**Bug 1 Debug Log – Player loses double their bet when they lose**Example of buggy behaviour:

daniel bets 10 on Rooster, starting with balance $100

Rolled Gourd, Fish, Prawn

daniel lost 10, balance now 80

Hypothesis 1: value of punter.balance is bad before the bet is lost (Punter.loseBet() function fires)

Prediction 1: When Punter.placeBet() is called, the placeBet() function subtracts the bet from balance. Then when the Punter loses a round, the bet is subtracted again.

Test 1: Place a breakpoint on line Punter.java:104. Observe value of Punter.balance after a round is lost.

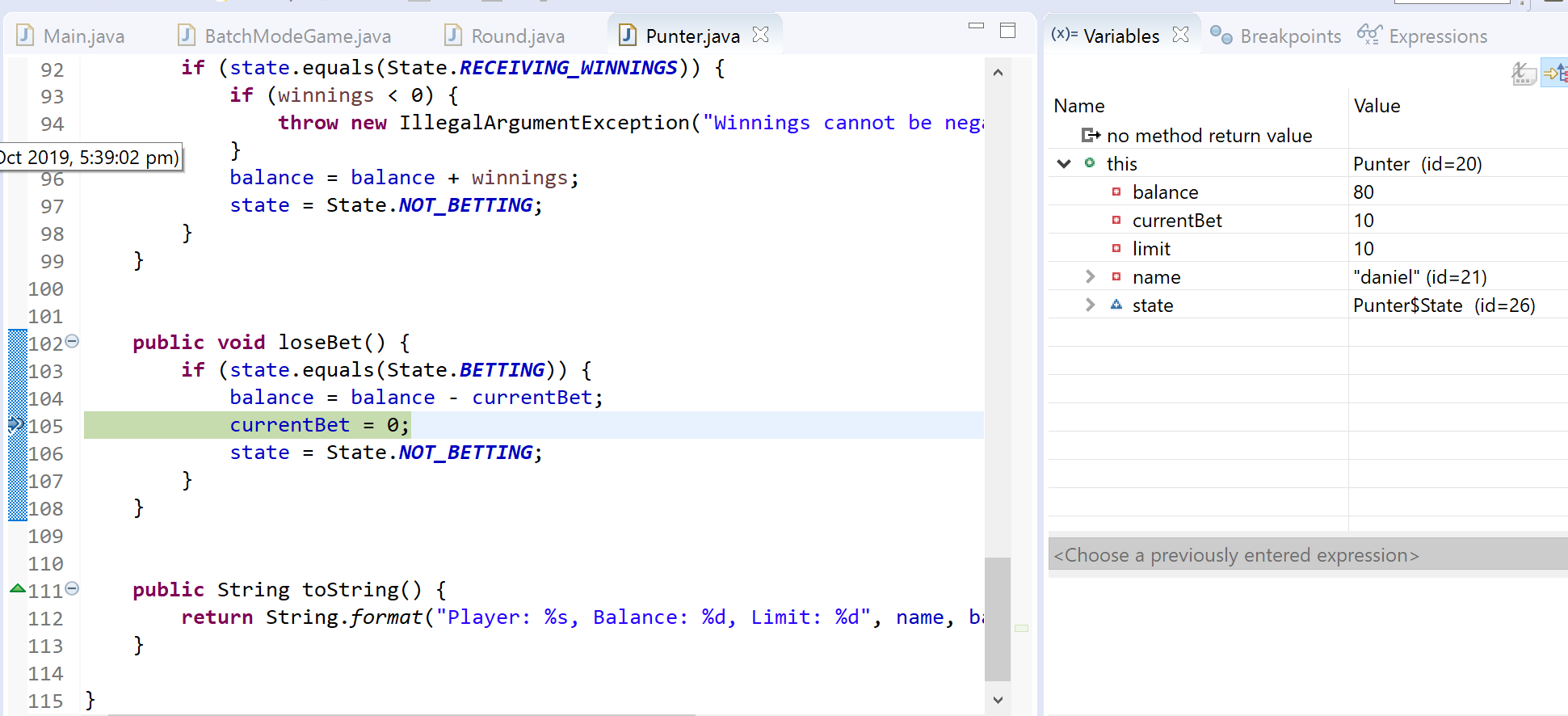
Result: Hypothesis 1 confirmed – Punter.balance arrives at punter.loseBet() with the original bet already subtracted

