

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

1. Demonstration:

When an item is returned late:

```
Scan Item (<enter> completes): 1
Inspecting
Item: 1
  Type:   Book
  Title:  Yes
  Author: No
  CallNo: 1
  State:  ON_LOAN
Loan:   7
  Borrower 1 : John Smith
  Item 1 : Book
Yes
  DueDate: 23/10/2022
  State: OVER_DUE
  Fines: $1.00

Overdue fine : $1.00
Is item damaged? (Y/N): n

Overdue fine : $1.00
Total fines : $1.00
Return processing complete
```

And the patron tries to pay their fine

The amount is 3 times greater than expected:

```
Choice :
P
Pay Fine Use Case UI

Swipe patron card (press <enter> to cancel): 1
Patron: 1
  Name:   John Smith
  Email:  dotcom
  Phone:  1234
  Fines Owed : $3.00

Enter amount (<Enter> cancels) : |
```

2. 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](#)

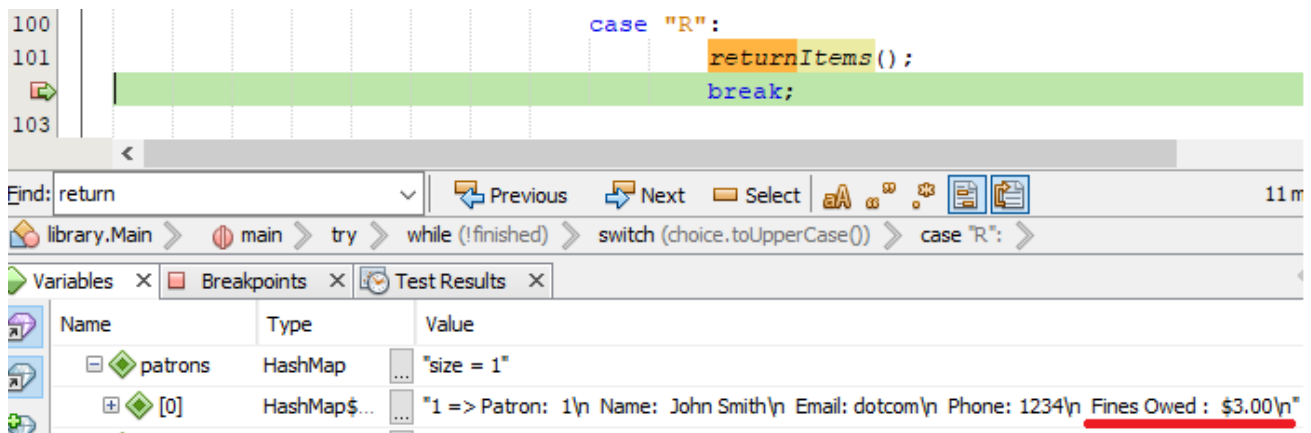
In summary it needs setup for a loan to be created, date to be incremented, and then follows the return item late use case.

3. 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.

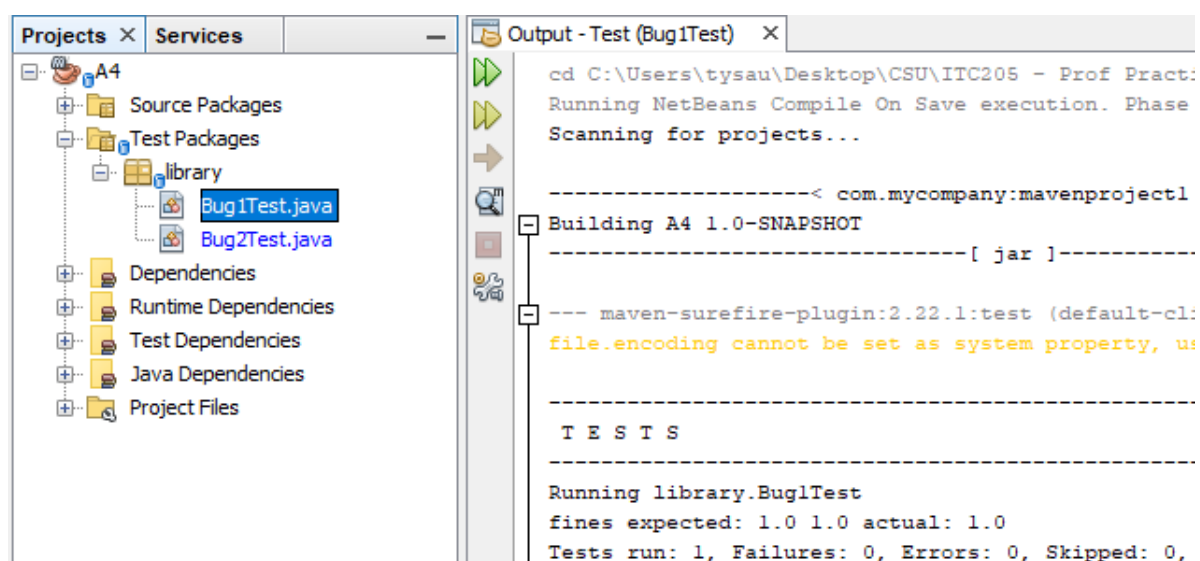


Automated Test:

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.java and click "Test":



Screenshot of the test code:

```
@ExtendWith(MockitoExtension.class)
public class Bug1Test {

    Library library;
    Loan loan;
    Patron patron;
    Calendar calendar;
    Item item;
    @Mock ReturnItemUI returnItemUI;
    ReturnItemControl returnItem;

    public Bug1Test() {
    }

    @BeforeEach
    public void setUp() {
        // Create library, patron, item, calendar, loan, returnItem control
        library = Library.getInstance();

        patron = library.addPatron("John", "Smith", "dotcom", 1234);
        item = library.addItem("No", "Yes", "1", ItemType.BOOK);
        calendar = Calendar.getInstance();
        loan = library.issueLoan(item, patron);

        returnItem = new ReturnItemControl();
        returnItem.setUi(returnItemUI);
    }

    @AfterEach
    public void tearDown() {
    }

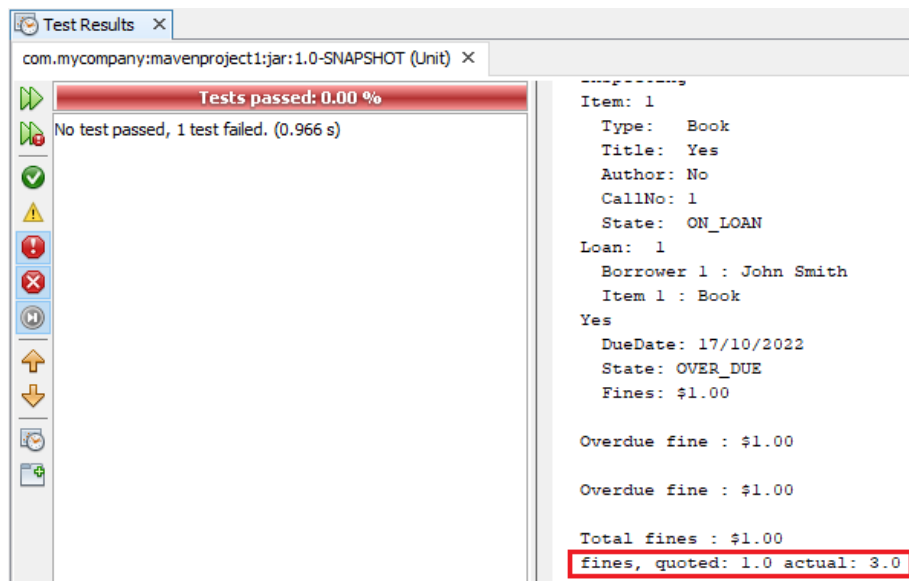
    @Test
    public void testItem() {
        // Setup
        Double quotedFee;
        Double actualFee;
        calendar.incrementDate(3);
        library.updateCurrentLoanStatus();

        // Act
        returnItem.itemScanned(item.getId());
        // Get the quote displayed on screen
        quotedFee = library.calculateOverDueFine(loan);
        // End the loan
        returnItem.dischargeLoan(false);
        // Get the actual fines owed after return
        actualFee = patron.finesOwed();

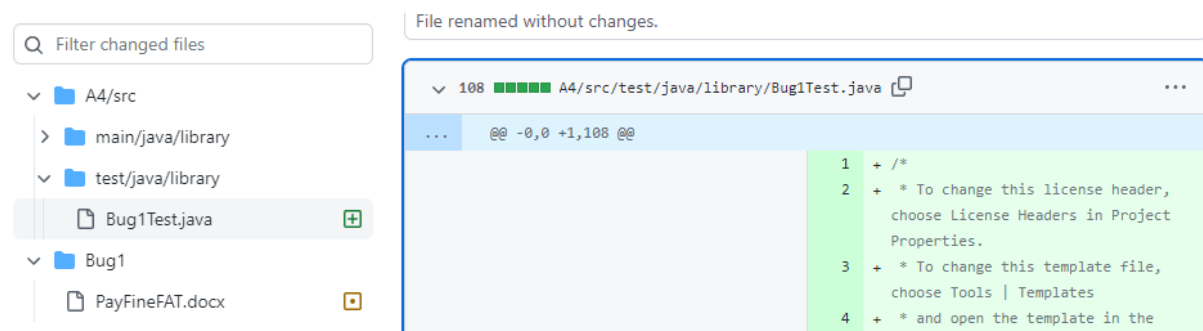
        // Assert / results
        System.out.println("fines expected: 1.0 " + quotedFee +
            " actual: " + patron.finesOwed());

        assertEquals(quotedFee, actualFee);
    }
}
```

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

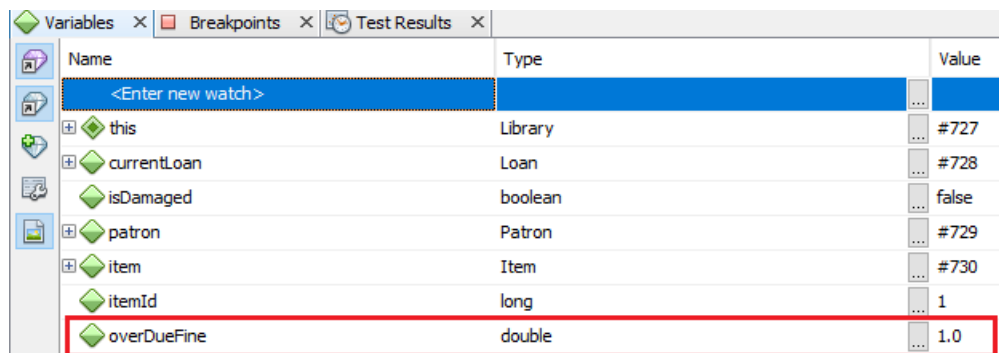


Version control: Screenshot of commit for the test code:



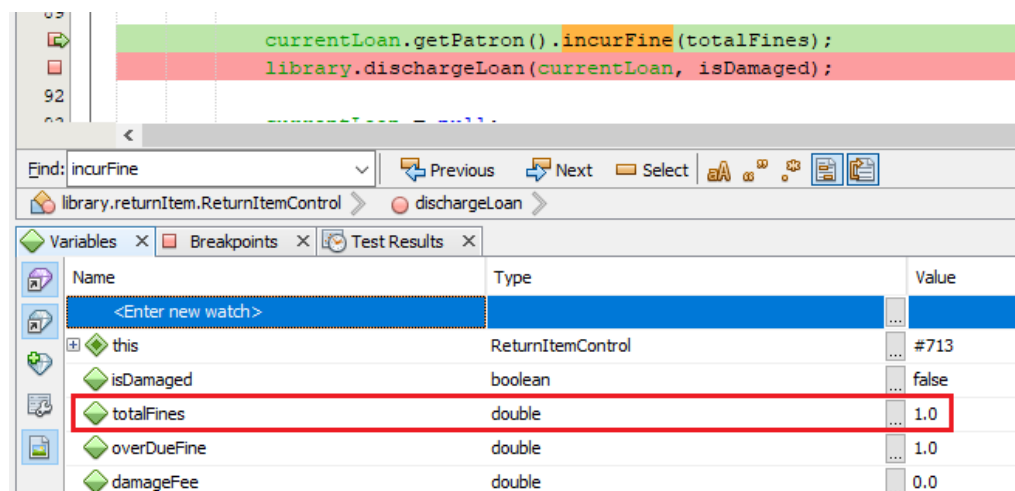
4. Tracing:

- 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:



Name	Type	Value
<Enter new watch>		
this	Library	#727
currentLoan	Loan	#728
isDamaged	boolean	false
patron	Patron	#729
item	Item	#730
itemId	long	1
overDueFine	double	1.0

