|  |  |
| --- | --- |
| 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**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений.**

**Отчет**

**По лабораторной работе №5**

**Название:** Исключения, файлы

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-23М |  |  | В.Д. Рожнов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Задание 1:** выполнить задания на основе лабораторной работы 1, 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.

Решение задания представлено в листингах 1-2.

Листинг 1 – Программный код модуля 1-го подзадания для задания 1

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

int n = 5; // Количество строк для вывода

String outputFilePath = "output.txt"; // Путь к файлу вывода

try (BufferedWriter writer = new BufferedWriter(new FileWriter(outputFilePath))) {

for (int i = 0; i < n; i++) {

String line = "Строка " + (i + 1);

// Вывод строки с переходом на новую строку

writer.write(line);

writer.newLine();

// Вывод строки без перехода на новую строку

writer.write(line);

}

System.out.println("Вывод успешно завершен.");

} catch (IOException e) {

System.err.println("Ошибка при работе с файлом: " + e.getMessage());

} catch (Exception e) {

System.err.println("Возникла ошибка: " + e.getMessage());

}

}

}

Листинг 2 – Программный код модуля 2-го подзадания для задания 1

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

public static void main(String[] args) {

String passwordPattern = "password"; // Строка-образец пароля

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

try {

System.out.print("Введите пароль: ");

String password = reader.readLine();

if (password.equals(passwordPattern)) {

System.out.println("Пароль верный.");

} else {

System.out.println("Пароль неверный.");

}

} catch (IOException e) {

System.out.println("Ошибка ввода-вывода: " + e.getMessage());

} catch (Exception e) {

System.out.println("Произошла ошибка: " + e.getMessage());

} finally {

try {

reader.close();

} catch (IOException e) {

System.out.println("Ошибка при закрытии потока ввода: " + e.getMessage());

}

}

}

}

**Задание 2:** выполнить задания из лабораторной работы 3, реализуя собственные обработчики исключений и исключения ввода/вывода.

Решение задания представлено в листингах 3-5.

Листинг 3 – Программный код модуля Main.java для задания 2

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

public static void main(String[] args) {

Book[] books = new Book[]{

new Book(1, "The Lord of the Rings", new String[]{"J.R.R. Tolkien"}, "HarperCollins", 1954, 1178, 25.99, "Hardcover"),

new Book(2, "The Hobbit", new String[]{"J.R.R. Tolkien"}, "Houghton Mifflin Harcourt", 1937, 310, 12.99, "Paperback"),

new Book(3, "Harry Potter and the Philosopher's Stone", new String[]{"J.K. Rowling"}, "Bloomsbury", 1997, 223, 9.99, "Paperback"),

new Book(4, "Harry Potter and the Chamber of Secrets", new String[]{"J.K. Rowling"}, "Bloomsbury", 1998, 251, 10.99, "Paperback"),

new Book(5, "Harry Potter and the Prisoner of Azkaban", new String[]{"J.K. Rowling"}, "Bloomsbury", 1999, 317, 11.99, "Paperback"),

new Book(6, "The Da Vinci Code", new String[]{"Dan Brown"}, "Doubleday", 2003, 454, 12.99, "Paperback"),

new Book(7, "Angels & Demons", new String[]{"Dan Brown"}, "Pocket Books", 2000, 736, 11.99, "Paperback"),

new Book(8, "Digital Fortress", new String[]{"Dan Brown"}, "St. Martin's Press", 1998, 384, 10.99, "Paperback"),

new Book(9, "The Lost Symbol", new String[]{"Dan Brown"}, "Doubleday", 2009, 672, 13.99, "Hardcover"),

new Book(10, "Origin", new String[]{"Dan Brown"}, "Doubleday", 2017, 480, 14.99, "Hardcover")

};

BookArray bookArray = new BookArray(books);

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

try {

// список книг заданного автора

System.out.println("Books by J.K. Rowling:");

Book[] jkRowlingBooks = bookArray.getByAuthor("J.K. Rowling");

for (Book book : jkRowlingBooks) {

System.out.println(book);

}

// список книг, выпущенных заданным издательством

System.out.println("Books published by Bloomsbury:");

Book[] bloomsburyBooks = bookArray.getByPublisher("Bloomsbury");

for (Book book : bloomsburyBooks) {

System.out.println(book);

}

// список книг, выпущенных после заданного года

System.out.println("Books published after 2000:");

Book[] booksAfter2000 = bookArray.getAfterYear(2000);

for (Book book : booksAfter2000) {

System.out.println(book);

}

} catch (NumberFormatException e) {

System.out.println("Ошибка преобразования числа: " + e.getMessage());

} finally {

try {

reader.close();

} catch (IOException e) {

System.out.println("Ошибка при закрытии потока ввода: " + e.getMessage());

}

}

}

}

Листинг 4 – Программный код модуля Book.java для задания 2

import java.util.Arrays;

public class Book {

private int id;

private String title;

private String[] authors;

private String publisher;

private int year;

private int pageCount;

private double price;

private String binding;

public Book(int id, String title, String[] authors, String publisher, int year, int pageCount, double price, String binding) {

this.id = id;

this.title = title;

this.authors = authors;

this.publisher = publisher;

this.year = year;

this.pageCount = pageCount;

this.price = price;

this.binding = binding;

}

public void setId(int id) {

this.id = id;

}

public int getId() {

return id;

}

public void setTitle(String title) {

this.title = title;

}

public String getTitle() {

return title;

}

public void setAuthors(String[] authors) {

this.authors = authors;

}

public String[] getAuthors() {

return authors;

}

public void setPublisher(String publisher) {

this.publisher = publisher;