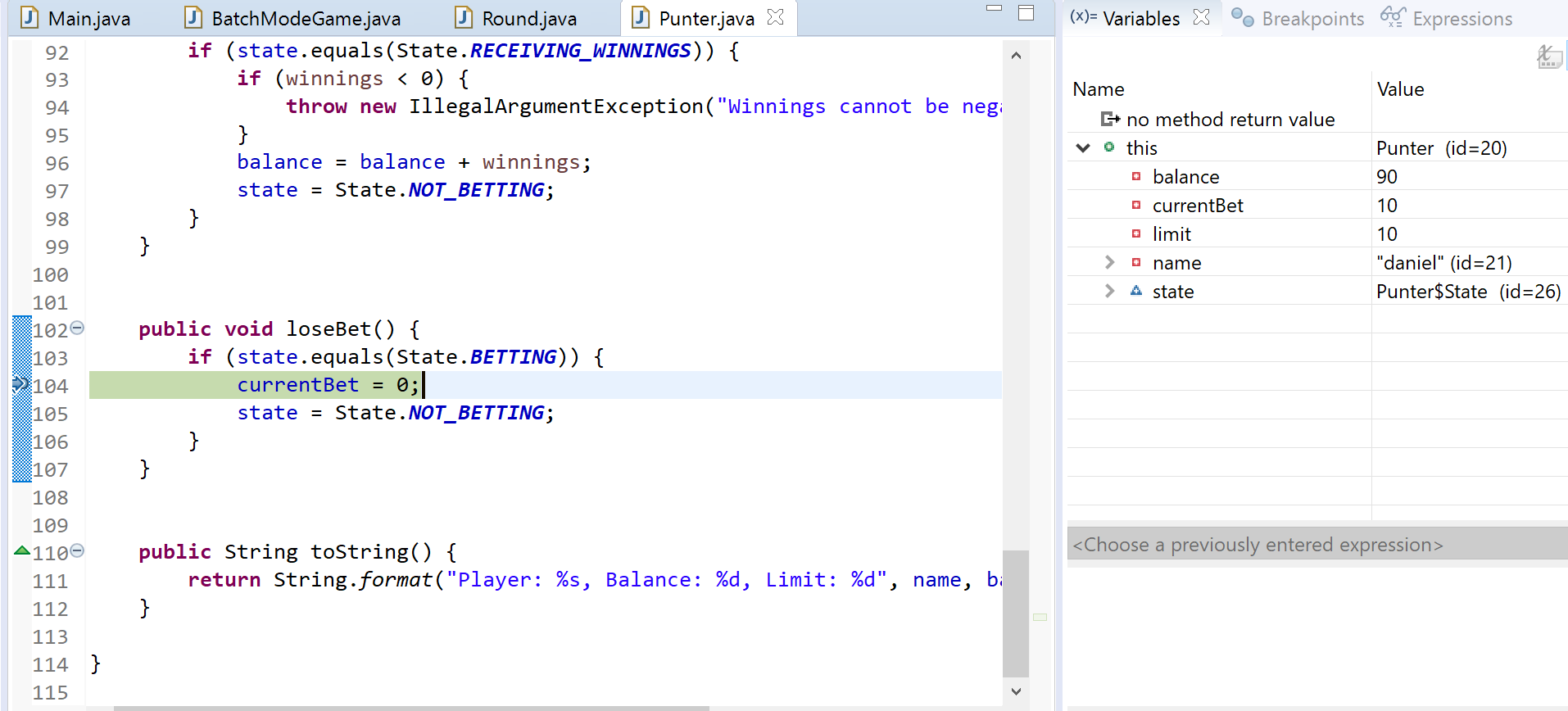
Bug Found: The Punter’s bet is being subtracted from the Punter twice if the Punter loses – once during Punter.placeBet(), and again during Punter.loseBet()

Hypothesis 2: Removing subtraction of bet from Punter.Balance in Punter.loseBet() will resolve bug

Prediction: When a Punter loses a round, the players bet has already been subtracted during Punter.placeBet(). There is no need to subtract the bet again during Punter.loseBet(). Punter.Balance will be the correct amount after Punter.loseBet() runs

Test: Place a breakpoint on line Punter.java:104. Observe value of Punter.balance after a round is lost.

Before:  


After:  


Program out post fix implementation (commit ID xxxxxx)

daniel bets 10 on Gourd, starting with balance $40

Rolled Rooster, Crab, Fish

daniel lost 10, balance now 30

**Bug 2 Debug Log – Player doesn’t receive any winnings**Example of buggy behaviour:

daniel bets 10 on Fish, starting with balance $80

Rolled Gourd, Fish, Prawn

daniel won 10, balance now 80

Hypothesis 1: Punter.returnBet() does not change the state of Punter to receive\_winnings, therefor when the receiveWinnings() function runs, it doesn’t update the balance of the Punter.

Prediction 1: When Punter.returnBet() is called, the state is set to NOT\_BETTING. When Punter.receiveWinnings in called, the state check for RECEIVE\_WINNINGS will return False, meaning the Punter’s balance is never updated with the winnings.

Test 1: Add a breakpoint on Punter:86 and Punter: 97. When the program stops at Punter:86, confirm punter.state is NOT\_BETTING. The program will not stop at Punter: 97 because line 92 will return False. These breakpoints will only fire when the punter wins a round.

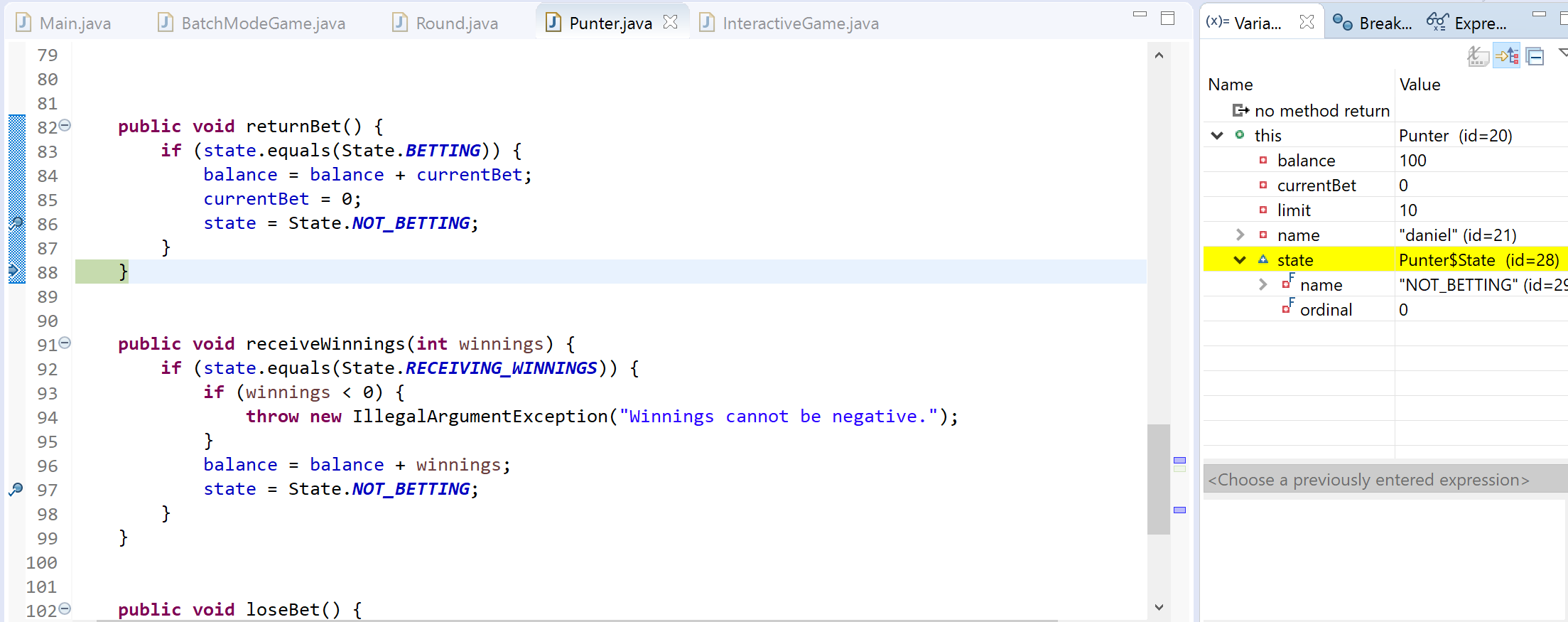
Result: Hypothesis 1 confirmed, due to the state of Punter NOT being RECEIVING\_WINNINGS, the Punter never receives his winnings from the receiveWinnings() function.

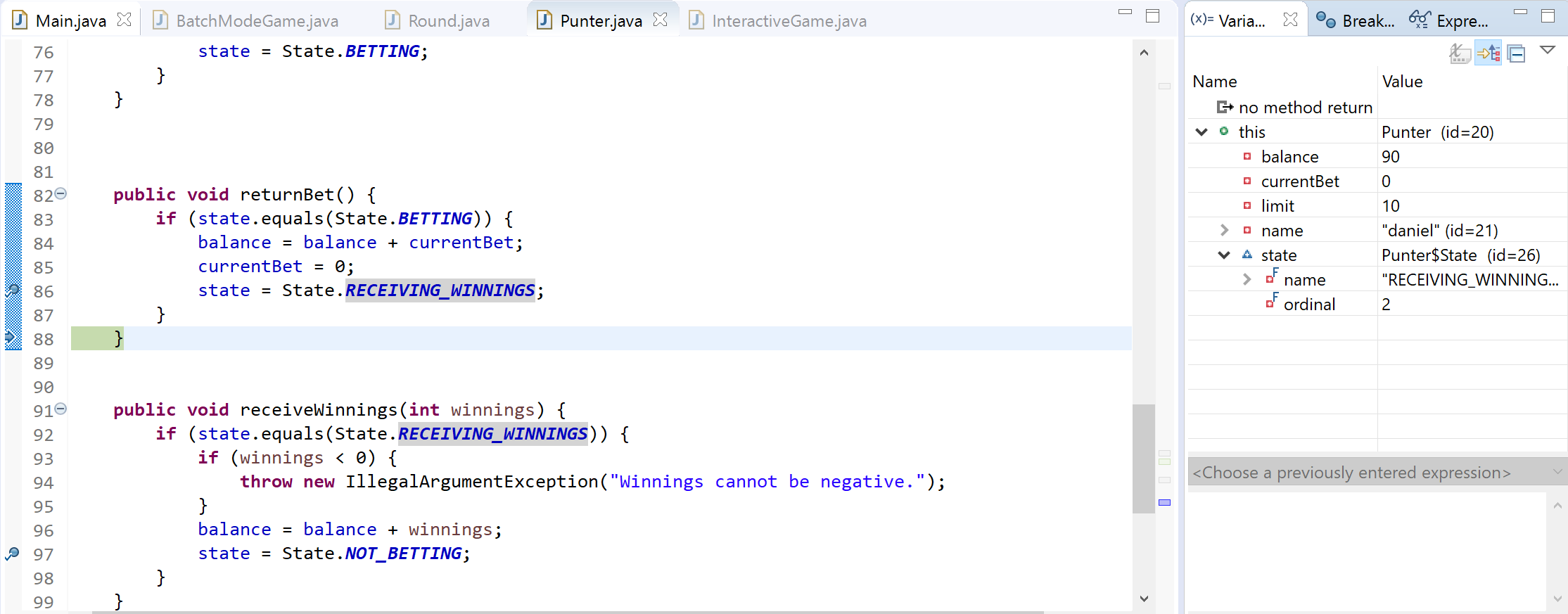
Hypothesis 2: On Punter:86, change the state from NOT\_BETTING to RECEIVE\_WINNINGS

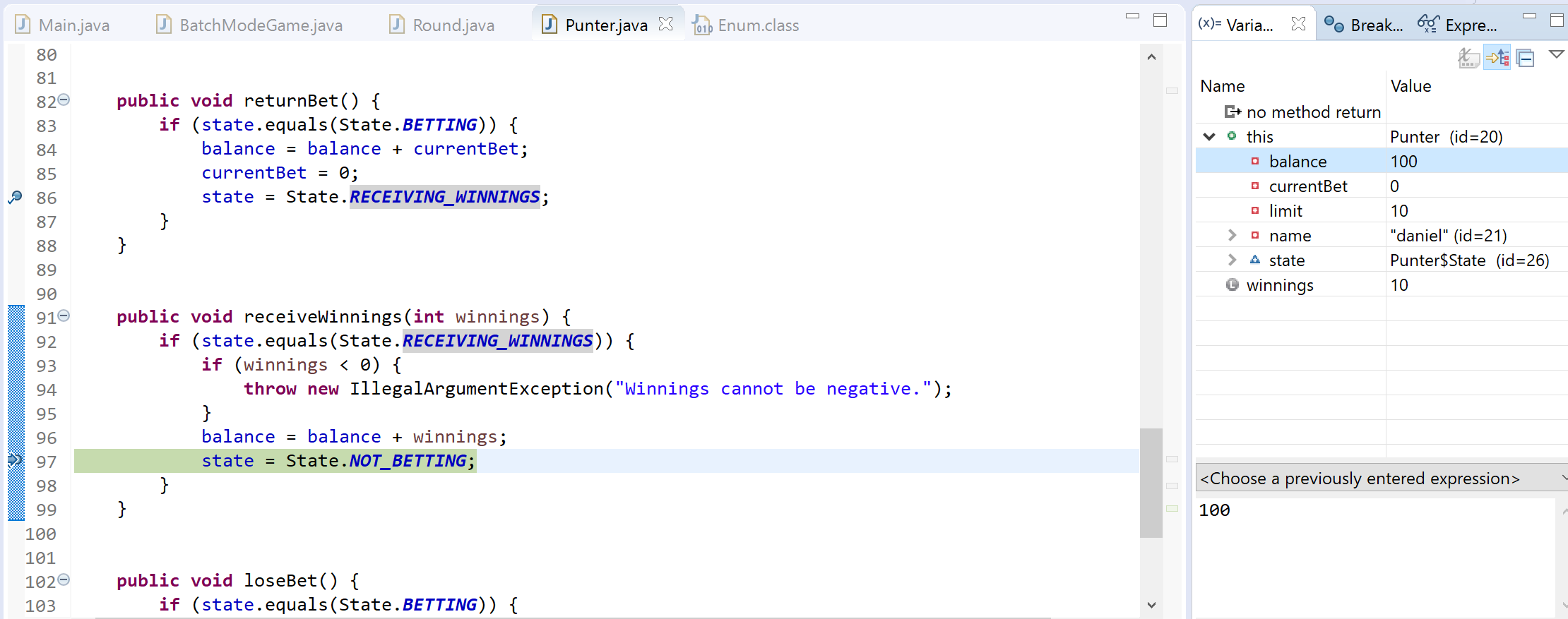
Prediction 2: After setting the Punter state to RECEIVE\_WINNINGS, Punter.receiveWinnings function will run and add the winnings to Punters balance.

Test 2: Add a breakpoint on Punter:86 and Punter: 97. When the program stops at Punter:86, confirm punter.state is RECEIVING\_WINNINGS. The program WILL stop at Punter: 97 and we’ll see the expected balance with the Punters winnings.

Result: Hypothesis 2 confirmed, player now receives the correct winnings.

Before:  


After:  




Program out post fix implementation (commit ID xxxxxx)

daniel bets 10 on Fish, starting with balance $90

Rolled Rooster, Prawn, Fish

daniel won 10, balance now 100

**Bug 3 Debug Log – Player cannot reach betting limit**

Example of buggy behaviour:

daniel lost 10, balance now 110

Play again (Y/N)? (default: Y)

Select Symbol: 1 - Fish, 2 - Prawn, 3 - Crab, 4 - Rooster, 5 - Gourd, 6 - Stag

2

Selected Prawn.

Enter bet (default $10): 100

Betting 100 could go below limit, voiding bet

Select Symbol: 1 - Fish, 2 - Prawn, 3 - Crab, 4 - Rooster, 5 - Gourd, 6 - Stag

2

Selected Prawn.

Enter bet (default $10): 99

daniel bets 99 on Prawn, starting with balance $110

Rolled Crab, Gourd, Fish

daniel lost 99, balance now 11

Hypothesis 1: Somewhere in the code there is a greater than sign (>) when it should be greater than or equal to (>=)

Prediction 1: The punter.balanceExceedsLimitBy() function returns TRUE if the balance – amount > limit when it needs to return FALSE

Test 1: Place a breakpoint on Punter:61. Observe that the *balance* – *amount* is equal to *limit,* and as a result punter.balanceExceedsLimitBy() returns FALSE.

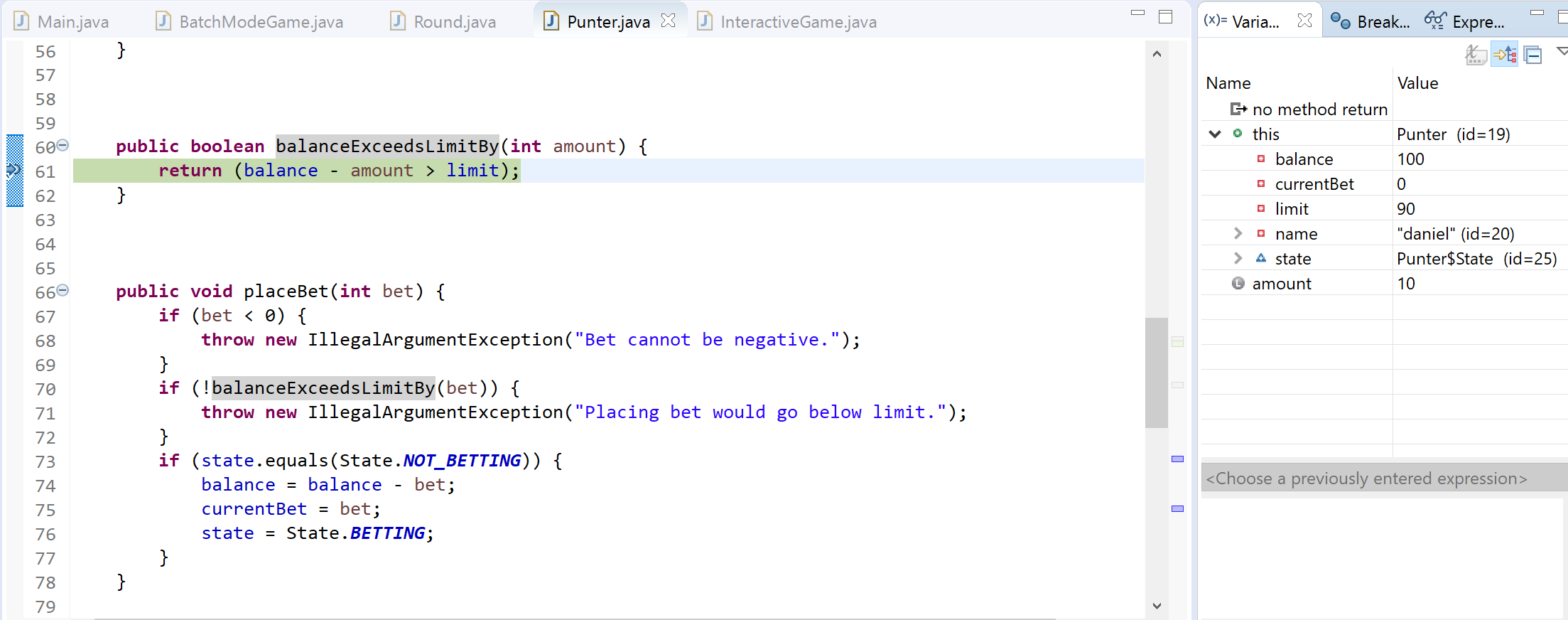
Result: Hypothesis 1 confirmed

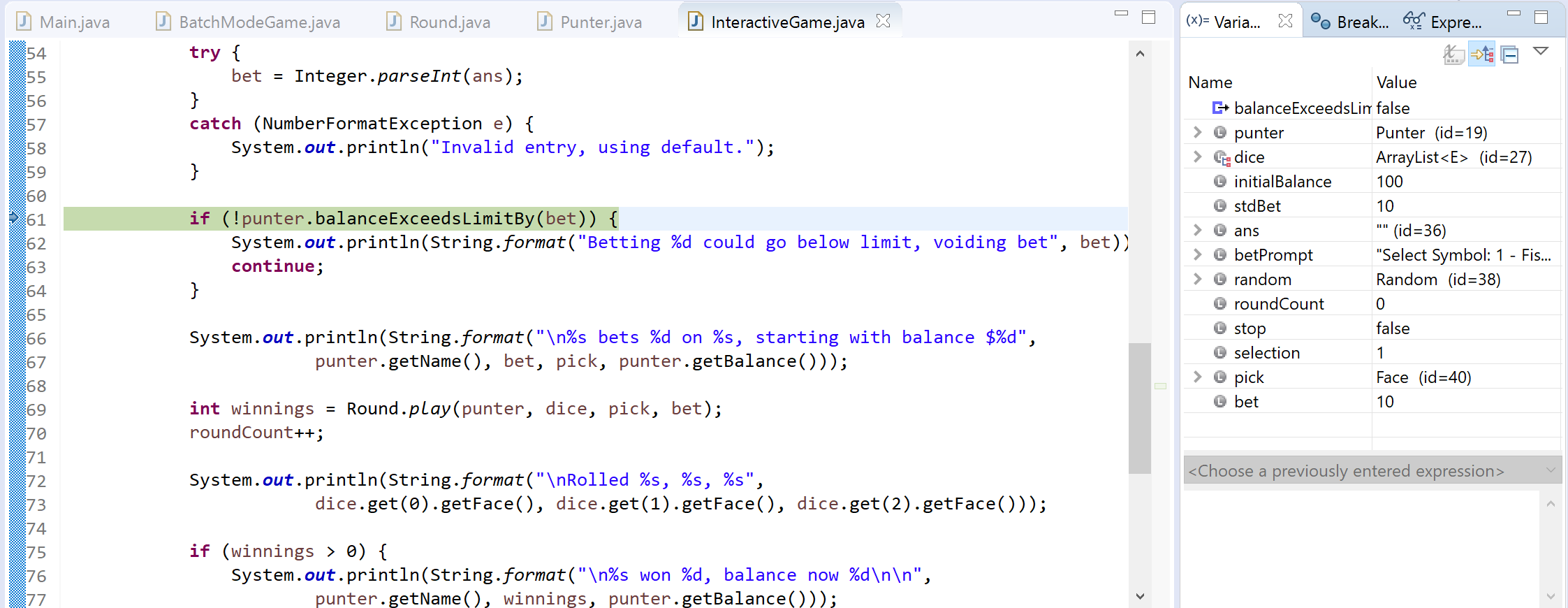
Hypothesis 2: changing the greater than sign (>) to greater or equal to (>=) should allow a Punter to match his limit

