|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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**

**Отчет по лабораторной работе № 5**

***по дисциплине «Языки программирования для работы с большими данными»***

Студент ИУ6-21М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е. А. Щербакова**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ П. В. Степанов**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

**Задание 1:**

Выполнить задания на основе варианта 1 лабораторной работы 4, 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.

**Код программы**

**Файл Exhibition.java**

package Exhibition;

import java.util.ArrayList;

import java.util.Date;

public class Exhibition {

class Author {

public String name;

public String surname;

public Author(String name, String surname) {

try {

this.name = name;

this.surname = surname;

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

}

class Painting {

public int width;

public int height;

public Author author;

public Painting(int width, int height, Author author) {

try {

this.width = width;

this.height = height;

this.author = author;

} catch(Exception e) {

System.out.println(e.getMessage());

}

}

}

public Date date;

public ArrayList<Painting> paintings = new ArrayList<Painting>();

public Exhibition(Date date) {

try {

this.date = date;

} catch(Exception e) {

System.out.println(e.getMessage());

}

}

public void addPainting(Painting painting) {

try {

this.paintings.add(painting);

} catch (Exception e) {

System.out.println(e.getMessage());

}

}

}

**Файл Main.java**

package Exhibition;

import java.util.Date;

import java.util.InputMismatchException;

import java.util.Scanner;

import Exhibition.Exhibition.Author;

import Exhibition.Exhibition.Painting;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int width = 0;

int height = 0;

try {

width = scanner.nextInt();

} catch(InputMismatchException e) {

System.out.println("Width must be int");

}

try {

height = scanner.nextInt();

} catch(InputMismatchException e) {

System.out.println("Height must be int");

}

try {

Exhibition exhibition = new Exhibition(new Date());

Author author = exhibition.new Author("Иван", "Иванов");

Painting painting = exhibition.new Painting(width, height, author);

exhibition.addPainting(painting);

} catch(InputMismatchException e) {

System.out.println(e.getMessage());

} catch (OutOfMemoryError e) {

System.out.println(e.getMessage());

}

}

}

**Задание 2**

Выполнить задания из варианта 2 лабораторной работы 4, реализуя собственные обработчики исключений и исключения ввода/вывода.

**Код программы:**

**Файл Main.java**

package Worker;

public class Main {

public static void main(String[] args) {

try {

Leed leed = new Leed("Иван", "Иванов");

Engineer worker = new Engineer("Петр", "Петров");

leed.work();

worker.work();

leed.letGo(worker);

} catch(OutOfMemoryError e) {

System.***out***.println(e.getMessage());

} catch (NullPointerException e){

System.***out***.println(e.getMessage());

}

}

}

**Файл Leed.java**

package Worker;

public class Leed extends Engineer {

public Leed(String name, String company) {

super(name, company);

}

public void letGo(Engineer worker) {

worker.goToWeekend();

}

}

**Файл Engineer.java**

}

package Worker;

public class Engineer implements Worker {

public String name;

public String company;

public boolean isWorkNow = true;

public Engineer(String name, String company) {

this.name = name;

this.company = company;

}

public void work() {

this.isWorkNow = true;

}

public void goToWeekend() {

this.isWorkNow = false;

}

}

**Файл Worker.java**

package Worker;

public interface Worker {

void work();

void goToWeekend();

}

**Вывод программы**

Work!

Work!

Go!

Weekend!

**Задание 3**

**Вариант 1**

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 каждой строке найти и удалить заданную подстроку.

**Код программы**

**Файл Main.java**

package String;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Paths;

import java.util.List;

import java.util.stream.Collectors;

import java.util.stream.Stream;

public class Main {

public static void main(String[] args) throws IOException {

Stream<String> stream = Files.lines(Paths.get("/Users/liza\_shch/eclipse-workspace/LR5/src/String/test.txt"));

List<String> result = stream

.map(line -> line.replaceAll("final", ""))

.collect(Collectors.toList());

BufferedWriter writer = new BufferedWriter(new FileWriter(new File("/Users/liza\_shch/eclipse-workspace/LR5/src/String/result.txt")));

for (String line: result) {

writer.append(line);

writer.newLine();

}

writer.close();

}

}

**Файл test.txt**Abc final

final sdefrr rfr

Ejderd final final

Rtfr

**Файл result.txt**Abc

sdefrr rfr

Ejderd

Rtfr

**Задание 4**

**Вариант 1**

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

Прочитать текст Java-программы и все слова public в объявлении атрибутов и методов класса заменить на слово private.

**Код программы**

**Файл Main.java**

package File;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.OutputStreamWriter;

import java.util.Scanner;

public class Main {

public static void main(String[] args) throws IOException {

File file = new File("/Users/liza\_shch/eclipse-workspace/LR5/src/File/test.txt");

File result = new File("/Users/liza\_shch/eclipse-workspace/LR5/src/File/output/test.txt");

result.getParentFile().mkdir();

result.createNewFile();

BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(result)));

Scanner sc = new Scanner(file);

while (sc.hasNextLine()) {

String line = sc.nextLine();

writer.append(line.replace("public", "private"));

writer.newLine();

}

sc.close();

writer.close();

}

}

**Файл test.txt**

class Main {

public run() {

}

private work() {

}

}

**Файл output/test.txt**

class Main {

private run() {

}

private work() {

}

}

**Вывод:** Программы работают корректно.