# **ITC205 Assessment 4: Debugging**

**Name:** Luke Johnson

**Student Number:** 11598797

**Subject:** ITC205

**Github URL:** https://github.com/lukexjohnson/DebuggingA4

**1.Replication.**

1. For each of the reported bugs, generate a test case in UAT format that will reliably reproduce the buggy behaviour.
2. Commit the test case documentation to version control.

**2.Simplification.**

1. For each of the reported bugs, produce a simplified and automated test that reliably reproduces the bug.
2. Commit the test code demonstrating the bug to version control.
3. Record output from your test, demonstrating the buggy behaviour.

**3.Tracing.**

1. For each of the reported bugs, trace back through the program’s dependences from the point of failure to find the origin of the bug. Use a sequence of hypotheses and tests to support this search, and document this sequence in a debugging log.
2. For each of the reported bugs, demonstrate that before the identified origin, variable values are ‘sane’ and that after the origin, at least one value is ‘infected’.
3. Record screen shots demonstrating the origin of the bug.

**4.Resolution.**

1. For each of the reported bugs, correct the bug.
2. Commit the corrected code to version control.
3. Record output listings from your automated test demonstrating that the buggy behaviour has been eliminated.
4. Carry out the user test case with the specified data and show that the buggy behaviour has been eliminated.

# **Bug 1: All service charges are reported as $0.00 when checking out**

**Hypothesis # 1.1:**

Values are producing bad data/not stored correctly.

**Class & Method:**

* **Class**: RecordServiceCTL
* **Method**: serviceDetailsEntered()
* **Breakpoint set at:** line 52

**Prediction:**

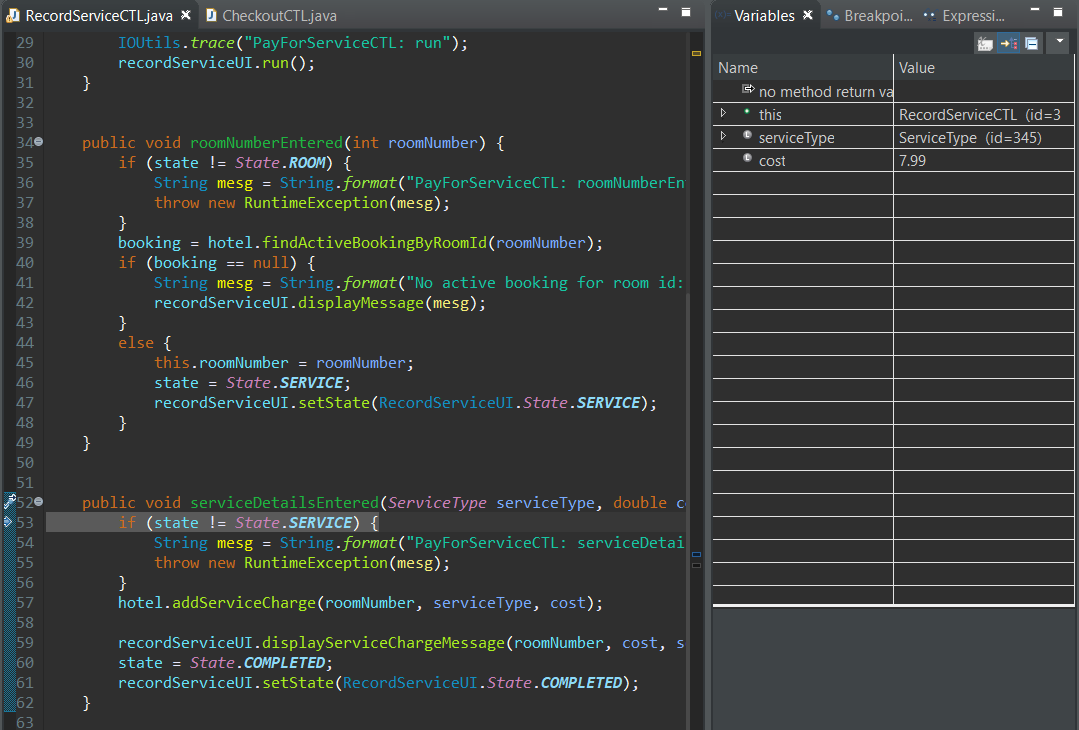
Charges accrued to account are not accurately captured/stored.

**Test:**

Check if ‘cost’ value is being stored correctly

**Result:**

Debug shows that cost variable is accurately storing the value as 7.99



**Hypothesis # 1.2:**

**Prediction:**

addServiceCharge method is not storing data for cost.