}

public String getPublisher() {

return publisher;

}

public void setYear(int year) {

this.year = year;

}

public int getYear() {

return year;

}

public void setPageCount(int pageCount) {

this.pageCount = pageCount;

}

public int getPageCount() {

return pageCount;

}

public void setPrice(double price) {

this.price = price;

}

public double getPrice() {

return price;

}

public void setBinding(String binding) {

this.binding = binding;

}

public String getBinding() {

return binding;

}

public String toString() {

return "Book{" +

"id=" + id +

", title='" + title + '\'' +

", authors=" + Arrays.toString(authors) +

", publisher='" + publisher + '\'' +

", year=" + year +

", pageCount=" + pageCount +

", price=" + price +

", binding='" + binding + '\'' +

'}';

}

}

Листинг 4 – Программный код модуля BookArray.java для задания 2

import java.util.ArrayList;

public class BookArray {

private Book[] books;

public BookArray(Book[] books) {

this.books = books;

}

public Book[] getBooks() {

return books;

}

public Book[] getByAuthor(String author) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

for (String bookAuthor : book.getAuthors()) {

if (bookAuthor.equals(author)) {

result.add(book);

break;

}

}

}

return result.toArray(new Book[result.size()]);

}

public Book[] getByPublisher(String publisher) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

if (book.getPublisher().equals(publisher)) {

result.add(book);

}

}

return result.toArray(new Book[result.size()]);

}

public Book[] getAfterYear(int year) {

ArrayList<Book> result = new ArrayList<Book>();

for (Book book : books) {

if (book.getYear() > year) {

result.add(book);

}

}

return result.toArray(new Book[result.size()]);

}

}

**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.

Найти в строке наибольшее число цифр, идущих подряд.

Решение задания представлено в листинге 5.

Листинг 5 – Программный код модуля Main.java для задания 3

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class Main {

public static void main(String[] args) {

// Проверяем, переданы ли аргументы командной строки

if (args.length < 2) {

System.out.println("Usage: java Main <input\_file> <output\_file>");

return;

}

String inputFile = args[0];

String outputFile = args[1];

try (BufferedReader reader = new BufferedReader(new FileReader(inputFile))) {

String line;

int maxConsecutiveDigits = 0;

// Читаем строки из входного файла

while ((line = reader.readLine()) != null) {

int consecutiveDigits = countConsecutiveDigits(line);

maxConsecutiveDigits = Math.max(maxConsecutiveDigits, consecutiveDigits);

}

// Записываем результат в выходной файл

FileWriter writer = new FileWriter(outputFile);

writer.write(String.valueOf(maxConsecutiveDigits));

writer.close();

System.out.println("Max consecutive digits: " + maxConsecutiveDigits);

} catch (IOException e) {

System.out.println("Error reading input.txt file: " + e.getMessage());

}

}

private static int countConsecutiveDigits(String line) {

int maxConsecutiveDigits = 0;

int currentConsecutiveDigits = 0;

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

if (Character.isDigit(line.charAt(i))) {

currentConsecutiveDigits++;

maxConsecutiveDigits = Math.max(maxConsecutiveDigits, currentConsecutiveDigits);

} else {

currentConsecutiveDigits = 0;

}

}

return maxConsecutiveDigits;

}

}

**Задание 4:** 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 стихотворения Анны Ахматовой подсчитать частоту повторяемости каждого слова из заданного списка и вывести эти слова в порядке возрастания частоты повторяемости.

Решение задания представлено в листинге 6.

Листинг 6 – Программный код модуля Main.java для задания 4

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.\*;

public class Main {

public static void main(String[] args) {

// Проверяем, переданы ли аргументы командной строки

if (args.length < 3) {

System.out.println("Usage: java Main <input\_file> <output\_file> <word\_list\_file>");

return;

}

String inputFile = args[0];

String outputFile = args[1];

String wordListFile = args[2];

try {

List<String> wordList = readWordList(wordListFile);

Map<String, Integer> wordFrequencyMap = new HashMap<>();

// Читаем строки из входного файла

try (BufferedReader reader = new BufferedReader(new FileReader(inputFile))) {

String line;

while ((line = reader.readLine()) != null) {

countWordFrequency(line, wordList, wordFrequencyMap);

}

}

// Сортируем слова по частоте повторяемости

List<Map.Entry<String, Integer>> sortedWords = new ArrayList<>(wordFrequencyMap.entrySet());

sortedWords.sort(Map.Entry.comparingByValue());

// Записываем результат в выходной файл

FileWriter writer = new FileWriter(outputFile);

for (Map.Entry<String, Integer> entry : sortedWords) {

writer.write(entry.getKey() + ": " + entry.getValue() + "\n");

}

writer.close();

System.out.println("Word frequencies written to " + outputFile);

} catch (IOException e) {

System.out.println("Error: " + e.getMessage());

}

}

private static List<String> readWordList(String wordListFile) throws IOException {

List<String> wordList = new ArrayList<>();

try (BufferedReader reader = new BufferedReader(new FileReader(wordListFile))) {

String word;

while ((word = reader.readLine()) != null) {

wordList.add(word.trim());

}

}

return wordList;

}

private static void countWordFrequency(String line, List<String> wordList, Map<String, Integer> wordFrequencyMap) {

String[] words = line.split("\\s+");

for (String word : words) {

if (wordList.contains(word)) {

wordFrequencyMap.put(word, wordFrequencyMap.getOrDefault(word, 0) + 1);

}

}

}

}

**Вывод:** в ходе выполнения данной лабораторной работы была освоена работа с исключениями и изучены приемы работы с файлами на языке Java.