**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа №4

По дисциплине «Информационные технологии и программирование»

Выполнил: Студент группы

БПИ 2301

Антонова Ирина

Москва

2024

**Цель:**

Изучение и применение исключений в Java

**Задание:**

Задание 1:

Необходимо написать программу, которая будет находить среднее арифметическое элементов массива. При этом программа должна обрабатывать ошибки, связанные с выходом за границы массива и неверными данными (например, если элемент массива не является числом).

Задание 2:

Необходимо написать программу, которая будет копировать содержимое одного файла в другой. При этом программа должна обрабатывать возможные ошибки, связанные с:

Вариант 1 Открытием и закрытием файлов

Вариант 2 Чтение и записью файлов

Задание 3:

Создайте Java-проект для работы с исключениями. Напишите свой собственный класс для обработки исключений. Создайте обработчик исключений, который логирует информацию о каждом выброшенном исключении в текстовый файл.

Вариант 4: Создайте класс CustomNumberFormatException, который будет использоваться для обработки исключения NumberFormatException. Реализуйте программу, которая пытается преобразовать строку в число (Integer.parseInt()), и, если строка не является числом, выбрасывайте исключение CustomNumberFormatException.

**Ход работы:**

Задание 1

Создадим папку LR4 и в ней файл ArrayAverage.java, в нем же класс с тем же названием. Создадим методы mein и getAverage.

public class ArrayAverage {

    public static void main(String[] args){

        Object[] arr1 = {1, 2, 3, 4};

        Object[] arr2 = {};

        Object[] arr3 = {"1", "2"};

        System.out.println(getAverage(arr1));

        System.out.println(getAverage(arr2));

        System.out.println(getAverage(arr3));

    }

В статическом методе getAverage реализуем работу с исключениями с помощью блоков try-catch. Внутри блока try находим среднее арифметическое массива, а в блоках catch обрабатываем исключения, когда массив пустой, состоит не из чисел или же ошибки связаны с выходом за границы массива (реализуется, если for (int i = 0; i < arr.length+1; i++))

    public static String getAverage(Object[] arr){

        int sum = 0;

        try{

            for (int i = 0; i < arr.length; i++){

                sum += (int) arr[i];

            }

            return("Average = " + sum/arr.length);

        }  catch (ArrayIndexOutOfBoundsException e) {

            return("Error: Array index out of bounds");

        } catch (ClassCastException e) {

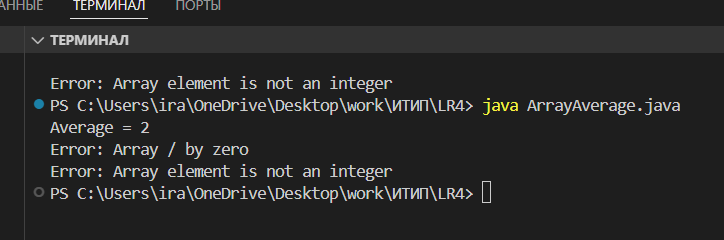
            return("Error: Array element is not an integer");

        } catch (ArithmeticException e) {

            return("Error: Array / by zero");

        }

    }

При вводе в консоль java ArrayAverage.java программа выведет  


Задание 2:

В той же папке LR4 создадим файл CopyFileJava.java и в нем класс с тем же названием, в котором будет реализовано задание 2. В классе создадим два метода mein и copyFile. Так же перед работой импортируем необходимые библиотеки для работы с файлами

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

В методе mein будет проверяться работа программы

    public static void main(String[] args) {

        System.out.println(copyFile("File.txt", "Copy.txt"));

        System.out.println(copyFile("someoneFile.txt", "Copy.txt"));

        System.out.println(copyFile("File.txt", "<.txt"));

    }

В методе copyFile реализуем обработки ошибок, связанные с открытием и закрытием файлов, чтением и записью

public static String copyFile(String inputFile, String outputFile) {

        FileReader reader = null;