**Class & Method:**

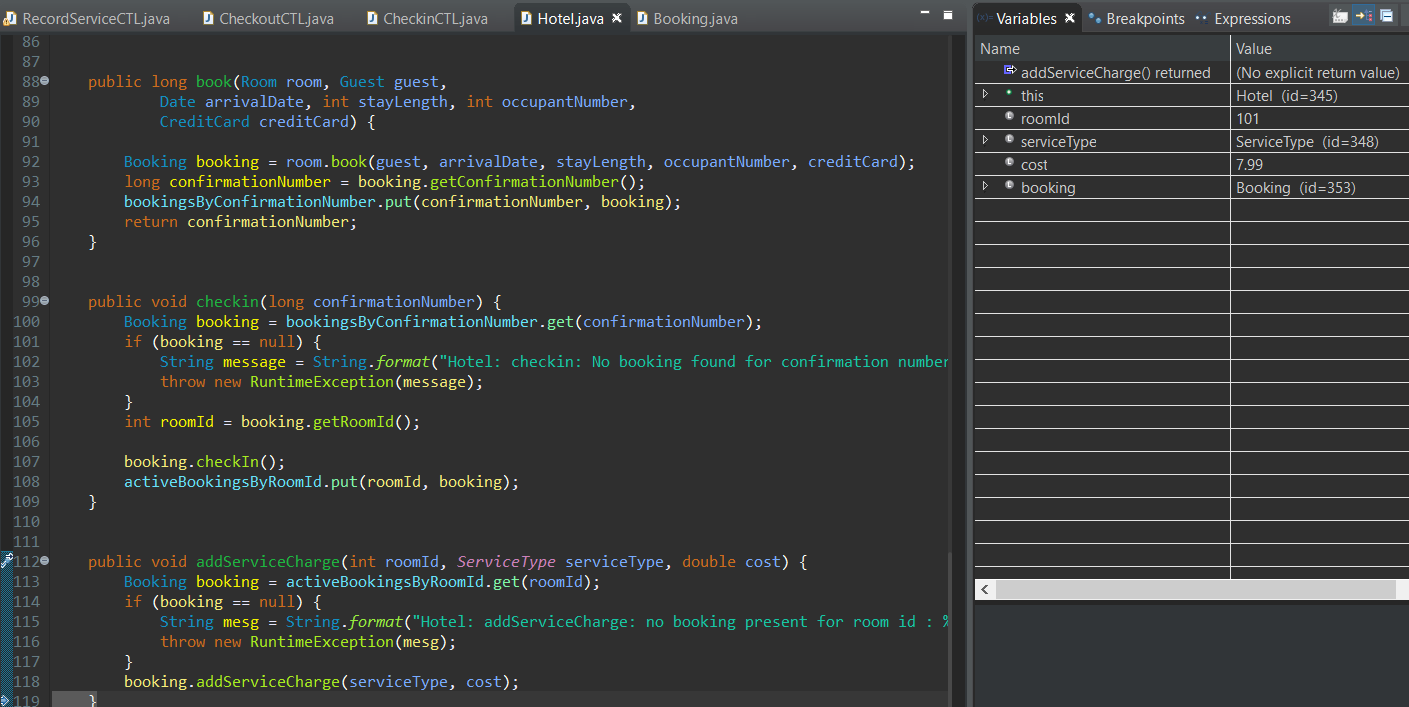
* **Class:** Hotel
* **Method**: addServiceCharge()
* **Breakpoint set at:** line 112

**Test:**

Confirm cost data is being passed to addServiceCharge

**Result:**

Value is correctly being passed to addServiceCharge



**Hypothesis # 1.3:**

**Prediction:**

The incorrect spelling of ‘cost’ in Booking classes method addServiceCharge set as ‘cosst’ causes the value in ‘cost’ variable not to change.

Booking class method addServiceCharge has a spelling mistake in its parameter ‘cosst’ on line 141 which causes the value in ‘cost’ not to change. Spelling needs to be amended in parameter from ‘cosst’ to ‘cost’

**Class & Method:**

* **Class:** Booking
* **Method:** addServiceCharge & checkOut
* **Breakpoint 1:** line 141
* **Breakpoint 2:** line 147

