Backwoods Regional Library System

The Backwoods Regional Library System (BRLS) is a system that is intended to

# Bug Report:

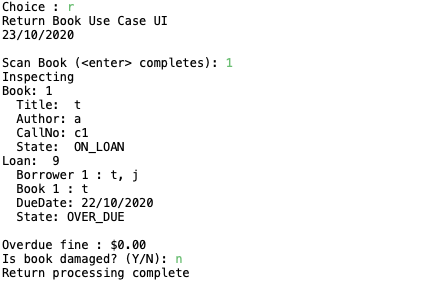
1. Incorrect calculation of fines: When a book becomes overdue by one day, no fine is imposed

2. Incorrect fine amount levied: When a fine is incurred, the amount of fine is half than the amount that is intended

3. Double fine amount charged: When a fine is issued, the value the system gives is correct, however when the patron tries to pay the fine they are charged double.

## Bug 1 – Incorrect Calculation of fines:

Replication:



Simplification:

H0 – problem in library.calculateOverDueFine

T0 – check if days overdue amount correct in method

After play – Patron is not issued a fine even though the loan is one day overdue

R0 – loan sane before run (one day overdue), amount of days overdue is returned as 0

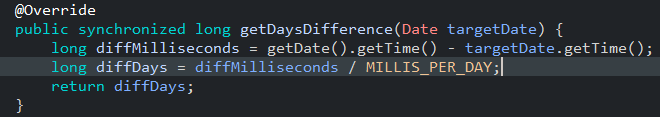
Hypothesis busted – bug must be in days overdue calculation (Calendar.getDaysDifference)

H1 – Calendar.getDaysDifference not returning correct day

T1 – check if amount of days difference between current date and loan date is correct

R1: loan sane before run (one day overdue), difference in milliseconds is correct yet difference in days in incorrect.





Hypothesis confirmed – bug is in Calendar.getDaysDifference

Simplification test:

test/library/entities/IncorrectCalculationOfFinesTest.java

H2 – diffDays calculation is incorrect due to a rounding issue

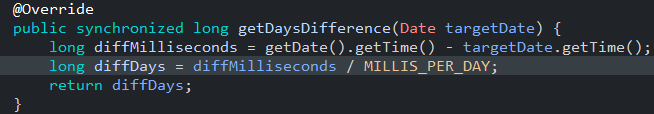
T2 – ensure that all values are correct going into the diffDays calculation

R2 – the milliseconds difference is correct yet the days difference is incorrect as the value is rounded down

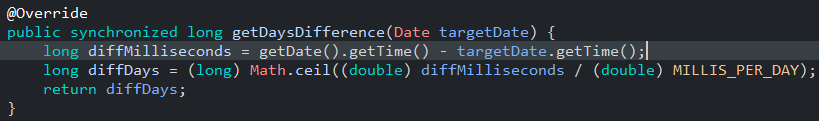
Hypothesis confirmed – The calculation of diffDays is incorrect due to a rounding error in the calculation.

H3 – Bug is in line 66 – should make diffDays round up

Before:

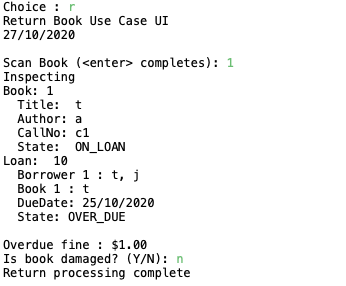


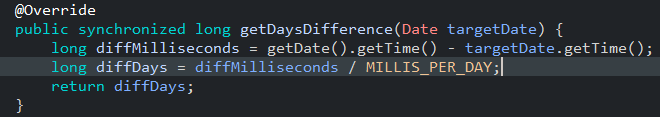
After:

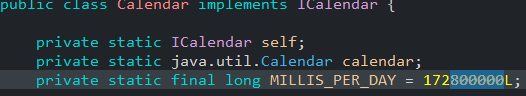
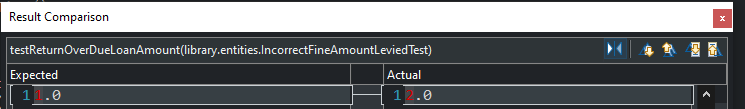
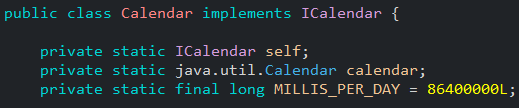


Hypothesis confirmed – The bug was caused by a rounding issue and correcting this issue solves the problem.