        FileWriter writer = null;

        try {

            // Открытие файлов

            reader = new FileReader(inputFile);

            writer = new FileWriter(outputFile);

            int i;

            // Чтение файлов посимвольно и запись в новый файл

            while ((i = reader.read()) != -1) {

                writer.write(i);

            }

            return "Successfully written";

        } catch (FileNotFoundException e) {

            // файл не найден

            return "Error: File not found - " + e.getMessage();

        } catch (IOException e) {

            //ошибка ввода-вывода

            return "Error: I/O exception - " + e.getMessage();

        } finally { // закрытие файла в любом случае

            try {

                if (reader != null) {

                    reader.close();

                }

                if (writer != null) {

                    writer.close();

                }

            } catch (IOException e) {

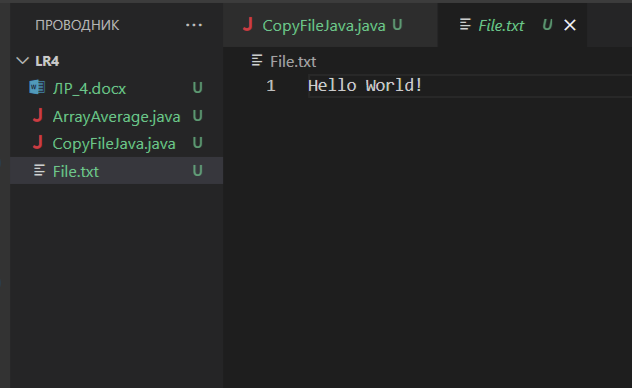
                return "Error: Failed to close file - " + e.getMessage();

            }

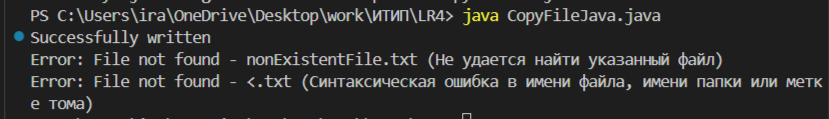
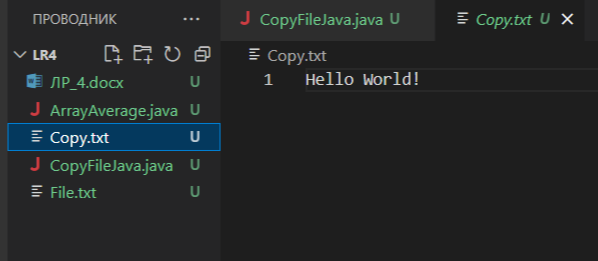
        }

    }

Кроме того для демонстрации правильной работы программы необходимо создать в папке LR4 файл File.txt, в котором напишем какой то текст, например:



После введения в терминаль команды java CopyFileJava.java программа выведет ответ. Также будет создан файл Copy.txt с скопированным из File.txt текстом

В данном случае невозможно продемонстрировать работу ошибки при закрытии файла, однако она может появиться при:

1. Проблемы с доступом к файлу:
   * Если файл был открыт, но затем доступ к нему был потерян (например, файл был удалён или перемещён).
2. Проблемы с файловой системой:
   * Если файловая система испытывает проблемы, такие как нехватка места на диске или повреждение файловой системы.
3. Проблемы с сетевыми файлами:
   * Если файл находится на сетевом диске и возникли проблемы с сетью (например, потеря соединения).
4. Проблемы с правами доступа:
   * Если у программы нет прав на закрытие файла (например, файл был открыт только для чтения, и программа пытается закрыть его после записи).