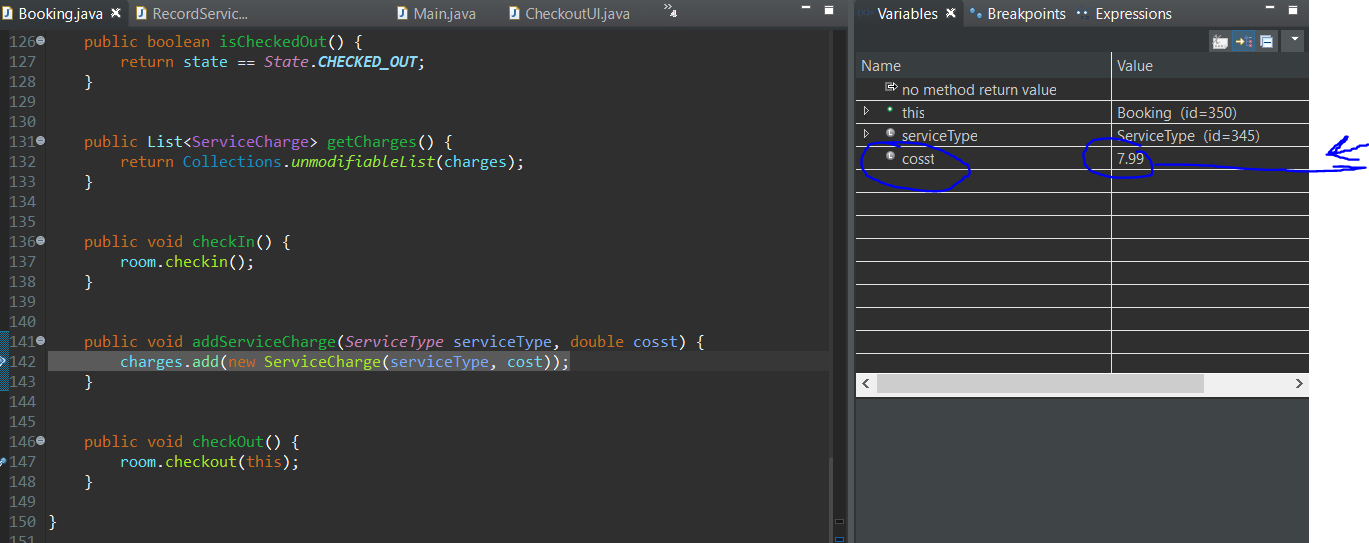
**Test:**

Check that cosst is changing the value in cost when it receives an argument.

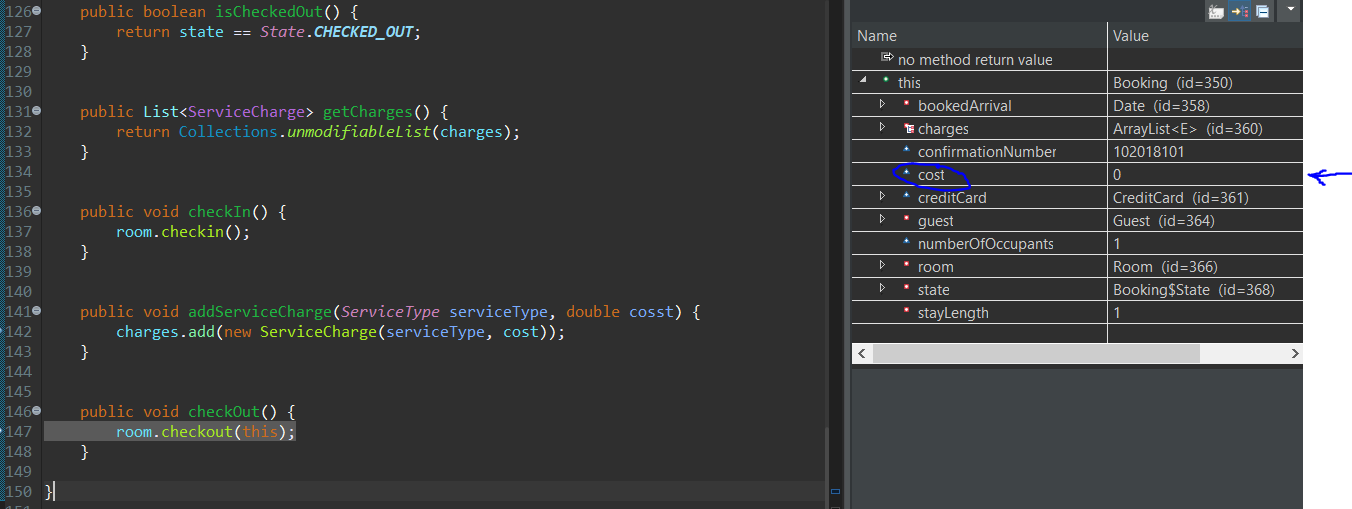
**Result:**

Cost value is not updating due to spelling error in line 141 ‘cosst’ – needs to be updated to cost

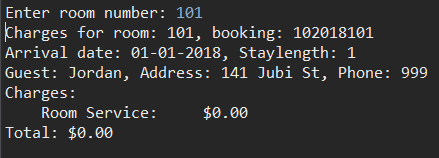
**Screenshot# 1** – ‘cosst’ [breakpoint 141 - addServiceCharge]



**Screenshot# 2** – ‘cost’ [breakpoint 147 - checkOut]

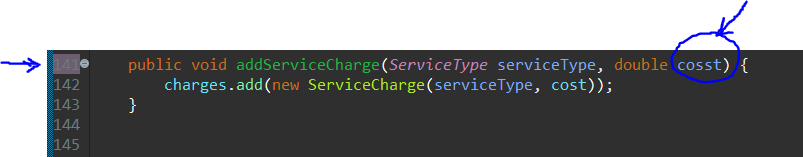


Screenshot# 3 – Program output.



