Java 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();
* }
* }
* }
* Output:
* File created: test.txt
* 2. Write a program to check whether a file exists at a given path.

Code:

* import java.io.File;
* import java.util.Scanner;
* public class CheckFileExists {
* public static void main(String[] args) {
* Scanner sc = new Scanner(System.in);
* System.out.print("Enter file path: ");
* String path = sc.nextLine();
* File file = new File(path);
* if (file.exists()) {
* System.out.println("File exists.");
* } else {
* System.out.println("File does not exist.");
* }
* sc.close();
* }
* Output:
* Enter file path: test.txt
* File exists.

}

* 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 WriteFile {
* public static void main(String[] args) {
* try {
* FileWriter writer = new FileWriter("hello.txt");
* writer.write("Hello, World!");
* writer.close();
* System.out.println("Successfully wrote to the file.");
* } catch (IOException e) {
* System.out.println("An error occurred.");
* e.printStackTrace();
* }
* }
* }
* Output:
* Successfully wrote to the file.  
  **hello.txt content:**
* Hello, World!
* 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 ReadFileBuffered {
* public static void main(String[] args) {
* try (BufferedReader br = new BufferedReader(new FileReader("test.txt"))) {
* String line;
* while ((line = br.readLine()) != null) {
* System.out.println(line);
* }
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* This is test file.
* Hello again!
* 5. Write a program to append a line of text to an existing file.
* Code:
* import java.io.FileWriter;
* import java.io.IOException;
* public class AppendFile {
* public static void main(String[] args) {
* try {
* FileWriter writer = new FileWriter("test.txt", true); // true = append mode
* writer.write("\nThis is an appended line.");
* writer.close();
* System.out.println("Text appended successfully.");
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* This is test file.
* Hello again!
* This is an appended line.
* 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 FileCount {
* public static void main(String[] args) {
* String fileName = "test.txt"; // Change file name if needed
* int lineCount = 0, wordCount = 0, charCount = 0;
* try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {
* String line;
* while ((line = br.readLine()) != null) {
* lineCount++;
* // Count characters (including spaces)
* charCount += line.length();
* // Count words (split by spaces, tabs, etc.)
* String[] words = line.trim().split("\\s+");
* if (!line.trim().isEmpty()) { // Avoid counting empty lines as a word
* wordCount += words.length;
* }
* }
* System.out.println("File: " + fileName);
* System.out.println("Number of lines: " + lineCount);
* System.out.println("Number of words: " + wordCount);
* System.out.println("Number of characters: " + charCount);
* } catch (IOException e) {
* System.out.println("An error occurred while reading the file.");
* e.printStackTrace();
* }
* }
* }
* Output:
* Hello World
* Java programming is fun.
* File: test.txt
* Number of lines: 2
* Number of words: 5
* Number of characters: 34
* 7. Write a program to copy content from one file to another using FileReader and FileWriter.
* Code:
* import java.io.\*;
* public class CopyFile {
* public static void main(String[] args) {
* try (FileReader fr = new FileReader("source.txt");
* FileWriter fw = new FileWriter("destination.txt")) {
* int ch;
* while ((ch = fr.read()) != -1) {
* fw.write(ch);
* }
* System.out.println("File copied successfully.");
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* File copied successfully.
* 8. Write a program that lists all the files in a directory.
* Code:
* import java.io.File;
* public class ListFiles {
* public static void main(String[] args) {
* File folder = new File("C:\\testfolder"); // Change path
* File[] files = folder.listFiles();
* if (files != null) {
* for (File f : files) {
* if (f.isFile()) {
* System.out.println(f.getName());
* }
* }
* } else {
* System.out.println("Folder not found.");
* }
* }
* }
* Output:
* data.txt
* image.png
* notes.txt
* 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 folder = new File("C:\\testfolder"); // Change path
* FilenameFilter filter = (dir, name) -> name.toLowerCase().endsWith(".txt");
* String[] txtFiles = folder.list(filter);
* if (txtFiles != null) {
* for (String file : txtFiles) {
* System.out.println(file);
* }
* }
* }
* }
* Output:
* data.txt
* notes.txt
* story.txt
* 10. Write a program to serialize and deserialize a Student object to and from a file.
* Code:
* import java.io.\*;
* class Student implements Serializable {
* int id;
* String name;
* double marks;
* Student(int id, String name, double marks) {
* this.id = id;
* this.name = name;
* this.marks = marks;
* }
* }
* public class SerializeDeserializeStudent {
* public static void main(String[] args) {
* // Serialize
* try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("student.ser"))) {
* Student s = new Student(101, "John", 85.5);
* oos.writeObject(s);
* System.out.println("Student object serialized.");
* } catch (IOException e) {
* e.printStackTrace();
* }
* // Deserialize
* try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream("student.ser"))) {
* Student s = (Student) ois.readObject();
* System.out.println("Deserialized Student:");
* System.out.println("ID: " + s.id);
* System.out.println("Name: " + s.name);
* System.out.println("Marks: " + s.marks);
* } catch (IOException | ClassNotFoundException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Student object serialized.
* Deserialized Student:
* ID: 101
* Name: John
* Marks: 85.5
* 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 ScannerTokens {
* public static void main(String[] args) {
* try (Scanner sc = new Scanner(new File("notes.txt"))) {
* while (sc.hasNext()) {
* System.out.println(sc.next());
* }
* } catch (FileNotFoundException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Java
* is
* fun
* 12. Write a program to search for a specific word in a file and count its occurrences.
* Code:
* import java.io.\*;
* import java.util.\*;
* public class WordSearchCount {
* public static void main(String[] args) {
* String fileName = "notes.txt";
* String searchWord = "Java"; // word to search
* int count = 0;
* try (Scanner sc = new Scanner(new File(fileName))) {
* while (sc.hasNext()) {
* String word = sc.next();
* if (word.equalsIgnoreCase(searchWord)) {
* count++;
* }
* }
* System.out.println("Word '" + searchWord + "' found " + count + " times.");
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Java is fun.
* I love Java programming.
* Java is platform-independent.
* Word 'Java' found 3 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) {
* Path filePath = Paths.get("testfile.txt");
* Path movedPath = Paths.get("movedfile.txt");
* try {
* // Create file
* Files.createFile(filePath);
* System.out.println("File created: " + filePath);
* // Move file
* Files.move(filePath, movedPath, StandardCopyOption.REPLACE\_EXISTING);
* System.out.println("File moved to: " + movedPath);
* // Delete file
* Files.delete(movedPath);
* System.out.println("File deleted: " + movedPath);
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* File created: testfile.txt
* File moved to: movedfile.txt
* File deleted: movedfile.txt
* 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 ReadAllLinesExample {
* public static void main(String[] args) {
* Path path = Paths.get("story.txt");
* try {
* List<String> lines = Files.readAllLines(path);
* System.out.println("File content:");
* for (String line : lines) {
* System.out.println(line);
* }
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Once upon a time, there was Java.
* Java loved coding.
* File content:
* Once upon a time, there was Java.
* Java loved coding.
* 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.nio.charset.StandardCharsets;
* import java.nio.file.StandardOpenOption;
* public class FilesWriteAppend {
* public static void main(String[] args) {
* Path path = Paths.get("sample.txt");
* try {
* // Write new content (overwrites if exists)
* Files.write(path, "Hello, Java!\n".getBytes(StandardCharsets.UTF\_8));
* System.out.println("Initial content written.");
* // Append content
* Files.write(path, "Appending this line.\n".getBytes(StandardCharsets.UTF\_8),
* StandardOpenOption.APPEND);
* System.out.println("Content appended.");
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Initial content written.
* Content appended.  
  Hello, Java!
* Appending this line.
* 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 DirectoryWalker {
* public static void main(String[] args) {
* Path startPath = Paths.get(".");
* try {
* Files.walk(startPath)
* .filter(Files::isRegularFile)
* .forEach(System.out::println);
* } catch (IOException e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Java is popular. Java is powerful. I like Java.
* Enter word to search: Java
* The word 'Java' occurred 3 times in the file.
* 17. Write a program to copy a file using Files.copy() with REPLACE\_EXISTING option.
* Code:
* import java.nio.file.\*;
* public class FileCopyExample {
* public static void main(String[] args) {
* try {
* Path source = Paths.get("source.txt");
* Path destination = Paths.get("destination.txt");
* // Create sample file
* Files.write(source, "This is the source file.".getBytes());
* // Copy with REPLACE\_EXISTING
* Files.copy(source, destination, StandardCopyOption.REPLACE\_EXISTING);
* System.out.println("File copied successfully from source.txt to destination.txt");
* } catch (Exception e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* File copied successfully from source.txt to destination.txt.
* 18. Write a program to check and print the size of a file in bytes using Files.size().
* Code:
* import java.nio.file.\*;
* public class FileSizeExample {
* public static void main(String[] args) {
* try {
* Path filePath = Paths.get("destination.txt");
* long size = Files.size(filePath);
* System.out.println("Size of file '" + filePath + "' is " + size + " bytes.");
* } catch (Exception e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Size of file 'destination.txt' is 27 bytes.
* 19. Write a program to serialize a class Employee and store it in employee.ser.
* Code:
* import java.io.\*;
* class Employee implements Serializable {
* private static final long serialVersionUID = 1L;
* String name;
* int id;
* double salary;
* public Employee(String name, int id, double salary) {
* this.name = name;
* this.id = id;
* this.salary = salary;
* }
* }
* public class EmployeeSerialization {
* public static void main(String[] args) {
* Employee emp = new Employee("John Doe", 101, 55000.50);
* try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("employee.ser"))) {
* oos.writeObject(emp);
* System.out.println("Employee object serialized to employee.ser");
* } catch (Exception e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Employee object serialized to employee.ser
* 20. Write a program to deserialize the employee.ser file and display the object data.
* Code:
* import java.io.\*;
* public class EmployeeDeserialization {
* 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 Data:");
* System.out.println("Name: " + emp.name);
* System.out.println("ID: " + emp.id);
* System.out.println("Salary: " + emp.salary);
* } catch (Exception e) {
* e.printStackTrace();
* }
* }
* }
* Output:
* Deserialized Employee Data:
* Name: John Doe
* ID: 101
* Salary: 55000.5