# A4 BUG1 REPORT

# Name: Ty Saunders

Student ID: 11748199

Assignment details: Debugging task

## Bug 1: Incorrect calculation of fines.

When an item is returned late, customer is quoted a correct late fee of $1 per item for each day late. However, the amount being recorded against their account is 3 times the expected amount.

Contents – Click links to jump to section:

1. [Demonstration](#Demonstration)
2. [Replication](#Replication)
3. [Simplification](#Simplification)
4. [Tracing](#Tracing)
5. [Resolution](#Resolution)
6. [Validation](#Validation)

1. Demonstration:

|  |  |
| --- | --- |
| When an item is returned late: |  |
| And the patron tries to pay their fine  The amount is 3 times greater than expected: |  |
|  |  |

1. Replication:

**System Test:**

The steps to manually replicate the error are contained in a separate document in the same directory as this document: Bug1SystemTest.pdf.

Here is a link to the document on github for quick reference: [Bug1SystemTest.pdf](https://github.com/tysau/A4debugging/blob/main/BugReports/Bug1SystemTest.pdf)

In summary it needs setup for a loan to be created, date to be incremented, and then follows the return item late use case.

1. Simplification:

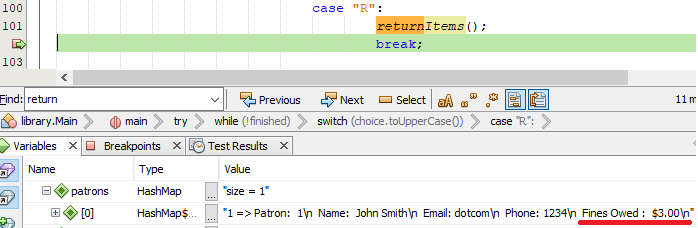
This section uses Hypothesis / Test / Result process to narrow down the cause to the point where a simplified automated test can be written to reproduce it easily.

* H1 – The payFineUI is displaying the result wrong
* T1 – Do a static review of the PayfineUI to look for errors.
* R1 – FALSE - Did not see anything at all related to displaying the fine or calculating it – moving on.
* H2 - Main method displaying wrong amount when PayFine is chosen
* T2 - Static review of MAIN.java to see what it might be displaying when PayFine is called.
* R2 – FALSE: Main.java does not display anything directly. Main calls PayFineUI, PayFineUI calls

PayFineControl.cardswiped(), cardSwiped method gets patron id and tells the UI to display the patron object.

Investigated the Patron code for display issues - can see that Patron’s toString method is not doing anything other than getting the fines owed, so issue is nothing to do with payFine. Value is not sane while PayFine is running but PayFine does nothing to modify the value. It exists before the PayFine process.

* H3: Issue exists after ReturnItemControl has completed
* T3: Run a manual test with a breakpoint after ReturnItemControl completes processing.
* R3: TRUE – Ran a test to return an overdue item and found that after the process has completed the fine amount is now $3.00 which is triple what it should be.

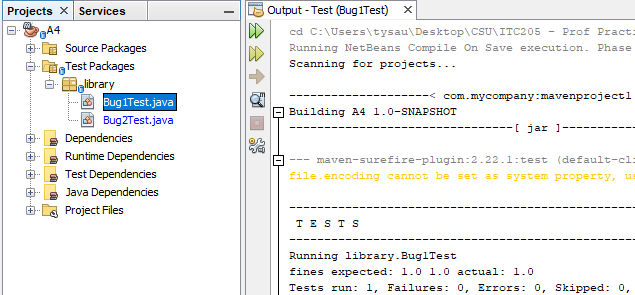


**Automated Test:**

From the simplification process we realise we can create a test using the Return Item Control to return an overdue item, then check the result of the finesOwed variable when return item finishes.

