкина.

Листинг 7 – Код основной программы

```
import java.io.File; // Import the File class
import java.io.FileNotFoundException; // Import this class to handle
errors
import java.io.FileWriter;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.util.HashMap;
import java.util.Scanner; // Import the Scanner class to read text files

public class App {
    public static void main(String[] args) throws Exception {

        // Открываем файл стихотворение Заболоцкого
        try {
            System.out.println("Отрабатываем стихотворение Н.
Заболоцкого");

            File poem = new File("data\\poem_orig.txt");

            String output_poem = "data\\poem_new.txt";

            File new_file = new File(output_poem);
            if (new_file.createNewFile()) {
                System.out.println("Файл создан: " + new_file.getName());
            } else {
                System.out.println("Файл уже существует");
            }

            FileWriter writer = new FileWriter(output_poem,
StandardCharsets.UTF_8);

            Scanner reader = new Scanner(poem, "UTF-8");

            while (reader.hasNextLine()) {
                System.out.print(".");
                String data = reader.nextLine();
                writer.write(capitalizeWords(data));
                writer.write("\n");
            }
            System.out.print("\n");
        }
    }
}
```

```

        System.out.println("Стихотворение Заболоцкого успешно
обработано");

        writer.close();
        reader.close();

    } catch (FileNotFoundException e) {
        System.out.println("Файл не найден");
        e.printStackTrace();
    } catch (IOException e) {
        System.out.println("Произошла ошибка I/O");
        e.printStackTrace();
    }
}

// Работаем с Пушкиным
try {
    System.out.println("Отрабатываем стихотворение А. Пушкина");

    File poem = new File("data\\poem_pushkin.txt");

    HashMap<String, Integer> words = new HashMap<>();
    HashMap<Character, Integer> characters = new HashMap<>();

    Scanner reader = new Scanner(poem, "UTF-8");

    while (reader.hasNextLine()) {
        System.out.print(".");
        String data =
removeAllNonAlphaNumeric(reader.nextLine());

        String[] splitted = data.split("\\s");

        for (String w:splitted) {
            if (w.equals("")) continue;

            int count = words.containsKey(w) ? words.get(w) : 0;
            words.put(w, count + 1);

            for (Character c:w.toCharArray()) {
                int c_count = characters.containsKey(c) ?
characters.get(c) : 0;
                characters.put(c, c_count + 1);
            }
        }
        System.out.print("\n");

        System.out.println("Стихотворение Пушкина успешно
обработано");

        System.out.println("Подсчет слов:");
        words.entrySet().forEach(entry -> {
            System.out.println(entry.getKey() + " " +
entry.getValue());
        });
    }
}

```



```

        System.out.println("Подсчет СИМВОЛОВ:");
        characters.entrySet().forEach(entry -> {
            System.out.println(entry.getKey() + " " +
entry.getValue());
        });

        reader.close();
    } catch (FileNotFoundException e) {
        System.out.println("Файл не найден");
        e.printStackTrace();
    }
}

static String capitalizeWords(String str) {
    if (str.equals("")) return "";

    String[] words = str.split("\\s");
    String capitalizeWord = "";

    for(String w:words){
        String first = w.substring(0,1);
        String afterfirst = w.substring(1);
        capitalizeWord += first.toUpperCase() + afterfirst + " ";
    }

    return capitalizeWord.trim();
}

static String removeAllNonAlphaNumeric(String s) {
    if (s == null) {
        return null;
    }
    return s.replaceAll("[^\\wА-Яа-я\\s]", "");
}
}

```

Приведем результаты выполнения разработанной программы. Покажем получившиеся в результате файлы.

```

ges' '-cp' 'E:\Projects\Java\Repo\lab_5_var_3_7
Отрабатываем стихотворение Н. Заболоцкого
Файл уже существует
.....
Стихотворение Заболоцкого успешно обработано
Отрабатываем стихотворение А. Пушкина
.....
Стихотворение Пушкина успешно обработано
Подсчет слов:
мной 1
Тянулись 1
без 3
божества 1
явилась 2
прежние 1
любовь 1
мгновенье 1
чистой 2
В 3
мечты 1
забыл 1
твой 1
божество 1
Шли 1
И 7
небесные 1
настало 1
порыв 1
глуши 1
сердце 1
долго 1
Передо 1
упоенье 1

