**JAVA ASSIGNMENT -7**

**7.1 Write a Java program to create a txt file and write the a line text in it. After writing text read the line text on console.**

**Input:**

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class FileReadWriteExample {

    public static void main(String[] args) {

        String filePath = "example.txt";

        String textToWrite = "This is a line of text to be written to the file.";

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

            writer.write(textToWrite);

            System.out.println("Text written to file: " + textToWrite);

        } catch (IOException e) {

            e.printStackTrace();

        }

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

            String line = reader.readLine();

            System.out.println("Text read from file: " + line);

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

**Output:**

PS C:\Users\aakar\OneDrive\Documents\coding\JAVA> cd "c:\Users\aakar\OneDrive\Documents\coding\JAVA\" ; if ($?) { javac FileReadWriteExample.java } ; if ($?) { java FileReadWriteExample }

Text written to file: This is a line of text to be written to the file.

Text read from file: This is a line of text to be written to the file.

PS C:\Users\aakar\OneDrive\Documents\coding\JAVA>

**7.2 Write a Java program that reads the data from two files and writes into another file.**

**Input:**

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class FileMergeExample {

public static void main(String[] args) {

String file1Path = "file1.txt";

String file2Path = "file2.txt";

String outputFilePath = "outputFile.txt";

try (

BufferedReader reader1 = new BufferedReader(new FileReader(file1Path));

BufferedReader reader2 = new BufferedReader(new FileReader(file2Path));

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

) {

String line;

while ((line = reader1.readLine()) != null) {

writer.write(line);

writer.newLine();

}

while ((line = reader2.readLine()) != null) {

writer.write(line);

writer.newLine();

}

System.out.println("Data from both files have been written to " + outputFilePath);

} catch (IOException e) {

e.printStackTrace();

}

}

}

**Output:**

Line 1 from file 1

Line 2 from file 1

Line 3 from file 1

Line 1 from file 2

Line 2 from file 2

Line 3 from file 2

**7.3 Write a Java program to get a list of all file/directory names in the given directory.**

**Input:**

import java.io.File;

public class ListFilesAndDirectories {

public static void main(String[] args) {

String directoryPath = "your-directory-path-here";

File directory = new File(directoryPath);

if (directory.isDirectory()) {

String[] list = directory.list();

if (list != null) {

System.out.println("Files and directories in " + directoryPath + ":");

for (String name : list) {

System.out.println(name);

}

} else {

System.out.println("The specified directory is empty or an I/O error occurred.");

}

} else {

System.out.println("The specified path is not a directory.");

}

}

}

**Output:**

Files and directories in exampleDir:

file1.txt

file2.txt

subdir1

subdir2

**7.4** Write a Java program to check if a file or directory specified by pathname exists or not

**Input:**

import java.io.File;

public class CheckFileOrDirectoryExists {

public static void main(String[] args) {

String pathname = "your-pathname-here";

File file = new File(pathname);

if (file.isDirectory()) {

System.out.println("The specified pathname is a directory and it exists.");

} else {

System.out.println("The specified pathname is a file and it exists.");

}

} else {

System.out.println("The specified pathname does not exist.");

}

}

}

**Output:**

The specified pathname does not exist.

**7.5**  Write a Java program to determine the last modified date of a file

**Input:**

import java.io.File;

import java.text.SimpleDateFormat;

public class LastModifiedDate {

public static void main(String[] args) {

// Specify the pathname

String pathname = "your-file-path-here"; // Replace with your actual file path

// Create a File object for the specified pathname

File file = new File(pathname);

// Check if the file exists

if (file.exists() && file.isFile()) {

// Get the last modified time

long lastModified = file.lastModified();

// Format the last modified time to a readable date

SimpleDateFormat sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss");

String formattedDate = sdf.format(lastModified);

// Print the last modified date

System.out.println("Last modified date of " + pathname + " is: " + formattedDate);

} else {

System.out.println("The specified pathname does not exist or is not a file.");

}

}

}

**Output:**

**Last modified date of C:\Users\aakar\Documents\example.txt is: 05/15/2024 14:30:45**

**7.6 Write a java program to Delete a file which is created in program 6.1**

**Input:**

import java.io.File;

public class DeleteFile {

public static void main(String[] args) {

String filePath = "example.txt"; // This should match the file created in Program 6.1

// Create a File object for the specified file

File file = new File(filePath);

// Check if the file exists

if (file.exists()) {

// Attempt to delete the file

if (file.delete()) {

System.out.println("File " + filePath + " has been deleted successfully.");

} else {

System.out.println("Failed to delete the file " + filePath + ".");

}

} else {

System.out.println("The specified file " + filePath + " does not exist.");

}

}

}

**Output:**

**File example.txt has been deleted successfully.**

**7.7 Write a java program to Read text from file from a specified index or skipping byte using FileInputStream.**

**Input:**

import java.io.FileInputStream;

import java.io.IOException;

public class ReadTextFromFile {

public static void main(String[] args) {

String filePath = "example.txt"; // Specify the file path

int startIndex = 10; // Specify the index to start reading from

int skipBytes = 5; // Specify the number of bytes to skip

try (FileInputStream fis = new FileInputStream(filePath)) {

// Skip the specified number of bytes

long skipped = fis.skip(skipBytes);

System.out.println("Skipped " + skipped + " bytes.");

// Read from the specified index

byte[] buffer = new byte[1024];

int bytesRead = fis.read(buffer, startIndex, buffer.length - startIndex);

if (bytesRead != -1) {

String text = new String(buffer, startIndex, bytesRead);

System.out.println("Text read from file starting at index " + startIndex + ":");

System.out.println(text);

} else {

System.out.println("End of file reached.");

}

} catch (IOException e) {

System.out.println("An error occurred while reading the file.");

e.printStackTrace();

}

}

}

**Output:**

**Skipped 5 bytes.**

**Text read from file starting at index 10:**

**ample text file. It contains some text that we will read using FileInputStream.**