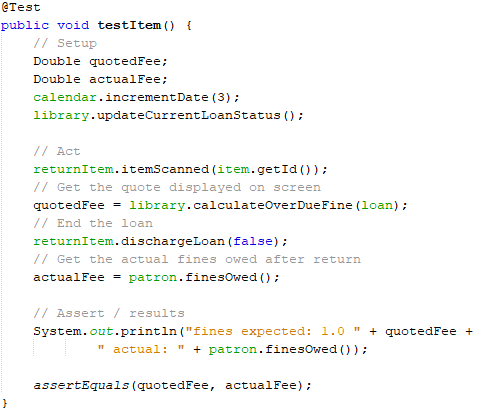
Test code can be found in this directory: **A4\src\test\java\library.Bug1Test.java**

The test can be run by opening the project, right-click Bug1Test.jave and click “Test”:

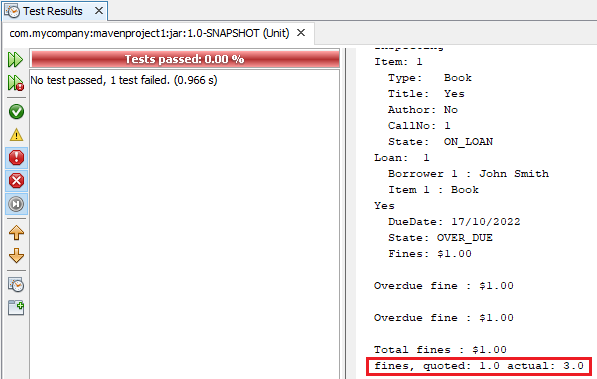


**Screenshot of the test code:**

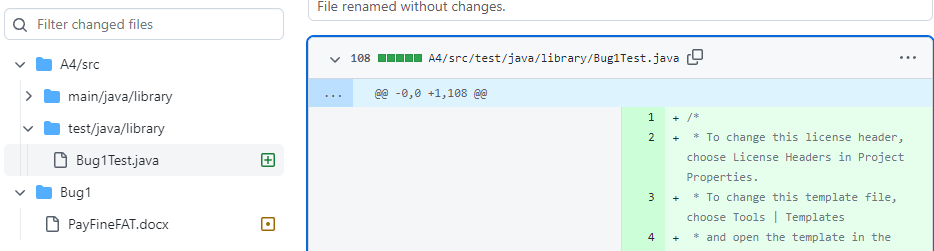
# 



**Screenshot of the test being run and failing as expected before the fix:**

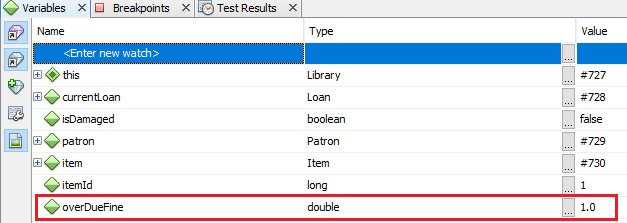
****

**Version control: Screenshot of commit for the test code:**

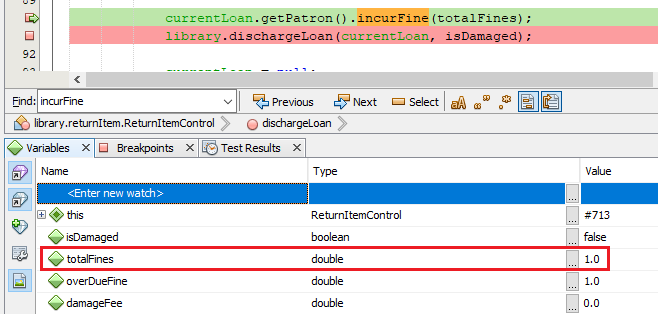


1. Tracing:

* H1: Patron’s incurFine method might be getting called with the wrong value.
* T1: Find where incurFine is being called and check what value is being passed by setting a break point.
* R1: FALSE: incurFine gets called by Library during dischargeLoan using currentLoan.getFines as an input. Breakpoint was set and found that the variable is being set correctly here:

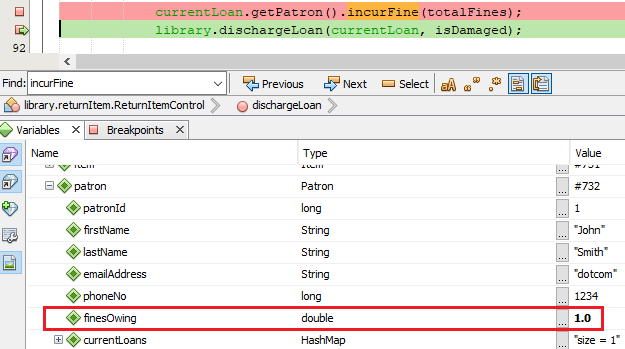


* H2 - Patron’s incurFine method might be getting called more than once.
* T2 - Scan the code base for other uses of incurFine method. Set a breakpoint on this line to check what value is being passed.
* R2 - TRUE: Found incurFine is getting called by ReturnItemControl during its dischargeLoan method using its “totalFines” variable as an input. By setting a break point we can see that incurFine is being called with 1.0. So we have accounted for $2 out of the $3, but there is still 1 extra fine being added somewhere.

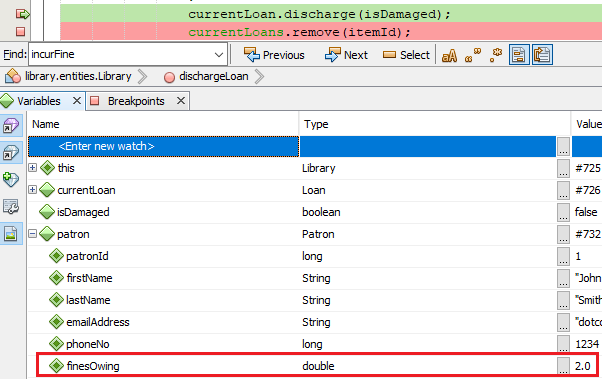


* H3 - Another method is adding an extra fine somehow.
* T3 - Set break points to get the patrons fine amount after both incurFine methods and after library.dischargeLoan is complete.
* R3 - TRUE: After patron.incurFine is called within ReturnItemControl.dischargeLoan, the patrons late fees are $1.00. After incurFine is called again within Library.dischargeLoan, the late fee is $2.00. After Loan.discharge is called, the late fee increases to $3.00. Therefore another issue must exist somewhere during **Loan.discharge**.

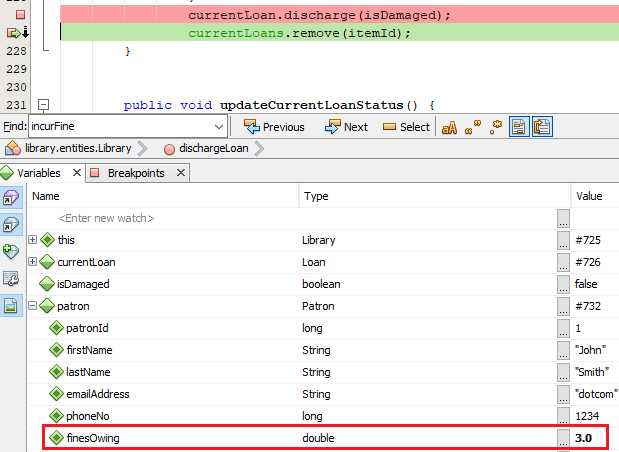
Patron.finesOwing = $1 after first time Patron.incurFine is called:



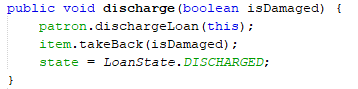
**Patron.finesOwing = $2 after the second time Patron.incurFine is called:**



**Patron.finesOwing = $3 after loan.discharge is complete:**

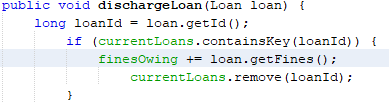


* **H5 - Another issue is occurring during Loan.discharge(), adding to the fine amount**
* **T5 - Review the Loan.discharge method to see where it is adding to the fee.**
* **R5 – Partially true: Loan.discharge is not adding any money directly. There are some methods it is calling from other classes which need to be investigated - Patron.dischargeLoan is the first of those methods we will investigate.**

