# Bug 2 – Debugging Log

The bug essentially boils down to the Main logic not playing a round despite the Player having enough balance to do so. So we look at the Main class to see which methods handle the round repetition logic.

This logic is handled by a while loop: while (player.balanceExceedsLimitBy(bet) && player.getBalance() < 200) {

The second part of the loop stops the game if the player wins too much. So, the method called in the first part of the while loop declaration was suspect.

This method, balanceExceedsLimitBy(int amount), is a simple Boolean logic check. If the player’s balance subtracting the amount passed exceeds the players lower limit, it returns true. However, given this statement : return (balance - amount > limit); and the context it is used in, I can see the problem is in the Boolean operator “>”.

I added a statement to print a line to the console showing the relevant values and the result of the logic check being used. This showed that the logic check being used was causing the bug. This is the output of the test demonstrating this:

Bug 2

John's balance is 100

Limit is 0, can John play with balance of 100 & bet of 5? Ans: true

John bet 5 on SPADE

Rolled HEART, CROWN, ANCHOR

Limit is 0, can John play with balance of 100 & bet of 5? Ans: true

John lost, balance now 95

…

…

…

Limit is 0, can John play with balance of 15 & bet of 5? Ans: true

John bet 5 on SPADE

Rolled HEART, CROWN, ANCHOR

Limit is 0, can John play with balance of 15 & bet of 5? Ans: true

John lost, balance now 10

Limit is 0, can John play with balance of 10 & bet of 5? Ans: true

John bet 5 on SPADE

Rolled HEART, CROWN, ANCHOR

Limit is 0, can John play with balance of 10 & bet of 5? Ans: true

John lost, balance now 5

Limit is 0, can John play with balance of 5 & bet of 5? Ans: false

John now has balance 5

------------- ---------------- ---------------

Testcase: testBug2(CrownAndAnchor.Bug2Replication): FAILED

expected:<0> but was:<5>

When the balance, subtracting the amount, is greater than the players limit, the function returns true, allowing the player to keep playing. However, this means that in the case of a player having exactly enough balance to play one more round, the two sides of the equation would be equal and the method would return false, preventing the player from playing that final round.

To fix the bug, the simplest method would be to change the “>” operator to an “>=” operator. This would provide the required functionality. Here are screenshots of the code, before and after:



