Bug Hypothesis and Tracing

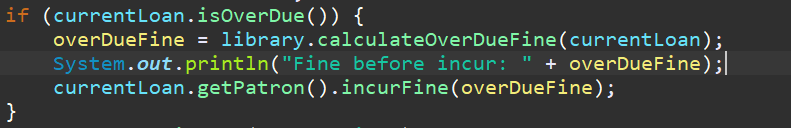
*Bug Name – Return Book Incorrect Fine*

# Hypothesis of cause

|  |  |  |
| --- | --- | --- |
| Hypothesis | Description | Was this the cause? |
| That in the returnBookControl the bug occurs |  | No |
| That the Patron incurFine method is broken |  | No |
| The currentLoan.getPatron().incurFine as seen the ReturnBookControl class is the issue | The method is called twice, once in the ReturnBookControl and in the dischargeLoan which calls it in the library class. | Yes |

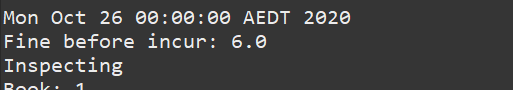
# Tests to narrow down bug

## Idea



Test that the value before being incurred is actually correct (should be from previous tests)

### Output:



### Conclusion:

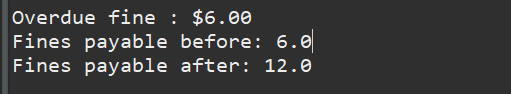
The value returns 6, which is correct in this case! Hence is in the IncurFines method of the patron class.

## Idea 2

### 

Test the finesPayable value to test what its state is before, and after.

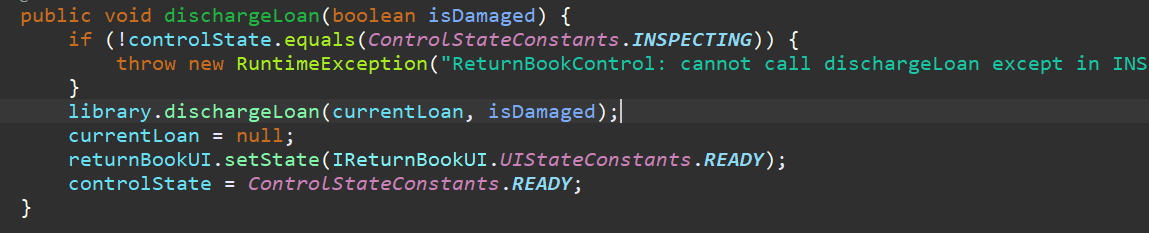
### Output



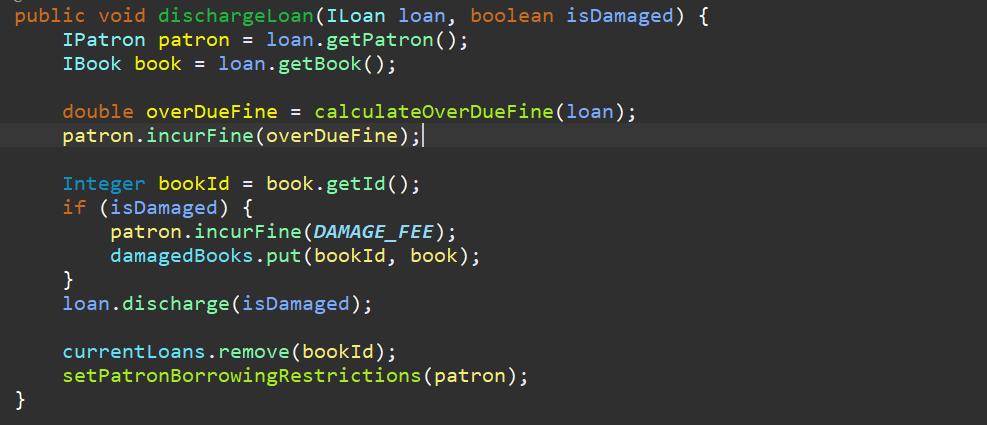
### Conclusion

The issue is somewhere else in the code, the method logic is sound. Hence the issue is in another method that also calls the incurFine method elsewhere.

## Idea 3

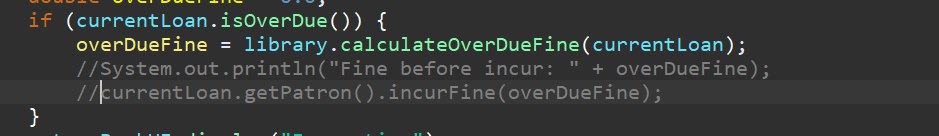


This method calls library.dischargeLoan(), but this method call does the following:

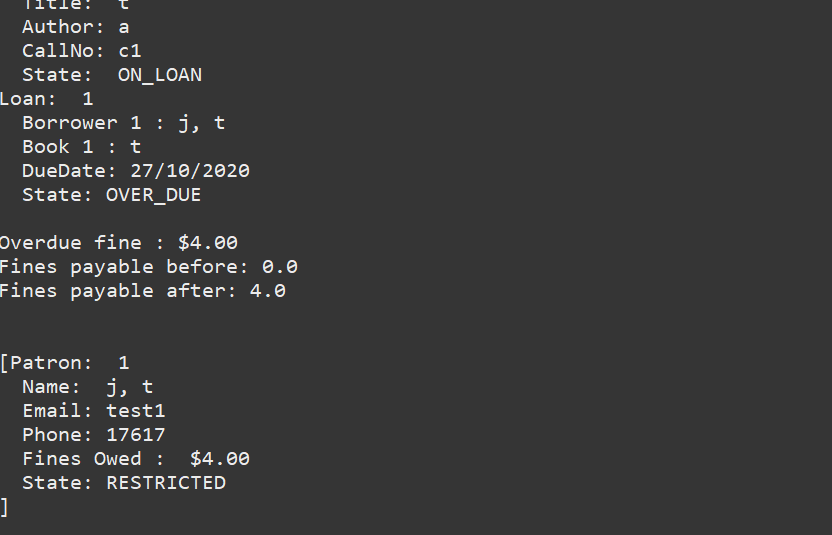


Hence this means that from the other call to calculateOverDueFine, we end up processing the value twice, hence why our value is already 6.

By removing the following code, the issue is resolved.



### Output

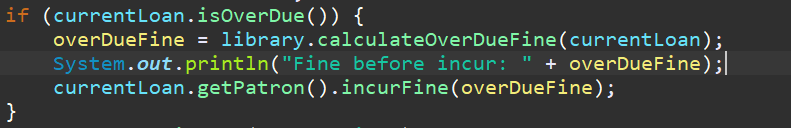


### Conclusion

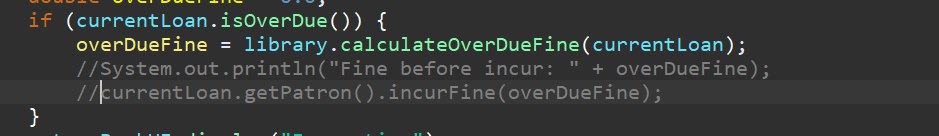
The error is fixed.

# Resolution

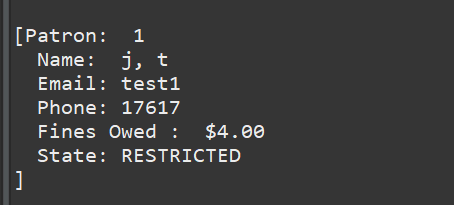
Buggy Code:



After fix:



## Result Automated Test



Hence Issue has been resolved, and was due to the double calling of the method.

## Result Manual Test