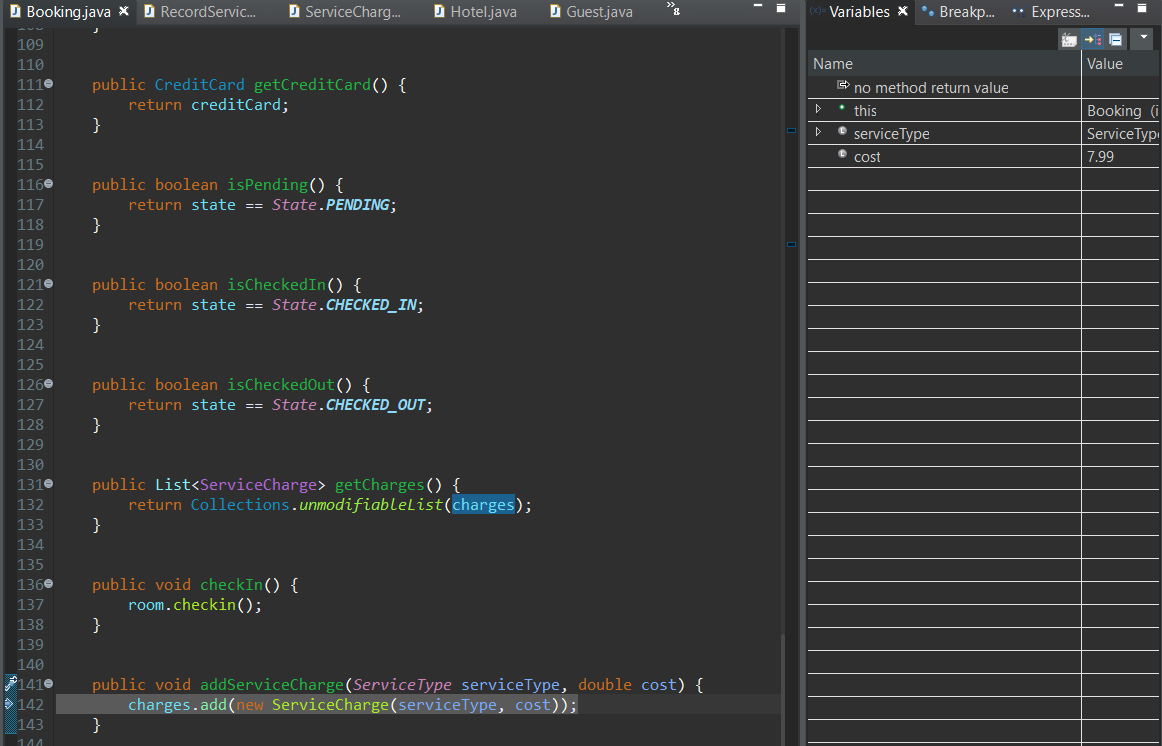
**Resolution**

A spelling error on line 141 in addServiceCharge parameters was found – ‘cosst’ has now been updated to ‘cost’. After amendments the correct value for cost is now showing.

