МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена трудового Красного Знамени федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Математическая кибернетика и информационные технологии

Отчет по лабораторной работе № 4

Выполнил: студент группы БВТ2402

Косякова Олеся Дмитриевна

Москва, 2025

Цель работы: Освоение механизма обработки исключений в языке Java, изучение различных аспектов работы с исключениями - от базовой обработки стандартных исключений до создания собственных классов исключений и организации системы их логирования.

Задание:

1) Написать программу, которая будет находить среднее арифметическое элементов массива. При этом программа должна обрабатывать ошибки, связанные с выходом за границы массива и неверными данными
2) Написать программу, которая будет копировать содержимое одного файла в другой. При этом программа должна обрабатывать возможные ошибки, связанные: с открытием и закрытием файлов; чтением и записью файлов.
3) Создайте Java-проект для работы с исключениями. Создайте обработчик исключений, который логирует информацию о каждом выброшенном исключении в текстовый файл

Ход работы:

Задание 1.

Создаем массив объектов, с помощью цикла проходим по каждому и проверяем число ли это или нет, если да, добавляем к счетчику единицу и складываем приведённое число к сумме, если нет, то выбрасываем ошибку неправильного формата значения. Так же прописываем исключения при выходе за границу и кэтч для неверного формата

```
public class ArrayAverage {
    Run | Debug
    public static void main(String[] args) {
        Object[] arr = {1, 2, "hello", 4, 5};
        int sum = 0;
        int count = 0;
        try {
            for (int i = 0; i <= arr.length; i++) { // намеренно выходим за границы
                if (arr[i] instanceof Integer) {
                    sum += (int) arr[i];
                    count++;
                } else {
                    throw new NumberFormatException("Элемент "  + arr[i] + " не является числом");
                }
            }
```

```
            double ans = (double) sum / count;
        System.out.println(ans);
    } catch (ArrayIndexOutOfBoundsException e) {
        System.err.println(x: "Выход за границу массива");
    } catch (NumberFormatException e) {
        System.err.println("Неверные данные: " + e.getMessage());
    }
```

Задание 2.

Создаю два файла и сразу проверяю на открытие каждый.

```
public class File {
    Run | Debug
    public static void main(String[] args) {
        FileInputStream first = null;
        FileOutputStream second = null;

        try {
            try {
                first = new FileInputStream(name: "lab4/exm.txt");
                System.out.println(x: "Исходный файл открыт");

            } catch (FileNotFoundException e) {
                System.err.println("Ошибка открытия исходного файла: " + e.getMessage());
                return;
            }

            try {
                second = new FileOutputStream(name: "lab4/copy.txt");
                System.out.println(x: "Копируемый файл создан");
            } catch (FileNotFoundException e) {
                System.err.println("Ошибка открытия копируемого файла: " + e.getMessage());
                return;
            }
```

Прохожусь циклом по каждому байту в файле и читаю его, пока не дойду до конца. Сразу проверяю на ошибку чтения файла. Далее записываю во второй файл все то, что прочитали в первом и так же проверяю.

```
int s;
while (true) {
    try {
        s = first.read();
        if (s == -1) {
            break;
        }
    } catch (IOException e) {
        System.err.println("Ошибка чтения из файла: " + e.getMessage());
        break;
    }

    try {
        second.write(s);

    } catch (IOException e) {
        System.err.println("Ошибка записи в файл: " + e.getMessage());
        break;
    }
}
System.out.println(x: "Данные успешно скопированы");
```

Прописываю finally блок кода который будет выполняться всегда. Проверяю на закрытие файла.

```
} finally {
    try {
        if (first != null) {
            first.close();
            // first.read();
        }
    } catch (IOException e) {
        System.err.println("Ошибка при закрытии исходного файла: " + e.getMessage());
    }
    try {
        if (second != null) {
            second.close();
        }
    } catch (IOException e) {
        System.err.println("Ошибка при закрытии копируемого файла: " + e.getMessage());
    }
    System.out.println(x: "Файлы закрыты");

}
```

Задание 3.

Создаю обработчик исключений, который наследуется от класса Exception. В конструкторе принимает сообщение об ошибке и передает родительскому классу.

```
public class CustomDivisionException extends Exception {
    public CustomDivisionException(String message) {
        super(message);
    }
}
```

Далее создаем класс для записи ошибок из консоли в текстовый файл. Создаем внутри объекта PrintWriter объект FileWriter с адресом файла и значением true, который указывает на запись данных в конец фала, а не их перезапись. Так как PrintWriter преподносит удобные инструменты для форматирования текста. Если не удалось записать, выводим ошибку. Далее закрываем наш файл.

```java
public class Log {
    public static void logEx(Exception e) {
        PrintWriter er = null;
        try {
            er = new PrintWriter(new FileWriter(fileName: "lab4/log.txt", append: true));
            er.println("Ошибка: " + e.getMessage());
        } catch (IOException err) {
        System.err.println("Не удалось записать в лог: " + err.getMessage());
    } finally {
        if (er != null) {
            er.close();
        }
    }
}
}
```

Выбрасываем ошибку если делитель равен нулю и передаем сообщение классу Log.

```java
public class DivisionTest {
    Run | Debug
    public static void main(String[] args) {
        int a = 2;
        int b = 0;
        try {
            if (b == 0) {
                throw new CustomDivisionException(message: "Ошибка. Деление на ноль");
            }
            double ans = (double) a/b;
            System.out.println(ans);
        } catch (CustomDivisionException e) {
            System.err.println(e.getMessage());
            Log.logEx(e);
        }
    }

}
```

Вывод: В результате выполнения лабораторной работы был получен комплексный опыт работы с исключениями в Java. На примере обработки ошибок при работе с массивами удалось освоить базовые принципы обработки исключений, связанных с выходом за границы массива и некорректными данными.

Ссылка на репозиторий: https://github.com/swydiz/information_technology