Задание 3:

В папке создадим файл Main.java. для реализации задания необходимо будет создать класс собственного исключения CustomNumberFormatException. Этот класс наследует NumberFormatException, что означает, что он также является типом исключения, связанного с ошибками преобразования строки в число. При создании исключения в него передаётся сообщение об ошибке, которое затем будет выведено при обработке исключения.

class CustomNumberFormatException extends NumberFormatException {

    public CustomNumberFormatException(String message) {

        super(message);

    }

}

Кроме того необходимо импортировать нужные пакеты

* FileWriter: класс используется для записи текста в файл.
* IOException: исключение, которое может возникнуть при работе с файлами (например, при отсутствии доступа к файлу).

import java.io.FileWriter;

import java.io.IOException;

В главном классе Main будут расписаны методы main, StringToInt и logException.

public class Main {

    public static void main(String[] args) {

        try {

            System.out.println(StringToInt("12"));

        } catch (CustomNumberFormatException e) {

            System.out.println(e);

        }

        try {

            System.out.println(StringToInt("1й2ц"));

        } catch (CustomNumberFormatException e) {

            System.out.println(e);

        }

    }

    public static int StringToInt(String str) throws CustomNumberFormatException {

        for (char c : str.toCharArray()) {

            if (!Character.isDigit(c)) {

                logException("Сannot convert string to number: " + str);

                throw new CustomNumberFormatException("Сannot convert string to number");

            }

        }

        return Integer.parseInt(str);

    }

    public static void logException(String message) {

        try (FileWriter writer = new FileWriter("errorFile.txt", true)) {

            writer.write(message + "\n");

        } catch (IOException e) {

            System.out.println(e);

        }

    }

}

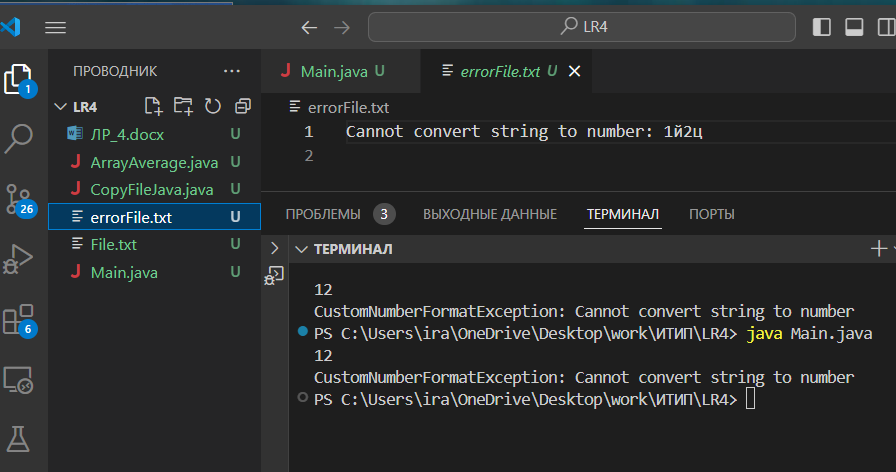
 Метод StringToInt проверяет, является ли строка числом:

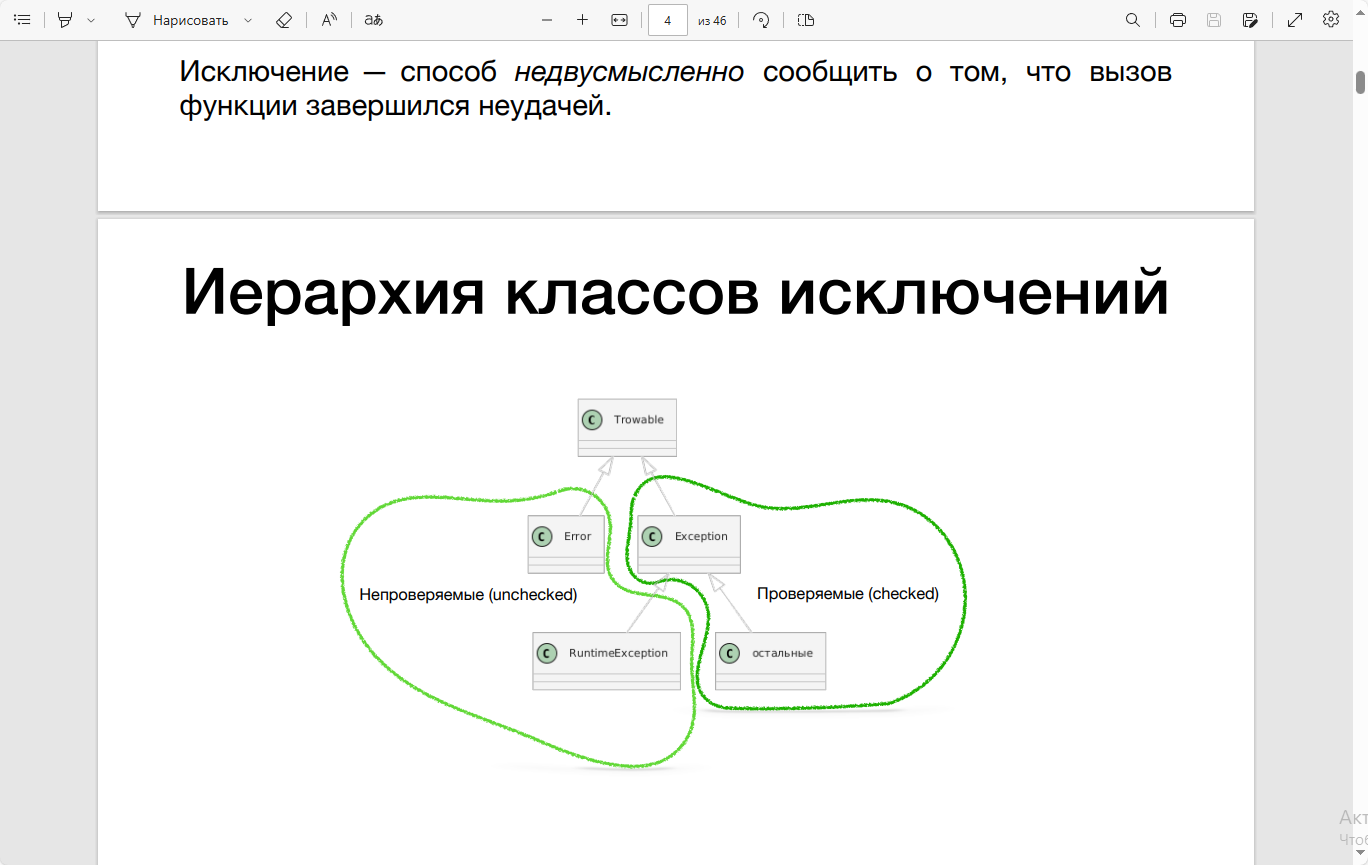
* + Цикл for проходит по всем символам строки str.
  + Метод Character.isDigit(c) проверяет, является ли символ c цифрой.
  + Если символ не является цифрой, метод logException записывает сообщение об ошибке в файл logFile.txt.
  + Затем выбрасывается исключение CustomNumberFormatException.
  + Если все символы строки str являются цифрами, метод Integer.parseInt(str) преобразует строку в целое число и возвращает результат.

**Метод**logException записывает сообщение об ошибке в файл:

* + Создает FileWriter с именем файла “logFile.txt” и флагом true, чтобы дописывать данные в файл.
  + Записывает message в файл и добавляет символ перехода на новую строку (“\n”).
* Обрабатывает исключение:
  + Блок try обертывает запись в файл.
  + Блок catch (IOException e) будет выполнен, если при записи в файл возникнет ошибка ввода-вывода. В этом случае на консоль будет выведено сообщение об ошибке.

При запуске программы командой java Main.java в терминале, программа выдает ответ, а также создает файл errorFile.txt с сообщением об ошибке





**Вывод:**

Были изучены исключения в Java, создано собственное исключение. Была проведена работа с файлами.

Ссылка на репозиторий с кодом https://github.com/k00kzaAntonovaIra/ITIP\_2024.git