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, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ 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

Название: Исключения, Файлы

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-22М (Группа) _____ (Подпись, дата) В.А. Трофимов (И.О. Фамилия)

Преподаватель _____ (Подпись, дата) П.В. Степанов (И.О. Фамилия)

Москва, 2023

Задания:

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.2.7. Повернуть матрицу на 90 (180, 270) градусов против часовой стрелки.

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

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.

3.6. В каждой строке стихотворения Анны Ахматовой подсчитать частоту повторяемости каждого слова из заданного списка и вывести эти слова в порядке возрастания частоты повторяемости.

3.7. В каждом слове стихотворения Николая Заболоцкого заменить первую букву слова на прописную.

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

4.6. Из файла удалить все слова, содержащие от трех до пяти символов, но при этом из каждой строки должно быть удалено только максимальное четное количество таких слов.

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

Код для решения задания 1:

```
package bdjava.lab5;

public class Lab5Exceptions {
    public static class EmptyLineException extends Exception {
        private String location;
        public EmptyLineException (String location) {
            super("Обязательно нужно ввести данные!");
            this.location = location;
        }
        public String getErrorLocation () {
            return location;
        }
    }
    public static class NegativeNumberException extends Exception {
        private String location;
        public NegativeNumberException(String location, String value) {
            super("Используйте пожалуйста положительные числа! Введенное значение:
" + value);
            this.location = location;
        }
        public String getErrorLocation () {
            return location;
        }
    }
}
```

Код для решения задания 1:

```
package bdjava.lab5;
import bdjava.lab3.var1.ContinuedFraction;
import bdjava.lab3.var1.Fraction;
import java.io.*;
import java.util.Scanner;

import static java.lang.System.out;

public class Part1 {
    public static void main(String[] args) {
        try {
            String fileName1 = "./src/bdjava/lab5/text1.txt";
            String fileName2 = "./src/bdjava/lab5/text2.txt";

            out.println("Создаём цепные дроби...");
            ContinuedFraction cf1 = new ContinuedFraction((short) 3, 2f);
            out.println(cf1.toString());
            ContinuedFraction cf2 = new ContinuedFraction((short) 2, 0f);
            out.println(cf2.toString());
            out.println("На ноль всё делется по законам математики.");
            out.println(cf2.calculate(false));

            out.println("Записываем в файл...");
```

```
            FileOutputStream fOutSteam = new FileOutputStream(fileName1);
            ObjectOutputStream outputObjStream = new
ObjectOutputStream(fOutSteam);
            outputObjStream.writeObject(cf1);
            outputObjStream.writeObject(cf2);
            fOutSteam.close();
            outputObjStream.close();

            out.println("Читаем из файла...");
            FileInputStream fInSteam = new FileInputStream(fileName1);
            ObjectInputStream inputObjStream = new ObjectInputStream(fInSteam);
            cf1 = (ContinuedFraction) inputObjStream.readObject();
            cf2 = (ContinuedFraction) inputObjStream.readObject();
            fOutSteam.close();
            inputObjStream.close();

            out.println("Смотрим, что причли из файла...");
            out.println(cf1.toString());
            out.println(cf2.toString());

            out.println("-------------------------------------------------");
            out.println("Создаём обычные дроби...");
            Fraction f1 = new Fraction(3, 4);
            out.println(f1.toString());
//            Fraction f2 = new Fraction(0, 32);
//            out.println(f2.toString());
//            out.println("Провоцируем ситуацию деления на 0...");
            Fraction f2 = new Fraction(2, 32);
            out.println(f2.toString());
            out.println("Делим первую дробь на вторую 0...");
            out.println(Fraction.division(f1, f2).toString());

            out.println("Записываем в файл...");
            fOutSteam = new FileOutputStream(fileName2);
            outputObjStream = new ObjectOutputStream(fOutSteam);
            outputObjStream.writeObject(f1);
            outputObjStream.writeObject(f2);
            fOutSteam.close();
            outputObjStream.close();

            out.println("Читаем из файла...");
            fInSteam = new FileInputStream(fileName2);
            inputObjStream = new ObjectInputStream(fInSteam);
            f1 = (Fraction) inputObjStream.readObject();
            f2 = (Fraction) inputObjStream.readObject();
            fOutSteam.close();
            inputObjStream.close();

            out.println("Смотрим, что причли из файла...");
            out.println(f1.toString());
            out.println(f2.toString());

        }
        catch (ArithmeticException ex) {
            out.println("Арифметическая ошибка!\nСистемная ошибка:");
            out.println(ex.getMessage());
```

```
        }
        catch (FileNotFoundException ex) {
            out.println("Файл не найден\nСистемная ошибка:");
            out.println(ex.getMessage());
        }
        catch (IOException ex) {
            out.println("Ошибка записи в поток или создания потока\nСистемная
ошибка:");
            out.println(ex.getMessage());
            out.println(ex.getStackTrace().toString());
            ex.printStackTrace();
        }
        catch (ClassNotFoundException ex) {
            out.println("Не найден класс\nСистемная ошибка:");
            out.println(ex.getMessage());
        }
    }
}
```

