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

import java.io.File;

import java.io.FilenameFilter;

public class FindFilesWithSpecificExtension {

public static void main(String[] args) {

// Replace with the actual folder path

String folderPath = "/path/to/your/folder";

// Replace with the desired extension (without the dot)

String extension = "txt";

File folder = new File(folderPath);

File[] files = folder.listFiles(new FilenameFilter() {

@Override

public boolean accept(File dir, String name) {

return name.toLowerCase().endsWith("." + extension);

}

});

if (files != null) {

for (File file : files) {

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

}

} else {

System.out.println("No files found with the specified extension.");

}

}

}

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

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class CheckNumbersFromFile {

public static void main(String[] args) {

String filename = "test.txt";

try {

checkNumbers(filename);

} catch (FileNotFoundException e) {

System.out.println("Error: File not found.");

} catch (PositiveNumberException e) {

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

}

}

public static void checkNumbers(String filename) throws FileNotFoundException, PositiveNumberException {

File file = new File(filename);

Scanner scanner = new Scanner(file);

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

while (scanner.hasNextLine()) {

int number = Integer.parseInt(scanner.nextLine());

if (number > 0) {

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

}

System.out.println(number);

}

System.out.println("All numbers are non-positive.");

}

}

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

import java.io.File;

import java.io.FileNotFoundException;

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

public class FindMostCommonWord {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

String directoryPath = scanner.nextLine();

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

try {

processFiles(directoryPath, wordCounts);

findAndPrintMostCommonWords(wordCounts);

} catch (FileNotFoundException e) {

System.out.println("Error: Directory not found.");

}

}

private static void processFiles(String directoryPath, Map<String, Integer> wordCounts) throws FileNotFoundException {

File directory = new File(directoryPath);

File[] files = directory.listFiles();

if (files != null) {

for (File file : files) {

if (file.isFile()) {

countWordsInFile(file, wordCounts);

}

}

}

}

private static void countWordsInFile(File file, Map<String, Integer> wordCounts) throws FileNotFoundException {

Scanner fileScanner = new Scanner(file);

while (fileScanner.hasNext()) {

String word = fileScanner.next().toLowerCase(); // Case-insensitive counting

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

}

}

private static void findAndPrintMostCommonWords(Map<String, Integer> wordCounts) {

int maxFrequency = 0;

for (int frequency : wordCounts.values()) {

maxFrequency = Math.max(maxFrequency, frequency);

}

System.out.println("Most common words:");

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

if (entry.getValue() == maxFrequency) {

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

}

}

}

}