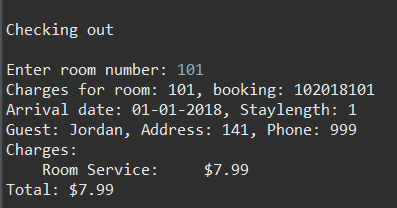


**Screenshot# 1 –** [breakpoint 147 - addServiceCharge]

Displays cost receiving the entered value of 7.99



**Screenshot# 2 – program output after amendment – now showing correct value.**



# **Bug 2: It is possible to charge a room for service after the guest has checked out**

**Hypothesis # 2.1:**

**Prediction:**

Room is not being checked out correctly in booking class checked out method

**Class & Method:**

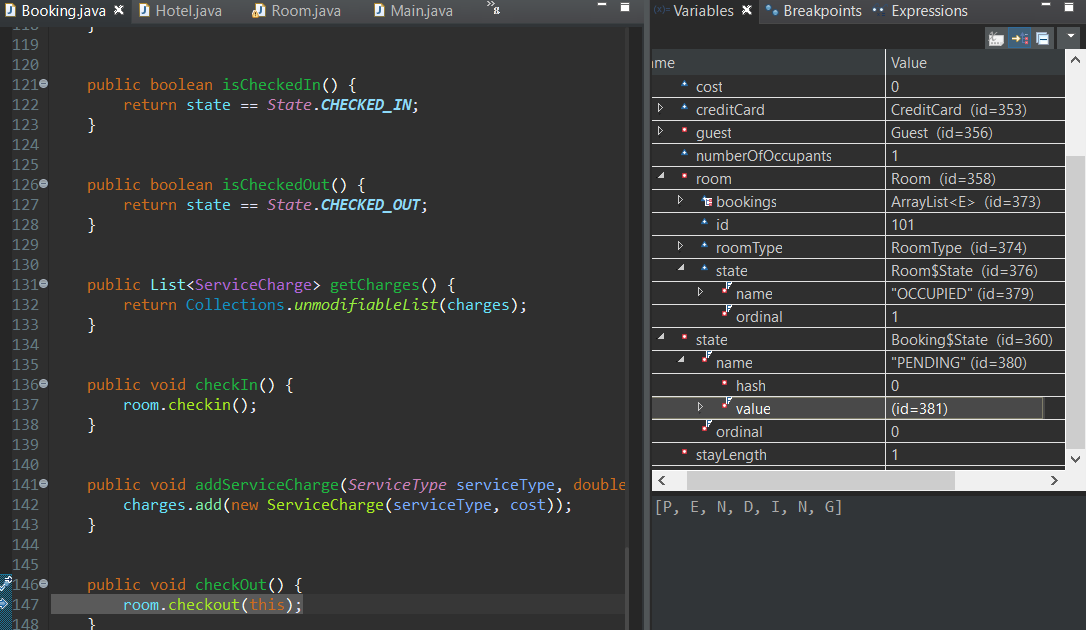
* **Class:** Booking
* **Method:** CheckOut
* **Breakpoint:** 146 Booking

**Test:**

Check state is correctly checking out

**Result:**

State is reliably changing from OCCUPIED to PENDING



**Hypothesis # 2.2:**

**Prediction:**

Checkout in Hotel class is not correctly checking out

**Class & Method:**

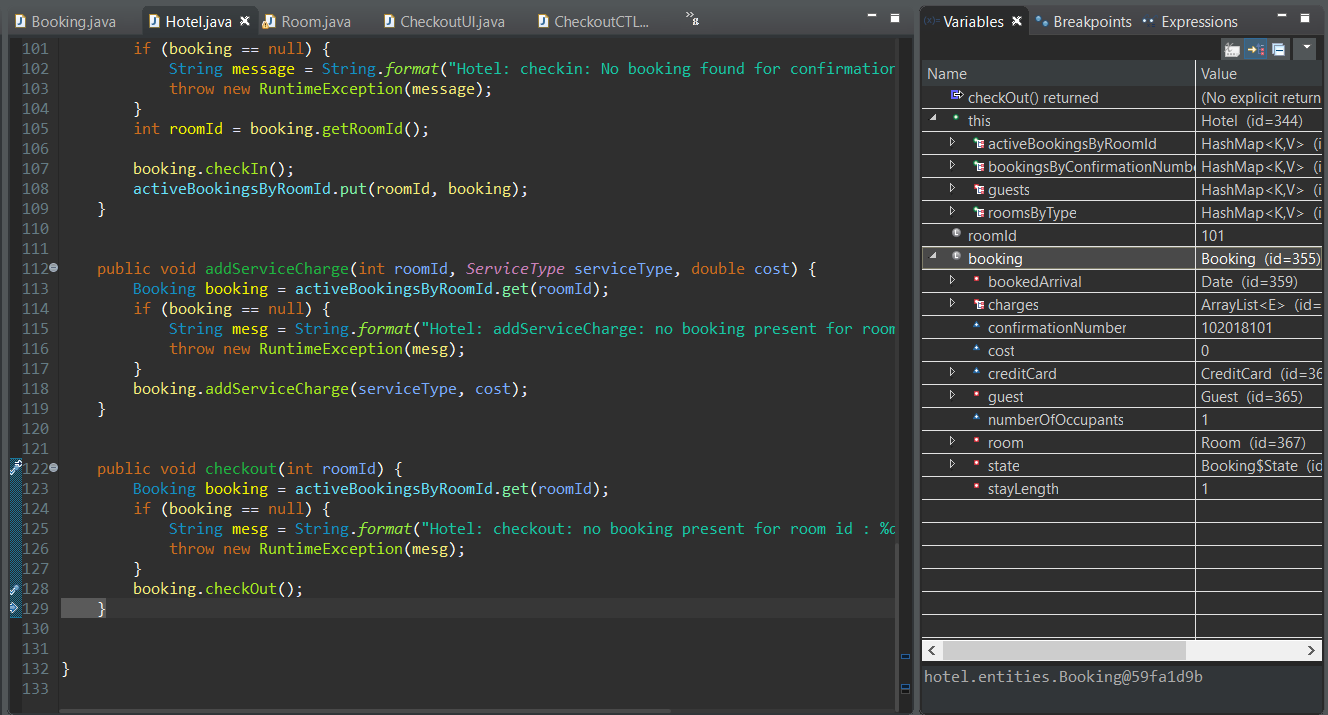
* **Class:** Hotel
* **Method:** CheckOut
* **Breakpoint:** line 122

**Test:**

Check that checkout method is being called correctly

**Result:**

Method was called correctly and displayed all values associated with checking out.