Код для решения задания 2:

```
package bdjava.lab5;
import bdjava.lab3.var1.ContinuedFraction;
import bdjava.lab3.var2.Phone;
import bdjava.lab3.var2.House;
import static java.lang.System.out;

import java.io.*;
import java.util.ArrayList;

public class Part2 {
    public static void main(String[] args) {
        out.println("Часть 1:");
        ArrayList<House> buf;
        ArrayList<House> l = new ArrayList<>();
        try {
            for (short i = 0; i < 20; i++)
                l.add(new House((short) (Math.random() * 100 + 1), (float)
(Math.random() * 100 + 1),
                        (short) (Math.random() * 40 + 1), (short) (Math.random() *
10 + 1), "Вешняки",
                        "Жилое строение", (short) (Math.random() * 100 + 1)));
            out.println("Сгенерированные квартиры:");
            for(House ap : l)
                out.println(ap);
            out.println();
            out.println("Поиск по номеру квартиры (3):");
            buf = House.search(l, (short)3, null, null, null);
            for(House ap : buf)
                out.println(ap);
            out.println();
            out.println("Квартиры, имеющие заданное число комнат (3) и
расположенных на этаже, " +
                    "который находится в заданном промежутке (2,25):");
            buf = House.search(l, (short)3, null, (short)2, (short)25);
```

```
            for(House ap : buf)
                out.println(ap);
            out.println();
            out.println("Квартиры, имеющие площадь, превосходящую 20 кв.м:");
            buf = House.search(l, null, 20f, null, null);
            for(House ap : buf)
                out.println(ap);
            out.println();
            String fileName = "./src/bdjava/lab5/text3.txt";
            out.println("Записываем в файл...");
            FileOutputStream fOutSteam = new FileOutputStream(fileName);
            ObjectOutputStream outputObjStream = new
ObjectOutputStream(fOutSteam);
            outputObjStream.writeObject(l.get(0));
            fOutSteam.close();
            outputObjStream.close();
            out.println("Читаем из файла...");
            FileInputStream fInSteam = new FileInputStream(fileName);
            ObjectInputStream inputObjStream = new ObjectInputStream(fInSteam);
            House q = (House) inputObjStream.readObject();
            fOutSteam.close();
            inputObjStream.close();
            out.println("Смотрим, что причли из файла...");
            out.println(q.toString());

            out.println("Часть 2:");
            ArrayList<String> names = new ArrayList<String>();
            names.add("Дима");
            names.add("Вова");
            names.add("Ваня");
            names.add("Лёня");
            names.add("Гоша");
            ArrayList<String> surname = new ArrayList<String>();
            surname.add("Иванов");
            surname.add("Борисов");
            surname.add("Вадисов");
            surname.add("Петросян");
            surname.add("Степанов");
            ArrayList<Phone> l2 = new ArrayList<>();
            for(short i=0; i<10; i++)
                l2.add(new Phone(names.get((int)(Math.random()*5)),
surname.get((int)(Math.random()*5)), "Александрович",
                        "Старый Гай 28 кв 5", "232354353", 21312, 6564, (short)
(Math.random()*300),
                        (short) (Math.random()*100)));
            out.println("Сгенерированные абоненты:");
            for(Phone ph : l2)
                out.println(ph);
            out.println();
            out.println("Сведения об абонентах, у которых время внутригородских
разговоров превышает заданное (150):");
            ArrayList<Phone> buf2 = Phone.search(l2, (short)150, null, null);
            for(Phone ph : buf2)
                out.println(ph);
            out.println();
            out.println("Сведения об абонентах, которые пользовались междугородной
```

```
связью:");
            buf2 = Phone.search(l2, null, true, null);
            for(Phone ph : buf2)
                out.println(ph);
            out.println();
            out.println("Сведения об абонентах в алфавитном порядке:");
            buf2 = Phone.search(l2, null, null, true);
            for(Phone ph : buf2)
                out.println(ph);
            fileName = "./src/bdjava/lab5/text4.txt";
            out.println("Записываем в файл...");
            fOutSteam = new FileOutputStream(fileName);
            outputObjStream = new ObjectOutputStream(fOutSteam);
            outputObjStream.writeObject(l2.get(0));
            fOutSteam.close();
            outputObjStream.close();
            out.println("Читаем из файла...");
            fInSteam = new FileInputStream(fileName);
            inputObjStream = new ObjectInputStream(fInSteam);
            Phone q2 = (Phone) inputObjStream.readObject();
            fOutSteam.close();
            inputObjStream.close();
            out.println("Смотрим, что причли из файла...");
            out.println(q2.toString());
        }
        catch (Lab5Exceptions.NegativeNumberException ex) {
            out.println(ex.getMessage());
            out.println(ex.getErrorLocation());
        }
        catch (Lab5Exceptions.EmptyLineException ex) {
            out.println(ex.getMessage());
            out.println(ex.getErrorLocation());
        }
        catch (FileNotFoundException ex) {
            out.println("Файл не найден\nСистемная ошибка:");
            out.println(ex.getMessage());
        }
        catch (IOException ex) {
            out.println("Ошибка записи в поток или создания потока\nСистемная
ошибка:");
            out.println(ex.getMessage());
            out.println(ex.getStackTrace().toString());
            ex.printStackTrace();
        }
        catch (ClassNotFoundException ex) {
            out.println("Не найден класс\nСистемная ошибка:");
            out.println(ex.getMessage());
        }
    }
}
```