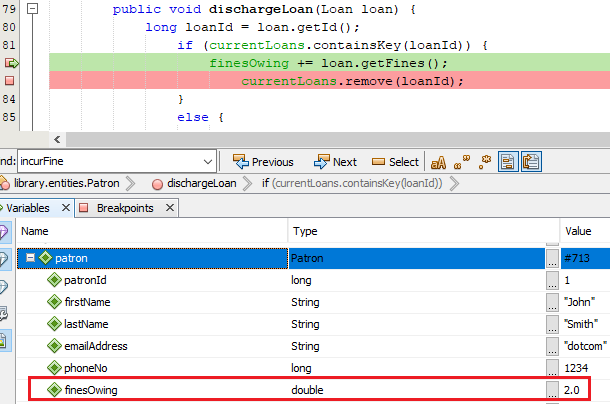
****

* **H6 - Patron.dischargeLoan is adding to the late fee amount.**
* **T6 - Review of Patron.dischargeLoan to check what it is doing.**
* **R6 - TRUE – We can see that Patron.dischargeLoan is directly adding to the overdue fines:**

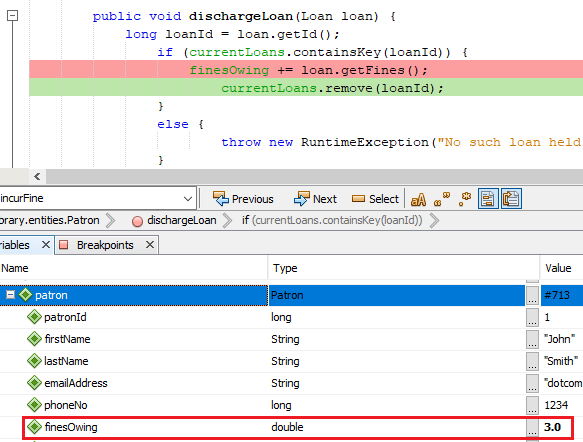
**Screenshot of Patron.dischargeLoan, showing it adding fines:**

****

**Before patron.dischargeLoan the fine amount is $2:**

****

**After patron.dischargeLoan the fine amount is $3:**

****

Tracing - conclusion:

**There are 2 problems –**

1. Patron.incurFine is being called twice, by ReturnItemControl and Library:

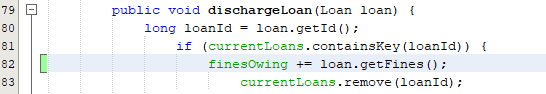
**ReturnItemControl.dischargeLoan:**



**Library.dischargeLoan:**



1. Patron.dischargeLoan is directly adding to the fine as well:



1. Resolution

The patron finesOwing variable is being added to 3 times instead of once. We could resolve this issue 3 different ways – 2 of the methods which add in fines need to be removed, and we must keep one.

Which one to keep: We see that Patron has a public incurFine method, so it makes sense that we use it. Therefore we can eliminate fine being added in patron.dischargeLoan. It also makes no sense for the return Item control to be having authority over fines so therefore we should remove the method from here as well. The patron.incurFine() method will be called during Library.dischargeLoan and will be removed from all other places it is called.

Fix details:

**Plan:**

* **Use a separate code branch other than main.**
* **Apply Fix 1, run test to confirm bug 1 is now showing a fine of $2 instead of $3 – which is an improvement on before the fix.**
* **Apply Fix 2, run test to confirm bug 1 is now resolved and the fine is now $1.**

**FIX 1: ReturnItemControl:**

**Remove line 90 from ReturnItemControl’s dischargeLoan method:**

****

**FIX 2: Patron:**

1. **Remove line 82 from Patron’s dischargeLoan method:**

****

1. Validation (Fix results):

**Fix 1 results – ReturnItemControl:**

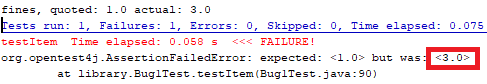
**Code before:**



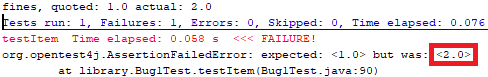
**Code after:**

****

**Test results before:**

****

**Test results after:**

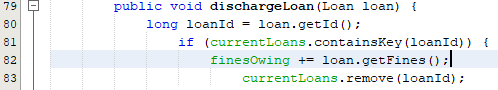
****

**Result:**

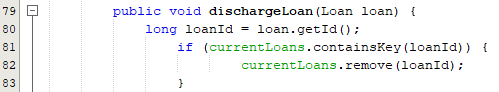
**FIXED – The amount of fine has decreased by $1 which means FIX 1 is successful.**

