# Day – 10

## I/O File Handling

1. Write a program to create a new text file named test.txt.

Code:

import java.io.File;  
import java.io.IOException;  
  
public class CreateFile {  
 public static void main(String[] args) {  
 try {  
 File file = new File("test.txt");  
 if (file.createNewFile()) {  
 System.out.println("File created: " + file.getName());  
 } else {  
 System.out.println("File already exists.");  
 }  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

2. Write a program to check whether a file exists at a given path.

Code:

import java.io.File;  
  
public class CheckFileExists {  
 public static void main(String[] args) {  
 File file = new File("test.txt");  
 if (file.exists()) {  
 System.out.println("File exists at: " + file.getAbsolutePath());  
 } else {  
 System.out.println("File does not exist.");  
 }  
 }  
}

3. Write a Java program to write "Hello, World!" into a file using FileWriter.

Code:

import java.io.FileWriter;  
import java.io.IOException;  
  
public class WriteToFile {  
 public static void main(String[] args) {  
 try (FileWriter writer = new FileWriter("test.txt")) {  
 writer.write("Hello, World!");  
 System.out.println("Successfully wrote to the file.");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

4. Write a program to read the content of a file line by line using BufferedReader.

Code:

import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class ReadFileLineByLine {  
 public static void main(String[] args) {  
 try (BufferedReader reader = new BufferedReader(new FileReader("test.txt"))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 System.out.println(line);  
 }  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

5. Write a program to append a line of text to an existing file.

Code:

import java.io.FileWriter;  
import java.io.IOException;  
import java.io.BufferedWriter;  
import java.io.File;  
  
public class AppendToFile {  
 public static void main(String[] args) {  
 try (FileWriter fw = new FileWriter("test.txt", true);  
 BufferedWriter writer = new BufferedWriter(fw)) {  
 writer.newLine();  
 writer.write("This is appended text.");  
 System.out.println("Successfully appended to the file.");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

6. Write a program to count the number of lines, words, and characters in a file.

Code:

import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class CountFileStats {  
 public static void main(String[] args) {  
 int lines = 0, words = 0, chars = 0;  
   
 try (BufferedReader reader = new BufferedReader(new FileReader("test.txt"))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 lines++;  
 chars += line.length();  
 String[] wordArray = line.split("\\s+");  
 words += wordArray.length;  
 }  
   
 System.out.println("Lines: " + lines);  
 System.out.println("Words: " + words);  
 System.out.println("Characters: " + chars);  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

7. Write a program to copy content from one file to another using FileReader and FileWriter.

Code:

import java.io.FileReader;  
import java.io.FileWriter;  
import java.io.IOException;  
  
public class CopyFileContent {  
 public static void main(String[] args) {  
 try (FileReader reader = new FileReader("test.txt");  
 FileWriter writer = new FileWriter("copy.txt")) {  
 int character;  
 while ((character = reader.read()) != -1) {  
 writer.write(character);  
 }  
 System.out.println("File copied successfully.");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

8. Write a program that lists all the files in a directory.

Code:

import java.io.File;  
  
public class ListFilesInDirectory {  
 public static void main(String[] args) {  
 File directory = new File(".");  
 File[] files = directory.listFiles();  
   
 if (files != null) {  
 System.out.println("Files in directory:");  
 for (File file : files) {  
 if (file.isFile()) {  
 System.out.println(file.getName());  
 }  
 }  
 }  
 }  
}

9. Write a program to filter and display only .txt files from a folder using FilenameFilter.

Code:

import java.io.File;  
import java.io.FilenameFilter;  
  
public class FilterTxtFiles {  
 public static void main(String[] args) {  
 File directory = new File(".");  
   
 FilenameFilter textFilter = (dir, name) -> name.toLowerCase().endsWith(".txt");  
   
 File[] files = directory.listFiles(textFilter);  
   
 if (files != null) {  
 System.out.println("Text files in directory:");  
 for (File file : files) {  
 System.out.println(file.getName());  
 }  
 }  
 }  
}

10. Write a program to serialize and deserialize a Student object to and from a file.

Code:

import java.io.\*;  
  
class Student implements Serializable {  
 private String name;  
 private int age;  
   
 public Student(String name, int age) {  
 this.name = name;  
 this.age = age;  
 }  
   
 @Override  
 public String toString() {  
 return "Student{name='" + name + "', age=" + age + "}";  
 }  
}  
  
public class StudentSerialization {  
 public static void main(String[] args) {  
 // Serialize  
 Student student = new Student("Alice", 20);  
 try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("student.ser"))) {  
 oos.writeObject(student);  
 System.out.println("Student serialized successfully.");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
   
 // Deserialize  
 try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream("student.ser"))) {  
 Student deserialized = (Student) ois.readObject();  
 System.out.println("Deserialized Student: " + deserialized);  
 } catch (IOException | ClassNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
}

11. Write a program to read a file using Scanner and display the tokens.

Code:

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
  
public class ReadWithScanner {  
 public static void main(String[] args) {  
 try (Scanner scanner = new Scanner(new File("test.txt"))) {  
 scanner.useDelimiter("\\s+");  
 while (scanner.hasNext()) {  
 System.out.println(scanner.next());  
 }  
 } catch (FileNotFoundException e) {  
 System.out.println("File not found.");  
 e.printStackTrace();  
 }  
 }  
}

12. Write a program to search for a specific word in a file and count its occurrences.

Code:

import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
import java.util.Scanner;  
  
public class SearchWordInFile {  
 public static void main(String[] args) throws IOException {  
 Scanner scanner = new Scanner(System.in);  
 System.out.print("Enter word to search: ");  
 String word = scanner.nextLine();  
 int count = 0;  
   
 try (BufferedReader reader = new BufferedReader(new FileReader("test.txt"))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 String[] words = line.split("\\s+");  
 for (String w : words) {  
 if (w.equalsIgnoreCase(word)) {  
 count++;  
 }  
 }  
 }  
 }  
   
 System.out.println("The word '" + word + "' appears " + count + " times.");  
 }  
}

13. Write a program to create, move, and delete a file using Files and Paths.

Code:

import java.io.IOException;  
import java.nio.file.\*;  
  
public class FileOperations {  
 public static void main(String[] args) throws IOException {  
 // Create  
 Path newFile = Files.createFile(Paths.get("temp.txt"));  
 System.out.println("File created: " + newFile);  
   
 // Move  
 Path movedFile = Files.move(newFile, Paths.get("renamed.txt"),   
 StandardCopyOption.REPLACE\_EXISTING);  
 System.out.println("File moved to: " + movedFile);  
   
 // Delete  
 Files.deleteIfExists(movedFile);  
 System.out.println("File deleted.");  
 }  
}

14. Write a program to read all lines of a file using Files.readAllLines() and print them.

Code:

import java.io.IOException;  
import java.nio.file.\*;  
import java.util.List;  
  
public class ReadAllLines {  
 public static void main(String[] args) {  
 try {  
 List<String> lines = Files.readAllLines(Paths.get("test.txt"));  
 System.out.println("File content:");  
 lines.forEach(System.out::println);  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

15. Write a program to write data into a file using Files.write() and append using StandardOpenOption.APPEND.

Code:

import java.io.IOException;  
import java.nio.file.\*;  
import java.util.Arrays;  
import java.util.List;  
  
public class WriteAndAppend {  
 public static void main(String[] args) {  
 List<String> lines = Arrays.asList("First line", "Second line");  
   
 try {  
 // Write  
 Files.write(Paths.get("data.txt"), lines);  
 System.out.println("File written successfully.");  
   
 // Append  
 Files.write(Paths.get("data.txt"),   
 Arrays.asList("Third line"),   
 StandardOpenOption.APPEND);  
 System.out.println("File appended successfully.");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

16. Write a program to walk through a directory tree and display file names using Files.walk().

Code:

import java.io.IOException;  
import java.nio.file.\*;  
  
public class WalkDirectory {  
 public static void main(String[] args) throws IOException {  
 Path start = Paths.get(".");  
 Files.walk(start)  
 .filter(Files::isRegularFile)  
 .forEach(System.out::println);  
 }  
}

17. Write a program to copy a file using Files.copy() with REPLACE\_EXISTING option.

Code:

import java.io.IOException;  
import java.nio.file.\*;  
  
public class CopyWithFiles {  
 public static void main(String[] args) {  
 try {  
 Files.copy(Paths.get("test.txt"),   
 Paths.get("test\_copy.txt"),   
 StandardCopyOption.REPLACE\_EXISTING);  
 System.out.println("File copied successfully.");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

18. Write a program to check and print the size of a file in bytes using Files.size().

Code:

import java.io.IOException;  
import java.nio.file.\*;  
  
public class GetFileSize {  
 public static void main(String[] args) {  
 try {  
 long bytes = Files.size(Paths.get("test.txt"));  
 System.out.println("File size: " + bytes + " bytes");  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}

19. Write a program to serialize a class Employee and store it in employee.ser.

Code:

import java.io.\*;  
  
class Employee implements Serializable {  
 private String name;  
 private int id;  
   
 public Employee(String name, int id) {  
 this.name = name;  
 this.id = id;  
 }  
   
 @Override  
 public String toString() {  
 return "Employee{name='" + name + "', id=" + id + "}";  
 }  
}  
  
public class SerializeEmployee {  
 public static void main(String[] args) {  
 Employee emp = new Employee("John Doe", 101);  
   
 try (ObjectOutputStream oos = new ObjectOutputStream(  
 new FileOutputStream("employee.ser"))) {  
 oos.writeObject(emp);  
 System.out.println("Employee serialized successfully.");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

20. Write a program to deserialize the employee.ser file and display the object data.

Code:

import java.io.\*;  
  
public class DeserializeEmployee {  
 public static void main(String[] args) {  
 try (ObjectInputStream ois = new ObjectInputStream(  
 new FileInputStream("employee.ser"))) {  
 Employee emp = (Employee) ois.readObject();  
 System.out.println("Deserialized Employee: " + emp);  
 } catch (IOException | ClassNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
}