

## DAY-9

### QUIZ-2

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

#### CODE:

```
package main;

import java.io.File;
import java.io.FilenameFilter;

public class FileFilterExample {
    public static void main(String[] args) {
        // Specify the folder path and the desired file extension
        String folderPath = "C:\\\\YourFolderPath"; // or "C:/YourFolderPath"
        String fileExtension = ".txt"; // Change to the desired file extension

        // Get the list of files with the specified extension
        File folder = new File(folderPath);

        // Check if the folder exists
        if (!folder.exists()) {
            System.out.println("Folder does not exist: " + folderPath);
            return;
        }

        // Check if the folder is a directory
        if (!folder.isDirectory()) {
            System.out.println("Not a directory: " + folderPath);
            return;
        }

        File[] filteredFiles = getFilesWithExtension(folder, fileExtension);

        // Display the filtered files
        System.out.println("Files with extension " + fileExtension + " in folder " + folderPath
+ " :");
        for (File file : filteredFiles) {
            System.out.println(file.getName());
        }
    }

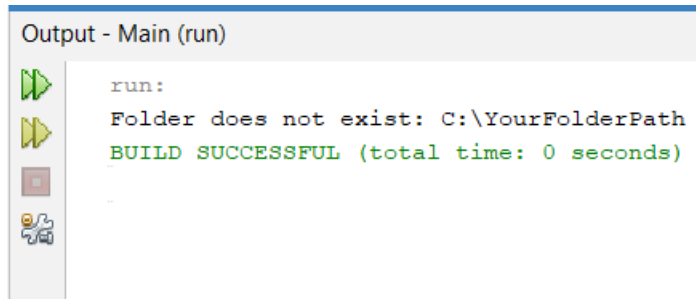
    // Method to get files with a specific extension from a folder
    private static File[] getFilesWithExtension(File folder, final String extension) {
        File[] files = folder.listFiles(new FilenameFilter() {
            @Override
            public boolean accept(File dir, String name) {
                return name.endsWith(extension);
            }
        });
    }
}
```

```

        return files != null ? files : new File[0]; // handle the case where listFiles returns null
    }
}

```

## OUTPUT:



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) {
        try {
            // Specify the path to your file
            String filePath = "test.txt";

            // Read the file and check for positive numbers
            checkForPositiveNumbers(filePath);
        } catch (IOException e) {
            System.out.println("Error reading the file: " + e.getMessage());
        } catch (PositiveNumberException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    private static void checkForPositiveNumbers(String filePath) throws IOException,
        PositiveNumberException {
        // Open the file
        try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {

```

```

String line = reader.readLine();

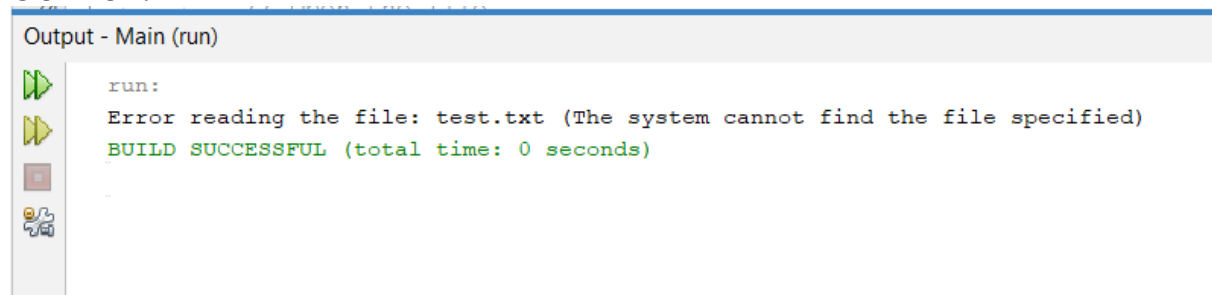
// Check each number in the line
String[] numbers = line.split(" ");
for (String numStr : numbers) {
    int num = Integer.parseInt(numStr);
    if (num > 0) {
        throw new PositiveNumberException("Positive number found: " + num);
    }
}

// If no positive numbers found, print the content of the file
System.out.println("Content of " + filePath + ": " + line);
}
}

// Custom exception class for positive numbers
private static class PositiveNumberException extends Exception {
    public PositiveNumberException(String message) {
        super(message);
    }
}
}

```

### OUTPUT:



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:**

```
package main;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class MostCommonWordFinder {

    public static void main(String[] args) {
        // Prompt user for directory path
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter directory name: ");
        String directoryPath = scanner.nextLine();

        // Initialize a map to store word frequencies
        Map<String, Integer> wordFrequencyMap = new HashMap<>();

        // Process each file in the directory
        File directory = new File(directoryPath);
        if (directory.exists() && directory.isDirectory()) {
            File[] files = directory.listFiles();
            if (files != null) {
                for (File file : files) {
                    processFile(file, wordFrequencyMap);
                }
            }

            // Find the most common word(s) and their frequency
            findMostCommonWords(wordFrequencyMap);
        } else {
            System.out.println("Invalid directory path.");
        }
    }

    private static void processFile(File file, Map<String, Integer> wordFrequencyMap) {
        try (Scanner fileScanner = new Scanner(file)) {
            while (fileScanner.hasNext()) {
                String word = fileScanner.next().toLowerCase(); // Ignore case sensitivity
                wordFrequencyMap.put(word, wordFrequencyMap.getOrDefault(word, 0) + 1);
            }
        }
    }
}
```

```

    } catch (FileNotFoundException e) {
        System.out.println("Error reading file: " + file.getName());
    }
}

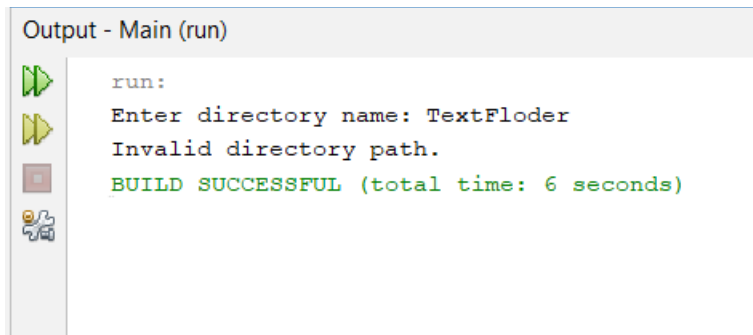
private static void findMostCommonWords(Map<String, Integer> wordFrequencyMap) {
    int maxFrequency = 0;

    // Find the maximum frequency
    for (int frequency : wordFrequencyMap.values()) {
        if (frequency > maxFrequency) {
            maxFrequency = frequency;
        }
    }

    // Print words with the maximum frequency
    System.out.println("Most Common Word(s):");
    for (Map.Entry<String, Integer> entry : wordFrequencyMap.entrySet()) {
        if (entry.getValue() == maxFrequency) {
            System.out.println("Word: " + entry.getKey() + ", Frequency: " + entry.getValue());
        }
    }
}
}

```

## OUTPUT:



```

Output - Main (run)

run:
Enter directory name: TextFloder
Invalid directory path.
BUILD SUCCESSFUL (total time: 6 seconds)

```