Код для решения задания 3:

```
package bdjava.lab5;
import org.jetbrains.annotations.NotNull;
```

```
import javax.swing.plaf.synth.SynthRadioButtonMenuItemUI;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.PublicKey;
import java.util.*;

import static java.lang.System.out;

public class Part3 {
    public static void main(String[] args) {
        if(args.length == 2) {
            try {
                String f1 = args[0];
                String f2 = args[1];
                PoemAnalyzer p = new PoemAnalyzer(f1, f2);
                p.estimateFrequencyNnAkhmatova();
            }
            catch (FileNotFoundException ex) {
                out.println(ex.getMessage());
            }
            catch (IOException ex) {
                out.println(ex.getMessage());
            }
        }
        else if (args.length == 0) {
            try {
                String f = "./src/bdjava/lab5/paths.txt";
                PoemAnalyzer p = new PoemAnalyzer(f);
                p.estimateFrequencyNnAkhmatova();
            }
            catch (FileNotFoundException ex) {
                out.println(ex.getMessage());
            }
            catch (IOException ex) {
                out.println(ex.getMessage());
            }
        }
        if(args.length == 2) {
            try {
                String f1 = args[0];
                String f2 = args[1];
                PoemAnalyzer p = new PoemAnalyzer(f1, f2);
                p.replacingLettersInZabolotsky();
            }
            catch (FileNotFoundException ex) {
                out.println(ex.getMessage());
            }
            catch (IOException ex) {
                out.println(ex.getMessage());
            }
```

```
        }
        else if (args.length == 0) {
            try {
                String f = "./src/bdjava/lab5/paths2.txt";
                PoemAnalyzer p = new PoemAnalyzer(f);
                p.replacingLettersInZabolotsky();
            }
            catch (FileNotFoundException ex) {
                out.println(ex.getMessage());
            }
            catch (IOException ex) {
                out.println(ex.getMessage());
            }
        }
    }
}

class PoemAnalyzer {
    private String source;
    private String target;
    static class WordRepetitions implements Comparable {
        private String word;
        private int count;
        public WordRepetitions (String word, int count) {
            this.word = word;
            this.count = count;
        }
        @Override
        public int compareTo(@NotNull Object o) {
            return this.count - ((WordRepetitions) o).count;
        }
        public String getWord() {
            return word;
        }
        public int getCount() {
            return count;
        }
        void inc () {
            this.count++;
        }
    }
    public static ArrayList <WordRepetitions> listForAkhmatova;
    static {
        listForAkhmatova = new ArrayList<WordRepetitions>();
        listForAkhmatova.add(new WordRepetitions("не знаю", 0));
        listForAkhmatova.add(new WordRepetitions("ничего", 0));
    }
    public PoemAnalyzer (String source, String target) throws
FileNotFoundException, IOException {
        FileInputStream fi = new FileInputStream(source);
        FileOutputStream fo = new FileOutputStream(target);
        fi.close();
        fo.close();
        this.source = source;
        this.target = target;
    }
```

```
    public PoemAnalyzer (String fileParams) throws FileNotFoundException,
NoSuchElementException, IOException {
        Scanner reader = new Scanner(new FileInputStream(fileParams));
        this.source = reader.nextLine();
        this.target = reader.nextLine();
        FileInputStream fi = new FileInputStream(this.source);
        FileOutputStream fo = new FileOutputStream(this.target);
        fi.close();
        fo.close();
        reader.close();
    }
    public void estimateFrequencyNnAkhmatova () throws NoSuchElementException,
IOException {
        Scanner reader = new Scanner(new FileInputStream(this.source));
        FileOutputStream fo = new FileOutputStream(this.target);
        PrintStream pfo = new PrintStream(fo);

        String line;
        while (reader.hasNextLine()) {
            line = reader.nextLine();
            for (WordRepetitions s : this.listForAkhmatova) {
                String sample = s.getWord().toLowerCase();
                int index = line.toLowerCase().indexOf(sample);
                while (index != -1) {
                    s.inc();
                    index = line.indexOf(sample, index + 1);
                }
            }
        }
        Collections.sort(this.listForAkhmatova);
        for (WordRepetitions s : this.listForAkhmatova)
            pfo.println("Слово: " + s.getWord() + "\nКоличество в тексте: " +
s.getCount() + "\n");
        pfo.close();
        reader.close();
        fo.close();
    }
    public void replacingLettersInZabolotsky () throws NoSuchElementException,
IOException {
        Scanner reader = new Scanner(new FileInputStream(this.source));
        FileOutputStream fo = new FileOutputStream(this.target);
        PrintStream pfo = new PrintStream(fo);

        String line;
        while (reader.hasNextLine()) {
            line = reader.nextLine();
            String[] words = line.split("\\s+");
            for (int i = 0; i < words.length; i++) {
                String firstLetter = words[i].substring(0, 1);
                String rest = words[i].substring(1);
                words[i] = firstLetter.toUpperCase() + rest;
            }
            String result = String.join(" ", words);
            pfo.println(result);
        }
```

```
        pfo.close();
        reader.close();
        fo.close();
    }
}
```

Код для решения задания 4:

```
package bdjava.lab5;

import org.jetbrains.annotations.NotNull;

import java.io.*;
import java.util.ArrayList;
import java.util.Collections;
import java.util.NoSuchElementException;
import java.util.Scanner;

import static java.lang.System.out;

public class Part4 {
    public static void main(String[] args) {
        try {
            //String f = "./src/bdjava/lab5/target551.txt";
            String f = "./src/bdjava/lab5/target552.txt";
            TextAnalyzer p = new TextAnalyzer(f);
            //p.deletingWords();
            p.removeDuplicateSpaces();
        }
        catch (FileNotFoundException ex) {
            out.println(ex.getMessage());
        }
        catch (IOException ex) {
            out.println(ex.getMessage());
        }
    }
}

class TextAnalyzer {
    private String source;
    private String target;
    public TextAnalyzer (String source, String target) throws
FileNotFoundException, IOException {
        FileInputStream fi = new FileInputStream(source);
        FileOutputStream fo = new FileOutputStream(target);
        fi.close();
        fo.close();
        this.source = source;
        this.target = target;
    }
    public TextAnalyzer (String fileParams) throws NoSuchElementException,
IOException {
        Scanner reader = new Scanner(new FileInputStream(fileParams));
        this.source = reader.nextLine();
        this.target = reader.nextLine();
```

```
        FileInputStream fi = new FileInputStream(this.source);
        FileOutputStream fo = new FileOutputStream(this.target);
        fi.close();
        fo.close();
        reader.close();
    }
    public void deletingWords () throws NoSuchElementException, IOException {
        Scanner reader = new Scanner(new FileInputStream(this.source));
        FileOutputStream fo = new FileOutputStream(this.target);
        PrintStream pfo = new PrintStream(fo);

        String line;
        while (reader.hasNextLine()) {
            line = reader.nextLine();
            String[] words = line.split("\\s+");
            ArrayList <String> newLine = new ArrayList<String>();
            int allowableAmount = 0;
            int currentAmount = 0;
            for (String word : words)
                if ((word.length() >= 3 && word.length() <= 5)) {
                    allowableAmount++;
                }
            allowableAmount &= 0b11111111111111111111111111111110;
            for (String word : words) {
                if (word.length() < 3 || word.length() > 5) {
                    newLine.add(word);
                }
                else if (currentAmount != allowableAmount)
                {
                    newLine.add(word);
                    currentAmount++;
                }
            }
            String result = String.join(" ", newLine);
            pfo.println(result);
        }

        pfo.close();
        reader.close();
        fo.close();
    }
    public void removeDuplicateSpaces () throws NoSuchElementException,
IOException {
        Scanner reader = new Scanner(new FileInputStream(this.source));
        FileOutputStream fo = new FileOutputStream(this.target);
        PrintStream pfo = new PrintStream(fo);

        String line;
        while (reader.hasNextLine()) {
            line = reader.nextLine();
            line = line.replaceAll("\\s+", " ");
            line = line.trim();
            pfo.println(line);
        }

        pfo.close();
```

```
        reader.close();
        fo.close();
    }
}
```

