



Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session 2025-2026

Vision: Dream of where you want.	Mission: Means to achieve Vision
---	---

Program Educational Objectives of the program (PEO): (broad statements that describe the professional and career accomplishments)

PEO1	Preparation	P: Preparation	Pep-CL abbreviation pronounce as Pep-si-IL easy to recall
PEO2	Core Competence	E: Environment (Learning Environment)	
PEO3	Breadth	P: Professionalism	
PEO4	Professionalism	C: Core Competence	
PEO5	Learning Environment	L: Breadth (Learning in diverse areas)	

Program Outcomes (PO): (statements that describe what a student should be able to do and know by the end of a program)

Keywords of POs:

Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

PSO Keywords: Cutting edge technologies, Research

“I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life.” to contribute to the development of cutting-edge technologies and Research.

Integrity: I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

Name and Signature of Student and Date

(Signature and Date in Handwritten)

Vedant Jiwanapurkar



Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session	2025-26 (ODD)	Course Name	Lab : Java Stack
Semester	5	Course Code	CT
Roll No	81	Name of Student	Vedant Jiwanapurkar

Practical Number	4
Course Outcome	Proper Understanding of Basic Java programs covering loops , arrays and conditionals and implementation of programs.
Aim	Practical based on file handling.
Problem Definition	<p>Problem Definition:</p> <ol style="list-style-type: none">1. Write a Java program to append text/ string in a given file.2. Write a Java program to determine number of bytes written to file using DataOutputStream.3. Write a Java program to read content from one file and write it into another file.4. Write a Java program to get the attributes of file.5. read this file and transfer its content in the collection. display the to name of students along with the addition of marks in the three subjects
Theory (100 words)	<p>Java file handling uses the java.io.File class to represent files and directories. Creating a File object is done using constructor.</p> <p>Key methods include:</p> <ul style="list-style-type: none">• boolean exists() – checks if the file exists.• boolean createNewFile() – creates a new file.• boolean delete() – deletes the file.• String getName() – returns the file name.• long length() – returns the size of the file in bytes.• boolean isFile() and boolean isDirectory() – check the type. <p>For reading and writing file contents, Java uses streams like:</p> <ul style="list-style-type: none">• FileReader and FileWriter (character streams)• FileInputStream and FileOutputStream (byte streams)• DataOutputStream for writing primitive data types in binary form.

Procedure and
Execution

(100 Words)

Algorithm:

1. Append Text/String to a File

- Open the file in append mode using FileWriter with append flag true.
- Write the text/string to the file.
- Close the file.

2. Determine Number of Bytes Written Using DataOutputStream

- Create FileOutputStream and wrap in DataOutputStream.
- Write data (e.g., string or primitives) using DataOutputStream methods.
- Close the stream.
- Get file size or use FileChannel.size() to find bytes written.

3. Read Content from One File and Write to Another

- Open input file with FileInputStream or BufferedReader.
- Open output file with FileOutputStream or BufferedWriter.
- Read data byte by byte or line by line from source.
- Write the data to destination file.
- Close both files.

4. Get File Attributes

- Create a File object for the file.
- Use methods like exists(), getName(), length(), canRead(), canWrite() to get attributes.
- Print or return these attribute values.

5. Read File Content into Collection & Display Summed Marks

- Open file using BufferedReader.
- Skip header line.
- For each line:
 - Split by delimiter to get student name and marks for 3 subjects.
 - Calculate sum of marks.
 - Store name and total marks in a collection (e.g., ArrayList or Map).
- Iterate collection and print each student's name with total marks.