**Hypothesis # 2.3:**

**Prediction:**

Room ID is not being removed from active bookings when checking out

**Class & Method:**

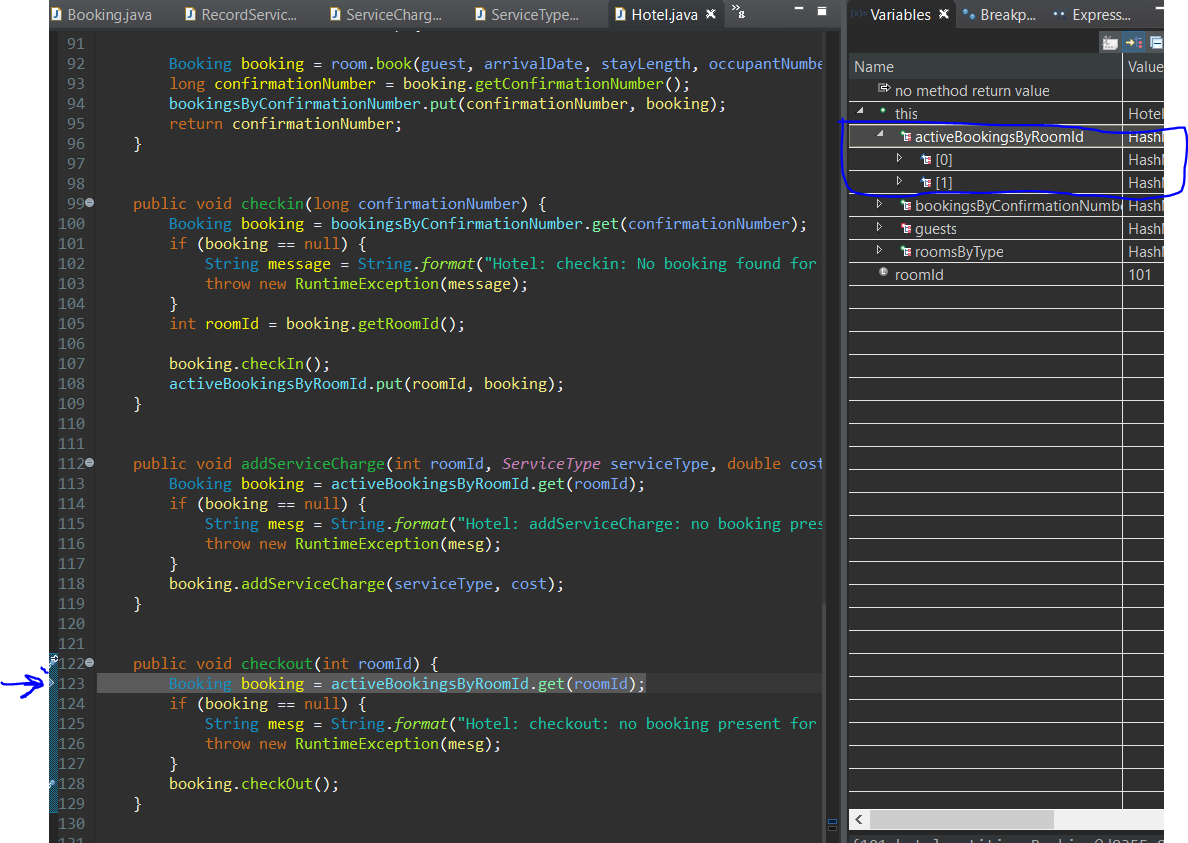
* **Class:** Hotel
* **Method:** Checkout
* **Breakpoint:** line 123 & line 129

