Java OOP Interview Questions and Answer  
  
 1. Difference between FileReader and BufferedReader:

* + FileReader reads characters directly from a file, one character at a time, accessing the disk each time, which is slower.
  + BufferedReader wraps any Reader (commonly FileReader), uses an internal buffer (default 8KB), and reads larger blocks from the disk, improving efficiency and speed.
  + BufferedReader has a readLine() method to read a whole line at once, which FileReader does not provide.​

1. Try-with-resources:
   * A Java construct that automatically closes resources (like streams) after use, simplifying resource management and avoiding explicit finally block calls for closing [knowledge].
2. Handling IOException:
   * IOException is handled via try-catch blocks or thrown to the caller with throws keyword.
   * In try-catch, catch the IOException and perform suitable recovery or logging actions [knowledge].
3. Checked and Unchecked Exceptions:
   * Checked exceptions are checked at compile-time and must be either caught or declared, e.g., IOException.
   * Unchecked exceptions are runtime exceptions not checked at compile-time, e.g., NullPointerException.​
4. File writing in Java:
   * File writing uses classes like FileWriter, BufferedWriter.
   * Data can be written by creating writer objects pointing to a file and then using write() methods [knowledge].
5. Difference between append and overwrite mode:
   * Append mode adds data to the end of a file without deleting existing content.
   * Overwrite mode replaces existing data, starting from the beginning of the file.​
6. Exception propagation:
   * When an exception is not caught in a method, it propagates up the call stack to the caller until handled or causes program termination.​
7. Logging exceptions:
   * Use logging frameworks like java.util.logging, Log4j, or printStackTrace() for simple output.
   * Logging captures exception details and stack trace for debugging [knowledge].
8. Stack trace:
   * A stack trace is a report of the active stack frames when an exception occurs.
   * It shows the sequence of method calls leading to the exception, helping identify the origin [knowledge].
9. When to use finally block:
   * The finally block is used to execute code that must run regardless of exceptions, such as closing resources, cleanup operations