- 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 breakpoint 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.



```
currentLoan.getPatron().incurFine(totalFines);
library.dischargeLoan(currentLoan, isDamaged);
```

Find: incurFine

library.returnItem.ReturnItemControl > dischargeLoan >

Name	Type	Value
<Enter new watch>		
this	ReturnItemControl	#713
isDamaged	boolean	false
totalFines	double	1.0
overDueFine	double	1.0
damageFee	double	0.0

- 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:

The screenshot shows an IDE with a breakpoint set at the `incurFine` method in `library.returnItem.ReturnItemControl`. The Variables window is open, showing the state of the `patron` object. The `finesOwing` property is highlighted with a red box and has a value of 1.0.

Name	Type	Value
patron	Patron	#732
patronId	long	1
firstName	String	"John"
lastName	String	"Smith"
emailAddress	String	"dotcom"
phoneNo	long	1234
finesOwing	double	1.0
currentLoans	HashMap	*size = 1*

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

The screenshot shows an IDE with a breakpoint set at the `incurFine` method in `library.entities.Library`. The Variables window is open, showing the state of the `patron` object. The `finesOwing` property is highlighted with a red box and has a value of 2.0.

Name	Type	Value
<Enter new watch>		
this	Library	#725
currentLoan	Loan	#726
isDamaged	boolean	false
patron	Patron	#732
patronId	long	1
firstName	String	"John"
lastName	String	"Smith"
emailAddress	String	"dotcom"
phoneNo	long	1234
finesOwing	double	2.0

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

The screenshot shows an IDE with a code editor and a variables window. The code editor displays a method `updateCurrentLoanStatus()` with two lines highlighted: `currentLoan.discharge(isDamaged);` and `currentLoans.remove(itemId);`. The variables window shows the state of the program, with the `patron.finesOwing` variable highlighted in red and showing a value of 3.0.

Name	Type	Value
<Enter new watch>		...
this	Library	#725
currentLoan	Loan	#726
isDamaged	boolean	false
patron	Patron	#732
patronId	long	1
firstName	String	"John"
lastName	String	"Smith"
emailAddress	String	"dotcom"
phoneNo	long	1234
finesOwing	double	3.0

- 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.

```
public void discharge(boolean isDamaged) {  
    patron.dischargeLoan(this);  
    item.takeBack(isDamaged);  
    state = LoanState.DISCHARGED;  
}
```

- 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:

```
public void dischargeLoan(Loan loan) {  
    long loanId = loan.getId();  
    if (currentLoans.containsKey(loanId)) {  
        finesOwing += loan.getFines();  
        currentLoans.remove(loanId);  
    }  
}
```

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

The screenshot shows the `dischargeLoan` method in the `Patron` class. The `if` block is highlighted in green, indicating it is the current execution point. Below the code, the variable table shows the state of the `patron` object.

Name	Type	Value
patron	Patron	#713
patronId	long	1
firstName	String	"John"
lastName	String	"Smith"
emailAddress	String	"dotcom"
phoneNo	long	1234
finesOwing	double	2.0

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

The screenshot shows the `dischargeLoan` method in the `Patron` class. The `if` block is highlighted in green, indicating it is the current execution point. Below the code, the variable table shows the state of the `patron` object after the method has executed.

Name	Type	Value
patron	Patron	#713
patronId	long	1
firstName	String	"John"
lastName	String	"Smith"
emailAddress	String	"dotcom"
phoneNo	long	1234
finesOwing	double	3.0

Tracing - conclusion:

There are 2 problems –

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

ReturnItemControl.dischargeLoan:

```
90 | | | | | currentLoan.getPatron().incurFine(totalFines);  
91 | | | | | library.dischargeLoan(currentLoan, isDamaged);  
92 | | | | |
```

Library.dischargeLoan:

```
218 | | | | | if (currentLoan.isOverDue()) {  
219 | | | | | double overDueFine = currentLoan.getFines();  
220 | | | | | patron.incurFine(overDueFine);
```

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

```
79 | □ | | | | public void dischargeLoan(Loan loan) {  
80 | | | | | long loanId = loan.getId();  
81 | | | | | if (currentLoans.containsKey(loanId)) {  
82 | ■ | | | | finesOwing += loan.getFines();  
83 | | | | | currentLoans.remove(loanId);
```