**Test:**

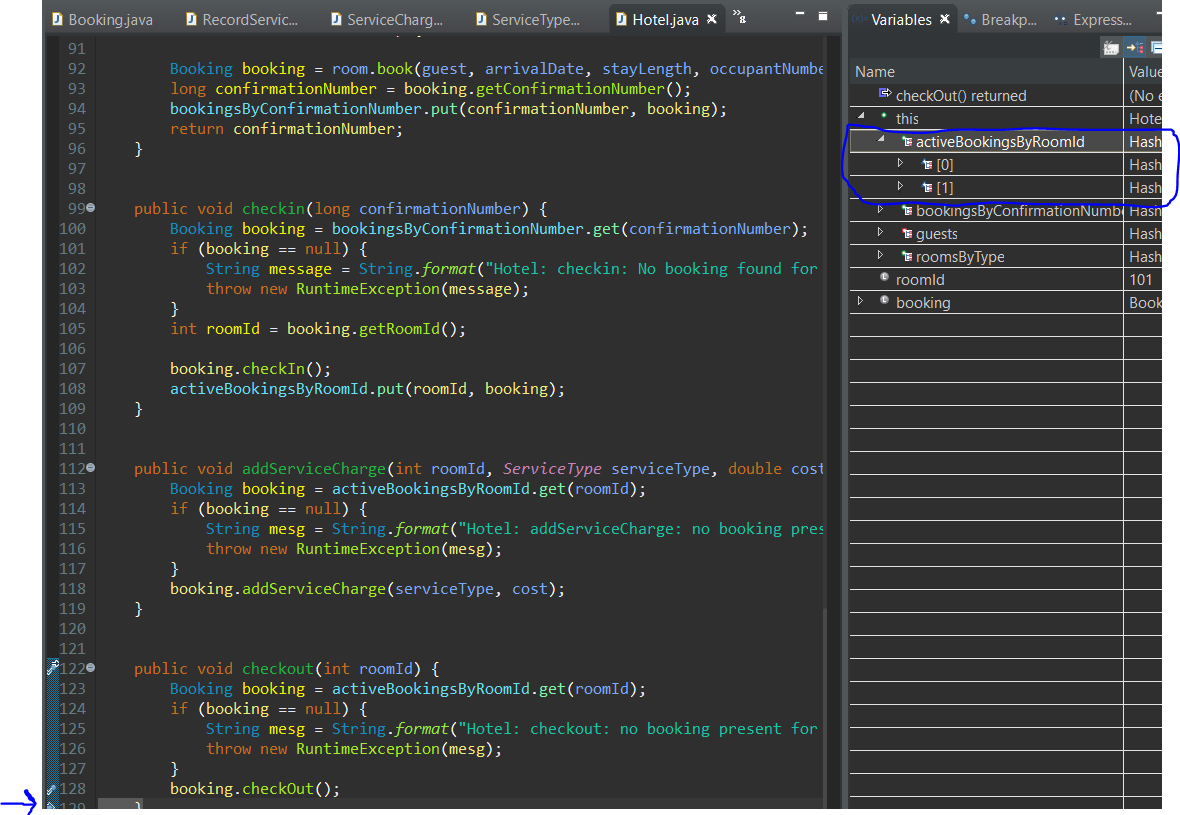
Test to see if room ID is being removed from active bookings

**Result:**

Screenshot# 1: Checkout starts with 2 active bookings by room ID



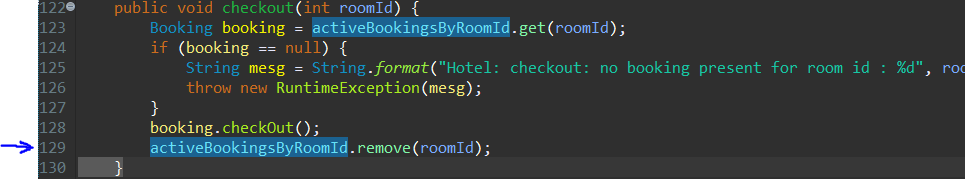
Screenshot# 2: Shows that 2 items are still displayed in active bookings after checkout