```

Рисунок 4 – Результаты подсчета слов

The screenshot shows a Java IDE with four tabs: README.md, App.java, poem_pushkin.txt, and poem_new.txt. The 'poem_new.txt' tab is active, displaying the following text:

```

data > poem_new.txt
1 Любите Живопись, Поэты!
2 Лишь Ей, Единственной, Дано
3 Души Изменчивой Приметы
4 Переносить На Полотно.
5
6 Ты Помнишь, Как Из Тьмы Былого,
7 Едва Закутана В Атлас,
8 С Портрета Рокотова Снова
9 Смотрела Струйская На Нас?
10
11 Ее Глаза – Как Два Тумана,
12 Полуулыбка, Полуплач,
13 Ее Глаза – Как Два Обмана,
14 Покрытых Иглою Неудач.
15
16 Соединенье Двух Загадок,
17 Полувосторг, Полуиспуг,
18 Безумной Нежности Припадок,
19 Предвосхищенье Смертных Мук.
20
21 Когда Потемки Наступают
22 И Приближается Гроза,
23 Со Дна Души Моей Мерцают
24 Её Прекрасные Глаза.

```

Рисунок 5 – Результат обработки файла и записи в новый

Задание 4.

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

Прочитать текст Java-программы и удалить из него все “лишние” пробелы и табуляции, оставив только необходимые для разделения операторов.

Листинг 8 – Код программы

```
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.util.List;
import java.util.ArrayList;

public class App {
    public static void main(String[] args) throws Exception {

        try {

            // Получаем путь файла
            Path inputPath = Paths.get("data\\Hello.java");

            // Читаем все строки файла с кодом программы
            List<String> strings = Files.readAllLines(inputPath,
StandardCharsets.UTF_8);
            ArrayList<String> outList = new ArrayList<String>();

            // Обрабатываем строки
            for (String s : strings) {
                String changed = s.trim().replaceAll(" +", " ");
                outList.add(changed);
            }

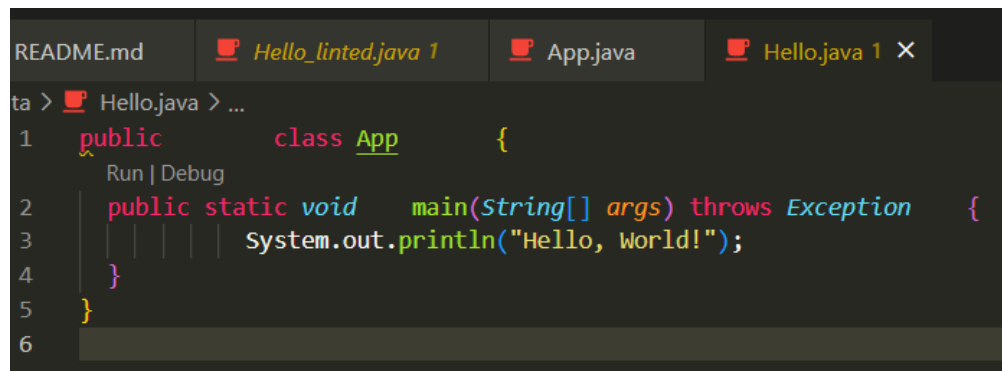
            // Создаем новую директорию, если не существует
            Files.createDirectories(Paths.get("output"));

            // Задаем путь для нового файла для вывода
            Path outputPath = Paths.get("output\\Hello_linted.java");

            // Записываем в новый файл получившийся код
            Files.write(outputPath, outList, StandardCharsets.UTF_8);

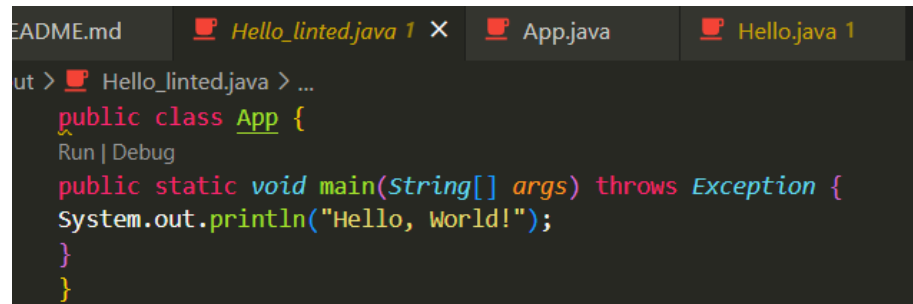
        } catch (FileNotFoundException e) {
            System.out.println("Файл не найден");
            e.printStackTrace();
        } catch (IOException e) {
            System.out.println("Произошла ошибка I/O");
            e.printStackTrace();
        }

    }
}
```



```
README.md  Hello_linted.java 1  App.java  Hello.java 1 X
ta > Hello.java > ...
1  public class App {
    Run | Debug
2  public static void main(String[] args) throws Exception {
3      System.out.println("Hello, World!");
4  }
5  }
6
```