Prediction 2: Using greater than or equal will allow a Punter to math his limit

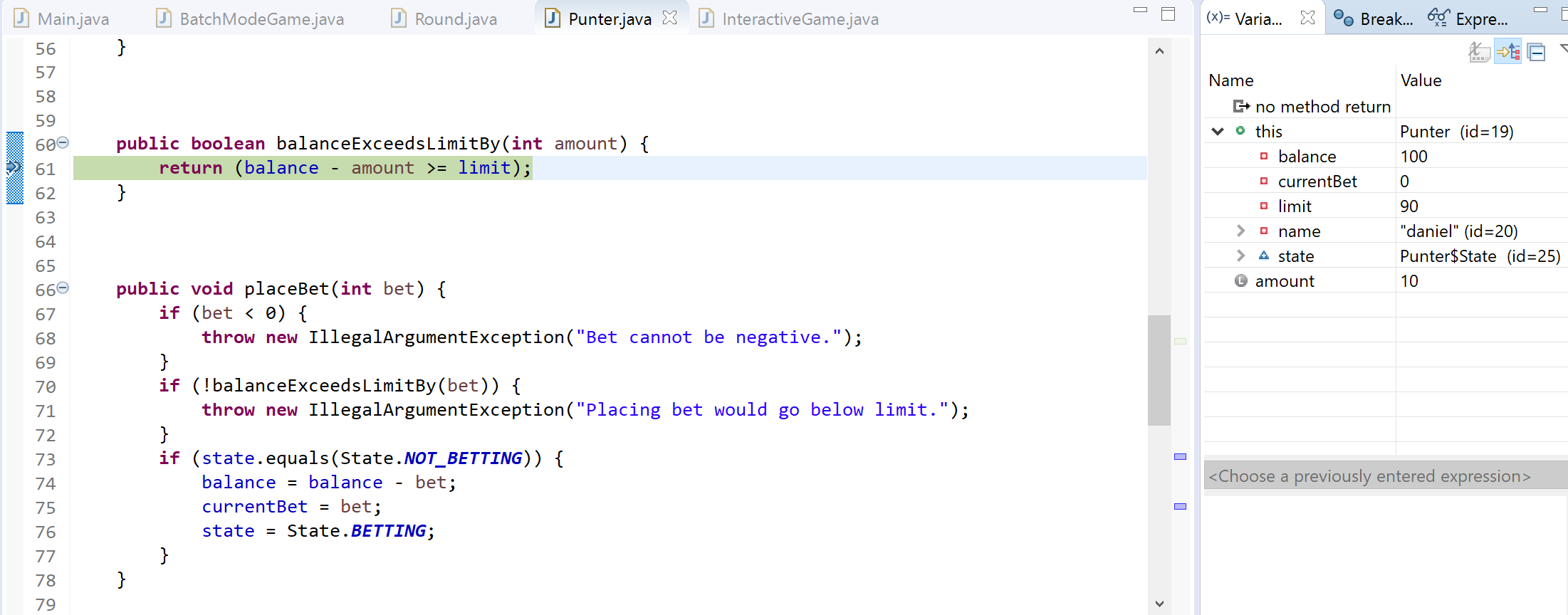
Test 2: Place a breakpoint on Punter:61. Observe that the *balance* – *amount* is equal to *limit,* and as a result punter.balanceExceedsLimitBy() returns TRUE.