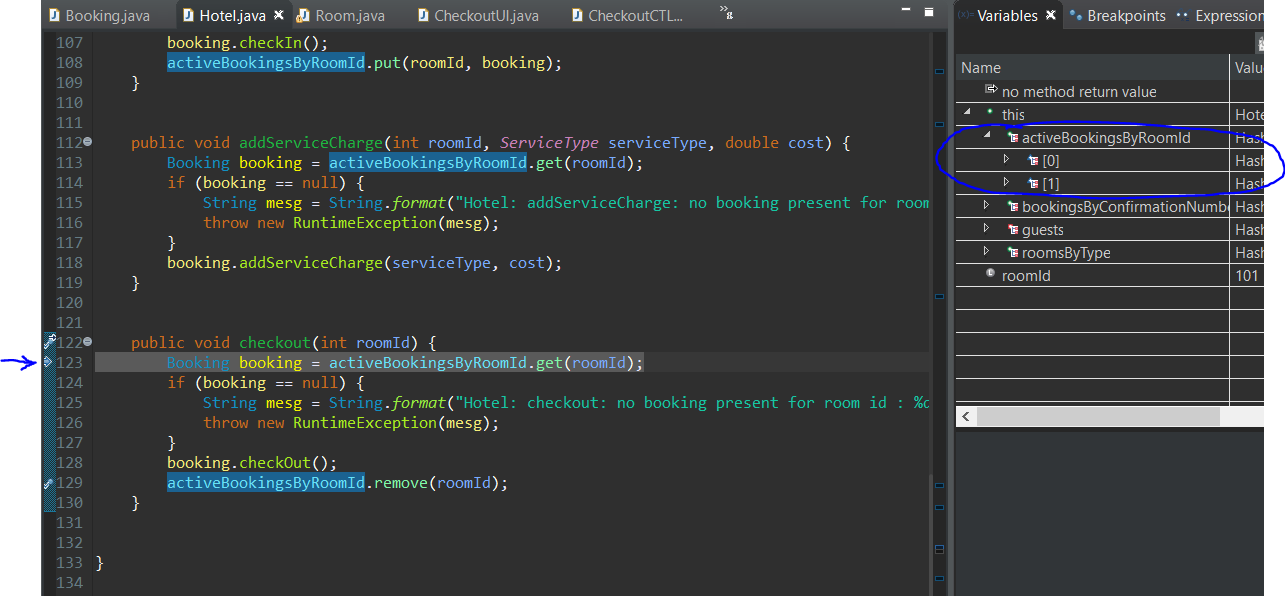
**Test confirmed – Room ID is not removed from active booking’s after checkout**

**Resolution**

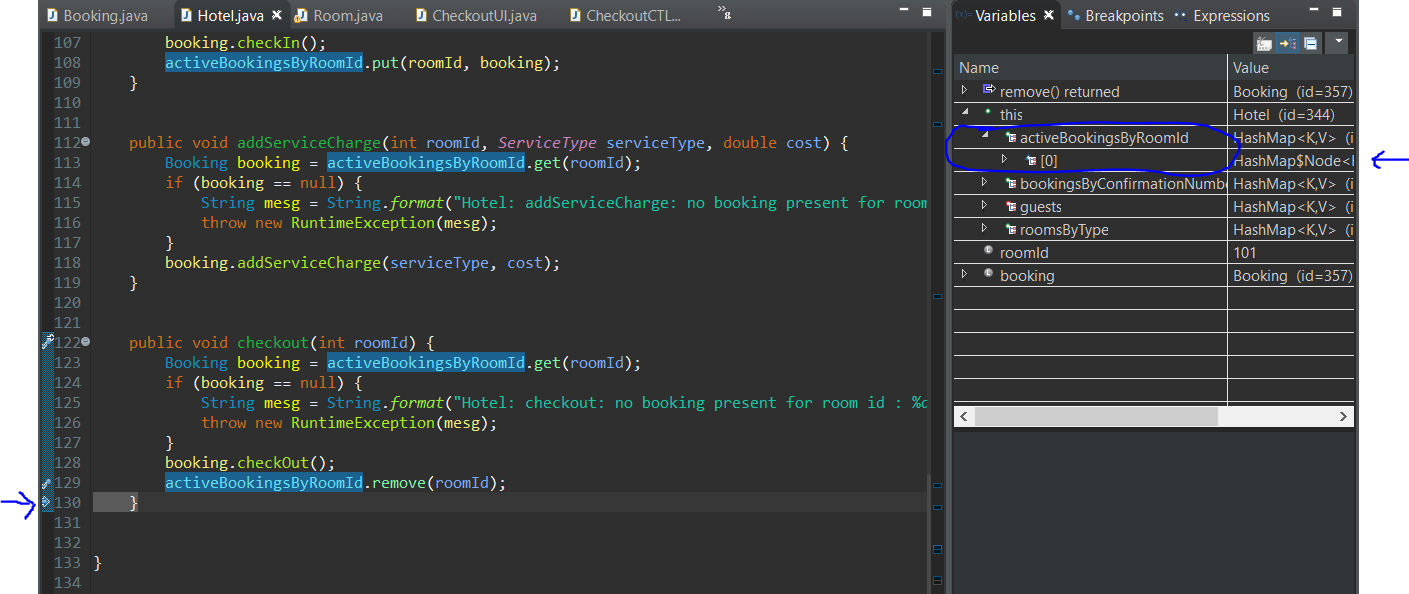
By adding in the **line 129,** it will remove the room id from active bookings. All break points have been left in the same place for second test.



Screenshot# 1 [Resolution]: enters checkout method with 2 items in active bookings by room id



Screenshot# 2 [Resolution]: shows that the 2nd item has been removed from active bookings successfully after checkout



**Output after line 129 added to remove room id from active booking by room Ids**

