|  |  |
| --- | --- |
| **Bug Number** | 1 |
| **Description** | When a patron comes to pay an overdue fine, the amount they are charged is greater than the amount charged against the overdue loan. |
| **Pre-conditions** | Remove library.obj, build library from scratch |

# Step 1 : Replication

* This bug can be replicated by running ‘Bug 1 FAT.docx’. Below is an example with sample output (before the bug is resolved):

|  |  |  |
| --- | --- | --- |
| **Step** | **Expected Output/Result** | **Actual** |
| Return item that is 2 days overdue | Item returns  Charged $2.00 overdue | Text  Description automatically generated |
| Pay fine for previous overdue loan | Fine owed equals $6.00 | Text  Description automatically generated |

# Step 2 : Simplification

|  |  |  |
| --- | --- | --- |
| **Automated Test** | | Bug1Test |
| **Description** | | Given an overdue item, return it and check the overdue loan fee against the amount the patron owes. |
| **Required File** | | Bug1Test.java |
| **Set up** |  | |
| **Test** |  | |
| **Result** |  | |

# Step 3 : Tracing

|  |  |
| --- | --- |
| **Debugging Log** | Bug 1 using Bug1Test |
| **Initial Observation** | Bug1Test confirms the bug. The amount the patron is required to pay does not equal the amount over the fine incurred on the overdue loan. |

## Hypothesis 1:

|  |  |
| --- | --- |
| **Description** | We know from the FAT that the fine incurred is correct until the patron returns the overdue item. After the item is returned the fine is calculated incorrectly.  We can therefore begin our debugging trace within the ReturnItemControl class. |
| **Class** | ReturnItemControl |
| **Line / Variable** | currentLoan.getPatron().incurFine(totalFines) |
| **Prediction** | totalFines = $6.00 |
| **Test** | 1. Enter breakpoint in line 90: 2. Run Bug 1 FAT 3. Step the program through until break point is reached 4. Observe incurFine(totalFines) to see the amount incurred. |
| **Result** | totalFines = $2.00: |
| **Conclusion** | Hypothesis is rejected |

## Hypothesis 2:

|  |  |
| --- | --- |
| **Description** | From the bug trace above, we now know that the variable in: currentLoan.getPatron().incurFine(totalFines) is getting the correct value for the fine on the overdue loan.  Now, we can test the .incurFine method in Patron class to see if the issue is caused within the method itself. In particular, we can check the finesOwing variable to see if it is infected. |
| **Class** | Patron.java |
| **Line / Variable** | 101 and 102  Variable: finesOwing  public void incurFine(double fine) {  finesOwing += fine; } |
| **Prediction** | finesOwing is infected. |
| **Test** | 1. Enter breakpoint in line 102:      1. Run Bug 1 FAT 2. Step the program through until break point is reached:   Return the 2-day overdue item:  fine = 2.0 == sane.  finesOwing = 2.0 + 2.0 = 4.0.  finesOwing = **infected.** |
| **Result** | finesOwing = 4.0 |
| **Conclusion** | Hypothesis is accepted. |

## Hypothesis 3:

|  |  |
| --- | --- |
| **Description** | From the bug trace above, we now know that finesOwing is infected. The amount seems to be doubled. We also know during the Bug1Test, the fine the patron incurs is $6.00 meaning that the fine is actually incurred 3 times. We now need to find the source of this duplication.  We can debug trace public void dischargeLoan(Loan loan) to see what value finesOwing is updated to. |
| **Class** | Patron.java |
| **Line / Variable** | Variable: finesOwing  Line: 82 |
| **Prediction** | finesOwing = $6.00 |
| **Test** | 1. Enter breakpoint in line 82:      1. Run Bug 1 FAT 2. Step the program through until break point is reached:   Return the 2-day overdue item:  Text  Description automatically generated  finesOwing = 4.0 == **infected.**  loan.getFines() = 2.0: |
| **Result** | finesOwing = 6.0: |
| **Conclusion** | Hypothesis is accepted. finesOwing += loan.getFines() is a point of duplicating the amount in finesOwing. |

## Hypothesis 4:

|  |  |
| --- | --- |
| **Description** | We now know that line 82 unnecessarily adds 2.0 to the finesOwing variable. We can comment out / remove this line of code and continue our debugging as there is still a point of duplication. |
| **Line / Variable** | ReturnItemControl.java: 91  library.dischargeLoan(currentLoan, isDamaged)  Library.java: 220  patron.incurFine(overDueFine);  Patron.java: 16  Private double finesOwing |
| **Prediction** | library.dischargeLoan duplicates incurFine |
| **Test** | 1. Enter breakpoints on lines listed above. 2. Run Bug 1 FAT 3. Step the program through until break point is reached:   Return the 2-day overdue item:  Text  Description automatically generated  library.dischargeLoan invoked which invokes  finesOwing becomes 4.0 |
| **Result** | currentLoan.getPatron().incurFine(totalFines);  and library.dischargeLoan(currentLoan, isDamaged);  duplicate .incurFine method. |
| **Conclusion** | Hypothesis is accepted |

# Step 3 : Resolution

|  |  |
| --- | --- |
| **Bug Source** | From the debugging step above, we now know that the fine is incurred multiple times:   1. Once with the call to currentLoan.getPatron().incurFine(totalFines); 2. Again, with the call to: library.dischargeLoan(currentLoan, isDamaged); 3. Finally with the call to: finesOwing += loan.getFines() |
| **Solution** | Remove the call to:  currentLoan.getPatron().incurFine(totalFines);  And  finesOwing += loan.getFines()      The call tolibrary.dischargeLoan(currentLoan, isDamaged);  Will incur the fine as dischargeLoan method in Library class calls .incurFine method so the extra call to incurFine is not necessary. |
| **Confirmation via automated test** | Bug1Test:  Result: |
| **Confirmation via FAT** | Return the overdue Item. Overdue fine = $2.00:    Pay the fine. Patrons fines = $2.00 as expected: |