## Bug 2 – Incorrect **Fine Amount Levied**:

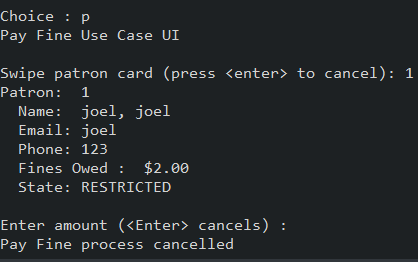
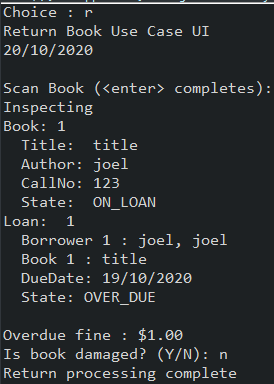
1. When a fine is incurred, the amount of the fine reported is half the amount intended
2. Replication:
3. 
4. Simplification:
5. H0 – problem in library.calculateOverDueFine
6. T0 – check if days overdue amount correct in method
7. After play – Patron is not issued a fine even though the loan is one day overdue
8. R0 – loan sane before run (one day overdue), amount of days overdue is returned as half of the expected amount
9. Hypothesis busted – bug must be in days overdue calculation (Calendar.getDaysDifference)
10. H1 – Calendar.getDaysDifference not returning correct day
11. T1 – check if amount of days difference between current date and loan date is correct
12. R1: loan sane before run (one day overdue), difference in milliseconds is correct yet difference in days in incorrect.
13. 



1. Hypothesis confirmed – bug is in Calendar.getDaysDifference
2. H2 – diffDays calculation is incorrect due to incorrect values
3. T2 – ensure that all values are correct going into the diffDays calculation
4. R2 – the milliseconds difference is correct yet the MILLIS\_PER\_DAY variable is double what it should be: it should be 86400000 instead of 172800000
5. Hypothesis confirmed – The calculation of diffDays is incorrect due to the incorrect MILLIS\_PER\_DAY variable.
6. Simplification test:
7. test/library/entities/IncorrectFineAmountLeviedTest.java
8. H3 – Bug is in line 10 – should make MILLIS\_PER\_DAY 86400000l
9. Before:
10. 
11. 
12. After:
13. 
14. 
    1. Hypothesis confirmed – The bug was caused by the wrong MILLIS\_PER\_DAY value and correcting it fixed the bug.

## Bug **3** – **Double Fine Amount Charged**

Replication:



Simplification:

H0 – problem in patron.incurFine as this method is what gives the patron the fine

T0 – check if method is giving the patron the correct fine

R0 – loan sane before run (one day overdue), incurred fine amount sane before run, method gives patron the correct amount

Hypothesis busted – If the patron.incurFine method is giving the patron the correct fine amount the bug must be elsewhere

H1 – patron.incurFine is being called twice and that is why the patron is being charged double the amount

T1 – The easiest way to find where the patron.incurFine method is being called in the system is through the linux command “grep -r incurFine”. This command returns all instances of the string “incurFine” and the file path that they are in.

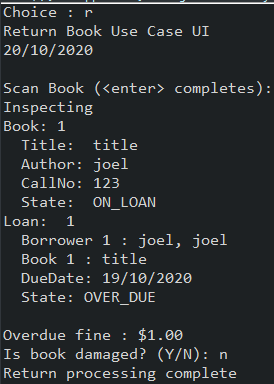
R1 – Searching through the code revealed that patron.incurFine is in fact called twice: once in ReturnBookControl at line 53 and once in Library.dischargeLoan at line 216. Assuming that the loan is overdue when the patron returns it, both of these functions are called by ReturnBookUI and therefore the patron is given the fine amount twice.

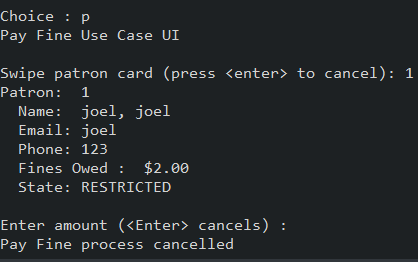
Hypothesis confirmed – The method patron.incurFine is being called twice, causing this bug to occur.

1. Simplification test:
2. Requires UI component – cannot be tested automatically

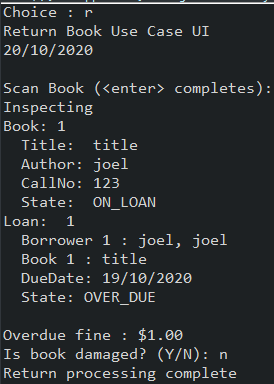
H1 – remove the patron.incurFine method call from ReturnBookControl will fix the bug

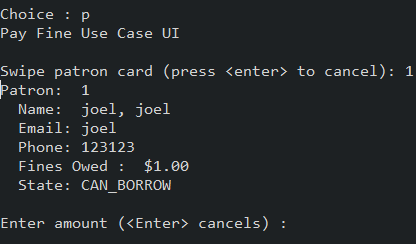
Before removing method call:





After removing method call:





Hypothesis confirmed – bug was fixed after the second method call was removed from ReturnBookControl.bookScanned().