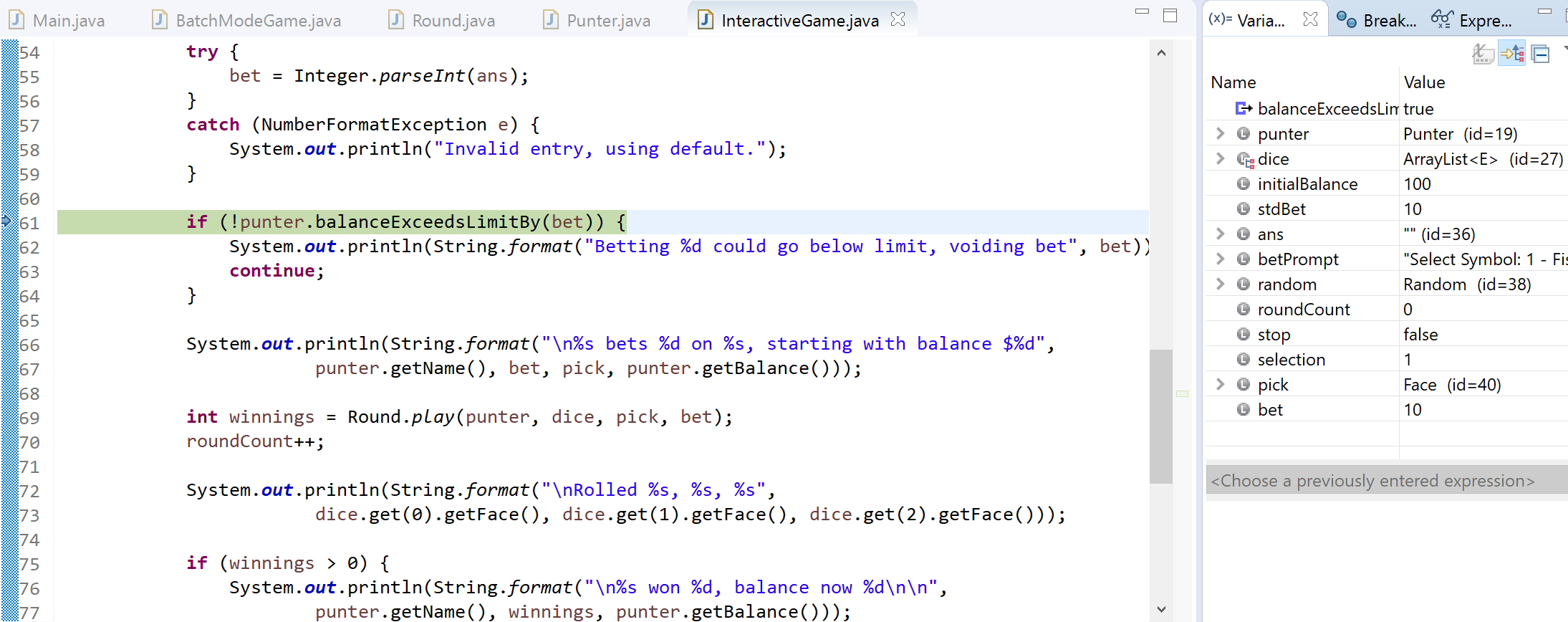
Before:





After:





Program out post fix implementation (commit ID xxxxxx)

Initialising player daniel with balance $100.00 and limit $90.00

playInteractive

Enter standard bet (default 10):

Invalid entry, using default.

Starting interactive game for daniel with initial balance $100.00, limit $90.00, and standard bet 10.00

Select Symbol: 1 - Fish, 2 - Prawn, 3 - Crab, 4 - Rooster, 5 - Gourd, 6 - Stag

1

Selected Fish.

Enter bet (default $10):

Invalid entry, using default.

daniel bets 10 on Fish, starting with balance $100

Rolled Rooster, Rooster, Fish

daniel won 10, balance now 110

**Bug 4 Debug Log – Odds in the game are incorrect**