# Java Practical Assignment – File Handling

# Submit by 1st August 2025

1. Create a Java program that Reads a text file (input.txt) using FileInputStream and Scanner. It should count the frequency of each word (ignore case and punctuation). Outputs the word frequencies to another file (word_count.txt) using FileOutputStream.
   **Example:**
   **Input :**Java is fun. Java is powerful. Java is everywhere!
   **Output:** java: 3
   is: 2
   fun: 1
   powerful: 1
   everywhere: 1
2. Write a Java program that:
   - Reads the contents of a file (input2.txt) using FileInputStream and Scanner.
   - Replaces every vowel in each word with * and every consonant with #.
   - Writes the modified content to a new file (masked_output.txt) using FileOutputStream.

   Example: Input: Hello Java World, Output: #*##* #*#* #*#*#
3. Create a Java program that:
   - Reads all lines from a file (lines.txt) using FileInputStream and Scanner.
   - Sorts the lines in ascending alphabetical order.
   - Reverses each line individually (character-wise).
   - Writes the sorted and reversed lines to a new file (reversed_sorted_lines.txt) using FileOutputStream.
     Example: I/P
     Banana
     Apple
     Grapes
     Output:
     **elppA**
     **ananaB**
     **separG**
4. Write a Java program that:
   - Reads a paragraph from a text file (paragraph.txt) using FileInputStream and Scanner.
   - Replaces **every second occurrence** of each word with the string "REDACTED".
   - Writes the modified paragraph to a new file (redacted_output.txt) using FileOutputStream.
     Example: I/P
     Data is the new oil. Data drives decisions. Data is power.
     O/P : Data is the new oil. REDACTED drives decisions. REDACTED is power.
5. Write a Java program that analyzes a server log file (server_log.txt) containing multiple log entries. Each entry includes a timestamp, a log level, and a message. Your task is to:
   a. **Parse the log file** using FileInputStream and Scanner.

b. **Extract and categorize** log entries by severity (INFO, WARNING, ERROR).
c. **Count** the number of occurrences of each log level.
d. **Find and redact** sensitive data (like email addresses and IP addresses).
e. **Generate two output files**:

- log_summary.txt — A summary of log counts per severity.
- sanitized_log.txt — The modified log with sensitive data redacted.

Example:

**Server_log.txt**

[2025-07-29 10:21:55] INFO User john.doe@example.com logged in from 192.168.1.5
[2025-07-29 10:23:10] WARNING Disk usage at 85%
[2025-07-29 10:25:42] ERROR Failed to connect to database at 10.0.0.2
[2025-07-29 10:26:01] INFO User jane.doe@example.com logged out

**log_summary.txt**

Log Level Summary:
INFO: 2
WARNING: 1
ERROR: 1

**sanitized_log.txt**

[2025-07-29 10:21:55] INFO User [REDACTED_EMAIL] logged in from [REDACTED_IP]
[2025-07-29 10:23:10] WARNING Disk usage at 85%
[2025-07-29 10:25:42] ERROR Failed to connect to database at [REDACTED_IP]
[2025-07-29 10:26:01] INFO User [REDACTED_EMAIL] logged out