Рисунок 6 – Исходный файл



```
README.md  Hello_linted.java 1 X  App.java  Hello.java 1
ut > Hello_linted.java > ...
public class App {
    Run | Debug
    public static void main(String[] args) throws Exception {
        System.out.println("Hello, World!");
    }
}
```

Рисунок 7 – Обработанный файл

Из текста Java-программы удалить все виды комментариев.

Листинг 9 – Код программы

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

public class App {
    public static void main(String[] args) throws Exception {

        try {

            // Получаем путь файла
            Path inputPath = Paths.get("data\\Hello.java");

            // Читаем все строки файла с кодом программы
            byte[] content = Files.readAllBytes(inputPath);
            String program = new String(content);

            // Обработываем строки - удаляем комментарии
            String out = program.trim()

            .replaceAll("/\\*(\\^*|\\[\\r\\n]|(\\|\\^*\\/|\\[\\r\\n]))*\\^*+/", "") //
Многострочные
                .replaceAll("\\/\\/\\/(.*)*", ""); // Однострочные

            // Создаем новую директорию, если не существует
            Files.createDirectories(Paths.get("output"));
```

```

        // Задаем путь для нового файла для вывода
        Path outputPath = Paths.get("output\\Hello_linted.java");

        // Записываем в новый файл получившийся код
        Files.write(outputPath, out.getBytes());

    } catch (FileNotFoundException e) {
        System.out.println("Файл не найден");
        e.printStackTrace();
    } catch (IOException e) {
        System.out.println("Произошла ошибка I/O");
        e.printStackTrace();
    }
}
}

```

```

1  public class App { // Inline comment
2      public static void main(String[] args) throws Exception {
3          // Newline comment
4          System.out.println("Hello, World!");
5
6          /* Multiline
7             comment
8             */
9      }
10 }

```

Рисунок 8 – Исходный файл

```

1  public class App {
2      public static void main(String[] args) throws Exception {
3
4          System.out.println("Hello, World!");
5
6      }
7  }
8

```

Рисунок 9 – Результирующий файл

Местоположение репозитория с файлами проекта

Файлы проекта расположены в репозитории веб-платформы для совместной разработки Github. Местоположение в репозитории:

https://github.com/s314/big-data-studies/tree/main/lab_5_var_1_7

https://github.com/s314/big-data-studies/tree/main/lab_5_var_1_8
https://github.com/s314/big-data-studies/tree/main/lab_5_var_2_7
https://github.com/s314/big-data-studies/tree/main/lab_5_var_2_8
https://github.com/s314/big-data-studies/tree/main/lab_5_var_3_7_8
https://github.com/s314/big-data-studies/tree/main/lab_5_var_4_7
https://github.com/s314/big-data-studies/tree/main/lab_5_var_4_8

Вывод

По итогам выполнения лабораторной работы были получены навыки программирования с использованием исключений в языке Java. Также был получен опыт работы с файлами и файловой системой.