Вывод:

В ходе выполнения заданий лабораторной работы 3 были реализованы обработчики исключений для контроля состояния потоков ввода/вывода и корректности выполнения математических операций. Также была предусмотрена обработка исключений, связанных с нехваткой памяти, отсутствием требуемой записи в файле и недопустимым значением поля. В задании 2.7 была реализована функция поворота матрицы на 90, 180 или 270 градусов против часовой стрелки. В заданиях из варианта 2 были реализованы собственные обработчики исключений и исключения ввода/вывода. Были выполнены задания, в которых требовалось ввести последовательность строк из текстового потока и выполнить указанные действия. В задании 3.6 была реализована функция подсчета частоты повторяемости каждого слова из заданного списка в каждой строке стихотворения Анны Ахматовой и вывод этих слов в порядке возрастания частоты повторяемости. В задании 3.7 была реализована функция замены первой буквы каждого слова в стихотворении Николая Заболоцкого на прописную. В задании 4.6 была реализована функция удаления всех слов, содержащих от трех до пяти символов, из файла, при этом из каждой строки удаляется только максимальное четное количество таких слов. В задании 4.7 была реализована функция удаления всех "лишних" пробелов и табуляций из текста Java-программы, оставляя только необходимые для разделения операторов. Для вывода результатов во всех заданиях была создана новая директория и файл средствами класса File. В результате выполнения всех заданий были получены функциональные программы, способные обрабатывать различные типы данных и выполнять различные операции с ними.