FIX 2 Deployment: Patron:

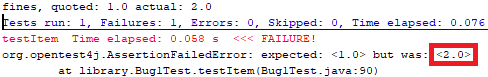
**Code before:**



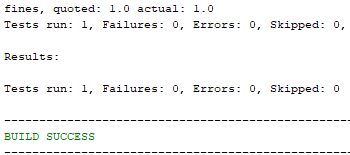
**Code after:**

****

**Before fix test results:**

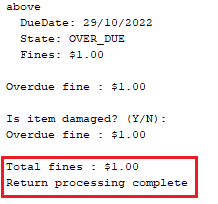
****

**After fix test results:**

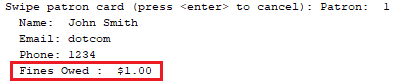
****

**User Test after fix deployment:**

**When returning item:**

****

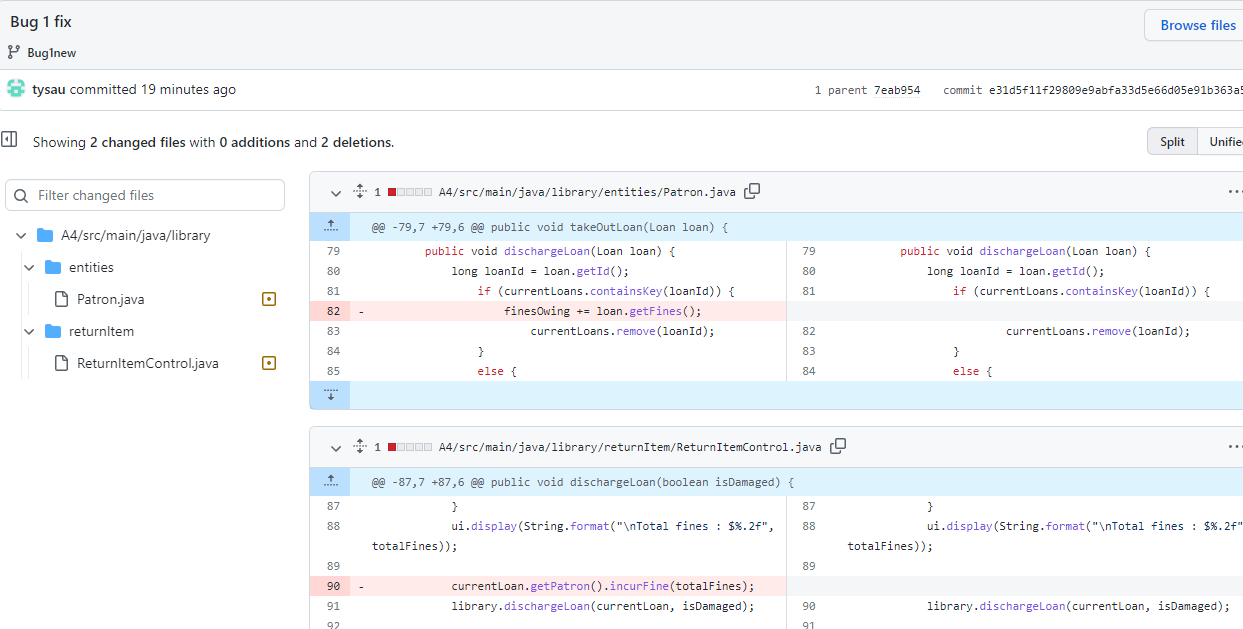
**When paying the fine:**

****

**Result:**

**FIXED – The amount of fine is now the correct amount which means FIX is successful.**

**Fix deployed and committed to repo - GitHub commit screenshot:**

****