Code:

```
1) import java.io.FileWriter;
import java.io.IOException;

public class P4_1 {
    public static void main(String[] args) {
        String data = "\nThis text will be appended to the file.";
        try (FileWriter fw = new FileWriter("student.csv", true)) { //
            true = append mode
            fw.write(data);
            System.out.println("Data appended successfully!");
        }
    }
}
```

```
} catch (IOException e) {
    System.out.println("Error: " + e.getMessage());
}
}
}
2) import java.io.*;

public class P4_2 {
    public static void main(String[] args) {
        String data = "Hello, this is a test.";
        try (FileOutputStream fos = new
FileOutputStream("output.txt");
        DataOutputStream dos = new DataOutputStream(fos)) {
            dos.writeBytes(data);
            System.out.println("Number of bytes written: " +
data.getBytes().length);
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
3) import java.io.*;

public class P4_3 {
    public static void main(String[] args) {
        try (FileReader fr = new FileReader("student.csv");
        FileWriter fw = new FileWriter("copy_student.csv")) {
            int c;
            while ((c = fr.read()) != -1) {
                fw.write(c);
            }
            System.out.println("File copied successfully!");
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
4) import java.io.File;

public class P4_4 {
    public static void main(String[] args) {
        File file = new File("student.csv");

        if (file.exists()) {
            System.out.println("File Name: " + file.getName());
            System.out.println("Path: " + file.getAbsolutePath());
            System.out.println("Readable: " + file.canRead());
            System.out.println("Writable: " + file.canWrite());
            System.out.println("Executable: " + file.canExecute());
            System.out.println("File Size (bytes): " + file.length());
            System.out.println("Last Modified: " + file.lastModified());
        } else {
            System.out.println("File not found!");
        }
    }
}
```

```
}  
}  
}  
5) import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.List;  
  
class Student {  
    private final String name;  
    private final int totalMarks;  
  
    public Student(String name, int totalMarks) {  
        this.name = name;  
        this.totalMarks = totalMarks;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public int getTotalMarks() {  
        return totalMarks;  
    }  
}  
  
public class P4_5 {  
    public static void main(String[] args) {  
        String fileName = "student.csv";  
        try {  
            Path path = Paths.get(fileName);  
            List<Student> studentCollection = Files.lines(path)  
                .skip(1)  
                .filter(line -> !line.trim().isEmpty())  
                .map(line -> line.split(","))  
                .filter(parts -> parts.length >= 8)  
                .map(parts -> new Student(  
                    parts[1].trim(),  
                    Integer.parseInt(parts[5].trim()) +  
                    Integer.parseInt(parts[6].trim()) +  
                    Integer.parseInt(parts[7].trim())  
                ))  
                .toList();  
  
            for (Student student : studentCollection) {  
                System.out.println(student.getName() + "," +  
student.getTotalMarks());  
            }  
        } catch (IOException | NumberFormatException e) {  
            System.err.println("An error occurred: " + e.getMessage());  
        }  
    }  
}
```

Output:

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\Java\jdk-24\bin\java.exe" "-Dfile.encoding=UTF-8" 10,sumedh,08/05/1994,66.66,orange,59,39,66
```

Process finished with exit code 0

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\Java\jdk-24\bin\java.exe" "-Dfile.encoding=UTF-8" 22
```

Number of bytes written: 22

Process finished with exit code 0

copy_student.csv

JavaLab.iml

output.txt

P4_3

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\bin\idea_rt.jar" 1430.0 1430.0 1430.0
```

File copied successfully!

Process finished with exit code 0

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\bin\idea_rt.jar" 1430.0 1430.0 1430.0
```

File Name: student.csv
Path: C:\Users\mithi\IdeaProjects\JavaLab\student.csv
Readable: true
Writable: true
Executable: true
File Size (bytes): 491
Last Modified: 1759855577374

Process finished with exit code 0

```
"C:\Program Files\Java\jdk-24\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\bin\idea_rt.jar" 1430.0 1430.0 1430.0
```

hemant,180
prashant,201
eshwar,175
vishvesh,233
rohit,225
soham,172
srijay,294
ajay,222
sumit,235

Process finished with exit code 0



Nagar Yuwak Shikshan Sanstha's
Yeshwantrao Chavan College of Engineering

(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)

Hingna Road, Wanadongri, Nagpur - 441 110

NAAC A++



Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu

Output Analysis	All five Java programs successfully demonstrate different file handling operations. The first program appends text to an existing file, confirming it with a success message. The second program uses DataOutputStream to write data and displays the exact number of bytes written. The third program copies the contents of one file to another accurately. The fourth program retrieves and displays important file attributes such as name, size, readability, and modification date. Finally, the fifth program reads student data from a file, stores it in a collection, and displays each student's total marks. Overall, these programs effectively showcase how Java handles file input/output operations, data transfer, and basic data analysis.
Link of student Github profile where lab assignment has	https://github.com/vedantjiwanapurkar-ctrl/LAB-JAVA_FSD



Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

been uploaded	
Conclusion	This series of programs demonstrates essential Java File I/O operations — writing, reading, copying, and analyzing file data. By integrating collections, data becomes easier to manage and process efficiently for analytical tasks like mark calculation.
Plag Report (Similarity index < 12%)	<div><div><div>Sentence wise results</div><div>Matched Sources</div></div><div><div>Sources</div><div><div>JAVA - Integer.parseInt () 예외처리</div><div>Compare text</div></div><div><div>https://itsward.tistory.com/entry/Java-Integer.parseInt-%EC%...</div><div><1%</div></div></div><div><div>Save Report</div><div>Download Report</div></div><div><div>96%</div><div>Unique Content</div></div><div><div>4%</div><div>Plagiarized content</div></div><div><div>COMPLETED</div><div>100%</div></div></div>
Date	07/10/25