**Experiment Report - 57 - test4\_BankTransferSystem**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | line 15 | Logic | √ | × | √ |
| E02 | line 17 | Semantic | √ | √ | × |
| E03 | line 40 | Semantic | √ | √ | √ |

Additional Errors Found by Self: 0

Self-Review Detection Rate: 100%

Peer 1 Detection Rate: 67%

Peer 2 Detection Rate: 67%

1. **Source Code**
2. package a;
3. import java.util.ArrayList;
4. import java.util.List;
5. class BankAccount06 {
6. private String accountHolder;
7. private double balance;
8. public BankAccount06(String accountHolder, double initialBalance) {
9. this.accountHolder = accountHolder;
10. this.balance = initialBalance;
11. }
12. public boolean transfer(BankAccount06 recipient, double amount, TransactionLogger logger) {
13. if (amount > 0 && amount < this.balance) {
14. this.balance -= amount;
15. recipient.balance -= amount;
16. logger.logTransaction(this.accountHolder, recipient.accountHolder, amount);
17. System.out.printf("Transfer successful! %s -> %s: %.2f%n", this.accountHolder, recipient.accountHolder, amount);
18. return true;
19. } else {
20. System.out.println("Transfer failed! Invalid amount or insufficient funds.");
21. return false;
22. }
23. }
24. public double getBalance() {
25. return balance;
26. }
27. public String getAccountHolder() {
28. return accountHolder;
29. }
30. }
31. class TransactionLogger {
32. private List<String> transactionLog = new ArrayList<>();
33. public void logTransaction(String sender, String recipient, double amount) {
34. String logEntry = String.format("Transaction: %s -> %s | Amount: %.2f%n", sender, recipient, amount);
35. transactionLog.add(logEntry);
36. }
37. public void printTransactionLog() {
38. System.out.println("Transaction History:");
39. for (String log : transactionLog) {
40. System.out.println(log);
41. }
42. }
43. }
44. public class c06\_BankTransferSystem {
45. public static void main(String[] args) {
46. TransactionLogger logger = new TransactionLogger();
48. // testcase-VT:
49. BankAccount06 alice1 = new BankAccount06("Alice", 1000);
50. BankAccount06 bob1 = new BankAccount06("Bob", 500);
51. alice1.transfer(bob1, 120.87, logger); // amount > 0 && amount <= this.balance
52. BankAccount06 alice2 = new BankAccount06("Alice", 1000);
53. BankAccount06 bob2 = new BankAccount06("Bob", 500);
54. alice2.transfer(bob2, 2640.44, logger); // amount > 0 && amount > this.balance
55. BankAccount06 alice3 = new BankAccount06("Alice", 1000);
56. BankAccount06 bob3 = new BankAccount06("Bob", 500);
57. alice3.transfer(bob3, -92, logger); // amount < 0
59. // testcase-FT:
60. // BankAccount06 alice1 = new BankAccount06("Alice", 1000);
61. // BankAccount06 bob1 = new BankAccount06("Bob", 500);
62. // alice1.transfer(bob1, 1.7976931348623157E308, logger); // amount > 0 && amount <= this.balance
63. // BankAccount06 alice2 = new BankAccount06("Alice", 1000);
64. // BankAccount06 bob2 = new BankAccount06("Bob", 500);
65. // alice2.transfer(bob2, 4.9E-324, logger); // amount > 0 && amount > this.balance
66. // BankAccount06 alice3 = new BankAccount06("Alice", 1000);
67. // BankAccount06 bob3 = new BankAccount06("Bob", 500);
68. // alice3.transfer(bob3, 3.313394118237622E307, logger); // amount < 0
70. // print log
71. logger.printTransactionLog();
72. }
73. }