**Experiment Report - 11 - test5\_file**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | Line 13 | Logic | √ | × | × |
| E02 | Line 34 | Logic | √ | × | √ |
| E03 | Line 48 | Syntax | × | √ | × |

Additional Errors Found by Self: 0

Self-Review Detection Rate: 67%

Peer 1 Detection Rate: 33%

Peer 2 Detection Rate: 33%

1. **Source Code**
2. import java.io.\*;
3. import java.time.LocalDateTime;
4. import java.time.format.DateTimeFormatter;
5. public class file {
6. public static String readFile(String filePath) throws IOException {
7. String data = null;
9. try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
10. data = br.readLine();
11. }
12. return null;
13. }
14. public void writeFile(String charge, String filePath) {
16. try {
17. // ファイルに書き込む。
18. BufferedWriter writer = new BufferedWriter(new FileWriter(filePath));
19. writer.write(charge);
20. writer.close();
21. System.out.println("ファイルに書き込みました: " + filePath);
22. } catch (IOException e) {
23. System.out.println("エラーが発生しました: " + e.getMessage());
24. e.printStackTrace();
25. }
26. }
27. public void writeFile2(String charge, String filePath, String message) {
28. try {
29. // ファイルを追記モードで開く
30. BufferedWriter writer = new BufferedWriter(new FileWriter(filePath));
32. // 現在の日付と時間を取得
33. LocalDateTime now = LocalDateTime.now();
34. DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
35. String formattedDate = now.format(formatter);
36. // 追記する前に改行を追加する
37. writer.newLine();
38. // chargeと日付をファイルに書き込む
39. writer.write(formattedDate + ": " + charge + message);
40. writer.close();
41. System.out.println("ファイルに書き込みました: " filePath);
42. } catch (IOException e) {
43. System.out.println("エラーが発生しました: " + e.getMessage());
44. e.printStackTrace();
45. }
46. }
47. }