5. 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:

```
💡 | | | | | currentLoan.getPatron().incurFine(totalFines);
```

FIX 2: Patron:

1. Remove line 82 from Patron's dischargeLoan method:

```
82 | | | | | finesOwing += loan.getFines();
```

6. Validation (Fix results):

Fix 1 results – ReturnItemControl:

Code before:

```
90 |         currentLoan.getPatron().incurFine(totalFines);
91 |         library.dischargeLoan(currentLoan, isDamaged);
92 |
```

Code after:

```
90 |         library.dischargeLoan(currentLoan, isDamaged);
91 |
92 |         currentLoan = null;
```

Test results before:

```
fines, quoted: 1.0 actual: 3.0
Tests run: 1, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 0.075
testItem Time elapsed: 0.058 s <<< FAILURE!
org.opentest4j.AssertionFailedError: expected: <1.0> but was: <3.0>
    at library.Bug1Test.testItem(Bug1Test.java:90)
```

Test results after:

```
fines, quoted: 1.0 actual: 2.0
Tests run: 1, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 0.076
testItem Time elapsed: 0.058 s <<< FAILURE!
org.opentest4j.AssertionFailedError: expected: <1.0> but was: <2.0>
    at library.Bug1Test.testItem(Bug1Test.java:90)
```

Result:

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

FIX 2 Deployment: Patron:

Code before:

```
79 | public void dischargeLoan(Loan loan) {
80 |     long loanId = loan.getId();
81 |     if (currentLoans.containsKey(loanId)) {
82 |         finesOwing += loan.getFines();
83 |         currentLoans.remove(loanId);
```

Code after:

```
79 public void dischargeLoan(Loan loan) {  
80     long loanId = loan.getId();  
81     if (currentLoans.containsKey(loanId)) {  
82         currentLoans.remove(loanId);  
83     }
```

Before fix test results:

```
fines, quoted: 1.0 actual: 2.0  
Tests run: 1, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 0.076  
testItem Time elapsed: 0.058 s <<< FAILURE!  
org.opentest4j.AssertionFailedError: expected: <1.0> but was: <2.0>  
    at library.Bug1Test.testItem(Bug1Test.java:90)
```

After fix test results:

```
fines, quoted: 1.0 actual: 1.0  
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0,  
  
Results:  
  
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0  
  
-----  
BUILD SUCCESS  
-----
```

User Test after fix deployment:

When returning item:

```
above  
  DueDate: 29/10/2022  
  State: OVER_DUE  
  Fines: $1.00  
  
Overdue fine : $1.00  
  
Is item damaged? (Y/N):  
Overdue fine : $1.00  
  
Total fines : $1.00  
Return processing complete
```

When paying the fine:

```
Swipe patron card (press <enter> to cancel): Patron: 1
Name: John Smith
Email: dotcom
Phone: 1234
Fines Owed : $1.00
```

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:

The screenshot shows a GitHub commit interface for a repository named 'BugInew'. The commit is titled 'Bug 1 fix' and was made by 'tysau' 19 minutes ago. It shows 2 changed files with 0 additions and 2 deletions. The files are:

- A4/src/main/java/library/entities/Patron.java
- A4/src/main/java/library/returnItem/ReturnItemControl.java

The diff view shows the following changes:

```
@@ -79,7 +79,6 @@ public void takeOutLoan(Loan loan) {
79     public void dischargeLoan(Loan loan) {
80         long loanId = loan.getId();
81         if (currentLoans.containsKey(loanId)) {
82 -             finesOwing += loan.getFines();
83             currentLoans.remove(loanId);
84         }
85         else {
79     public void dischargeLoan(Loan loan) {
80         long loanId = loan.getId();
81         if (currentLoans.containsKey(loanId)) {
82             currentLoans.remove(loanId);
83         }
84         else {

@@ -87,7 +87,6 @@ public void dischargeLoan(boolean isDamaged) {
87     }
88     ui.display(String.format("\nTotal fines : $%.2f",
89         totalFines));
90 -     currentLoan.getPatron().incurFine(totalFines);
91     library.dischargeLoan(currentLoan, isDamaged);
87     }
88     ui.display(String.format("\nTotal fines : $%.2f",
89         totalFines));
90     library.dischargeLoan(currentLoan, isDamaged);
91 }
```