**File and exception handling**

1.  Write a Java program to get files with  specific extension from a specified folder.

**CODE:**

import java.io.File;

import java.util.ArrayList;

import java.util.List;

public class File{

public static void main(String[] args) {

String folderPath = "C:\Users\DELL\Downloads\";

String fileExtension = "txt";

List<File> filteredFiles = getFilesWithExtension(folderPath, fileExtension);

System.out.println("Files with extension '" + fileExtension + "':");

for (File file : filteredFiles) {

System.out.println(file.getAbsolutePath());

}

}

private static List<File> getFilesWithExtension(String folderPath, String fileExtension) {

List<File> result = new ArrayList<>();

File folder = new File(folderPath);

if (folder.exists() && folder.isDirectory()) {

File[] files = folder.listFiles();

if (files != null) {

for (File file : files) {

if (file.isFile() && file.getName().toLowerCase().endsWith("." + fileExtension.toLowerCase())) {

result.add(file);

}

}

}

} else {

System.err.println("Specified folder does not exist or is not a directory.");

}

return result;

}

}

2.  Write a Java program that reads a list of numbers from a file and throws an exception if any of the numbers are positive.

**Output:**

Content of test.txt: -1 -2 -3 4

Error: Positive number found: 4

**CODE:**

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class PositiveNumberChecker {

public static void main(String[] args) {

String filePath = "test.txt";

try {

checkForPositiveNumbers(filePath);

} catch (IOException e) {

System.err.println("Error reading the file: " + e.getMessage());

} catch (PositiveNumberException e) {

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

}

}

private static void checkForPositiveNumbers(String filePath) throws IOException, PositiveNumberException {

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

String line = reader.readLine();

System.out.println("Content of " + filePath + ": " + line);

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

for (String numberStr : numbers) {

int number = Integer.parseInt(numberStr);

if (number > 0) {

throw new PositiveNumberException("Positive number found: " + number);

}

}

System.out.println("No positive numbers found.");

}

}

static class PositiveNumberException extends Exception {

public PositiveNumberException(String message) {

super(message);

}

}

}

3.  You are given a directory path that contains a number of text files. Each text file contains words separated by spaces.

Your task is to write a Java program that finds the most common word across all the files. Consider a word as a sequence of characters separated by spaces. Ignore case sensitivity, meaning "hello" and "Hello" should be considered the same word.

Write a Java program that takes the directory path as input and outputs the most common word along with its frequency. If there are multiple words with the same highest frequency, output all of them.

**Input:**

Enter directory name : TextFolder

**Output:**

Word: world, Frequency: 3  
Word: java, Frequency: 2  
Word: hello, Frequency: 2  
Word: is, Frequency: 1  
Word: a, Frequency: 1  
Word: programming, Frequency: 1  
Word: language, Frequency: 1

**CODE:**

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

public class Frequency {

public static void main(String[] args) {

System.out.print("Enter directory name: ");

String directoryPath = System.console().readLine();

try {

Map<String, Integer> wordFrequencyMap = findMostCommonWord(directoryPath);

System.out.println("\nMost common word(s) and their frequency:");

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

System.out.println("Word: " + entry.getKey() + ", Frequency: " + entry.getValue());

}

} catch (IOException e) {

System.err.println("Error reading files: " + e.getMessage());

}

}

private static Map<String, Integer> findMostCommonWord(String directoryPath) throws IOException {

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

File directory = new File(directoryPath);

File[] files = directory.listFiles();

if (files != null) {

for (File file : files) {

if (file.isFile()) {

updateWordFrequencyMap(file, wordFrequencyMap);

}

}

}

return wordFrequencyMap;

}

private static void updateWordFrequencyMap(File file, Map<String, Integer> wordFrequencyMap) throws IOException {

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

String line;

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

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

for (String word : words) {

word = word.toLowerCase(); // Ignore case sensitivity

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

}

}

}

}

}