Topic Related Questions

**1.Difference between File Reader and Buffered Reader?**

**Ans:-**  **File Reader** → Reads characters from file, no buffering, slower for large files.

 **Buffered Reader** → Wraps File Reader, buffers data in memory (default 8 KB), has readLine() for reading whole lines quickly.

 Best practice: Buffered Reader br = new Buffered Reader(new File Reader("file.txt"));

**2.** **What is try-with-resources?**

**Ans:-**

Java 7+ feature: auto-closes resources after use.

Syntax:-

try (File Reader fr = new File Reader("data.txt")) { ... }

 No need for finally just to close streams.

**3.** **How to handle IO Exception?**

**Ans:-** Use try-catch block:

try { ... } catch (IO Exception e) { e.printStackTrace(); }

 Always show a clear error message.

 Can throw IO Exception to let caller handle it.

**4.What are checked and unchecked exceptions?**

**Ans:** **Checked** → Compile-time check, must handle (IO Exception, SQL Exception).

**Unchecked** → Runtime errors, handling optional (NullPointerException, ArithmeticException).

**5.** **How does file writing work in Java?**

**Ans:-** Data is first written to an internal buffer.

Actual write to file happens when flush() or close() is called.

Example:

File Writer fw = new File Writer("file.txt");

fw.write("Hello");

fw.close();

**6.** **What is the difference between append and overwrite mode?**

**Ans:-** **Append** → new File Writer(file, true) → keeps old content, adds at end.

**Overwrite** → new File Writer(file) → deletes old content before writing.

**7.** **What is exception propagation?**

**Ans:-** If a method doesn’t handle an exception, it’s passed to the caller.

Travels up the stack until caught or program ends.

**8.** **How to log exceptions?**

**Ans:-** For debugging:

e.printStackTrace();

For production: use java.util.logging, Log4j, or SLF4J for storing error details in files.

**9.** **What is a stack trace?**

**Ans:-** Detailed report showing where and why an exception happened.

Includes class name, method name, and line numbers.

**10.** **When to use finally block?**

**Ans:-** Always runs after try (and catch if present).

Used for cleanup: close files, release memory, disconnect DB.

Runs even if an exception is thrown (unless JVM crashes).