Name: Antonio Scalfaro  
Date: 2/27/2024  
Week: 6 - Black-box Unit Test the Reservation Class of a Small Bed & Breakfast Reservation System

Embed here a copy of your complete Java unit test source code (e.g., TestReservation.java):

package assignment\_3;

import java.text.SimpleDateFormat;

import java.util.\*;

public class TestReservation {

private static String datePattern = "MMM dd, yyyy";

private static SimpleDateFormat sdf = new SimpleDateFormat(datePattern);

public static void main(String[] args){

// TODO Auto-generated method stub

testConstructorAndGetters();

testSettersAndGetters();

testReservationBillAmount();

testReservationNumberOfDays();

}

public static void testConstructorAndGetters() {

System.out.println();

System.out.println("Testing Constructor and Getters");

System.out.println("----------------------------------------");

Reservation r = new Reservation(1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022");

Reservation r2 = new Reservation(7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022");

Reservation r3 = new Reservation(9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022");

Assert.assertNotEqualsUUID(r.getReservationID(), r2.getReservationID());

Assert.assertEqualsDate(r.getReservationDate(), new Date());

Assert.assertNotEqualsString(r.getReservationStartDate(), r3.getReservationStartDate());

Assert.assertEqualsInt(r.getGuestID(), 1);

Assert.assertNotEqualsInt(r.getGuestID(), r2.getGuestID());

Assert.assertEqualsString(r.getRoomType(), r2.getRoomType());

Assert.assertEqualsString(r.getReservationStartDate(), r2.getReservationStartDate());

Assert.assertEqualsString(r.getReservationEndDate(), r2.getReservationEndDate());

}

public static void testReservationBillAmount() {

System.out.println();

System.out.println("Testing Calculate Reservation Bill Amount");

System.out.println("----------------------------------------");

Reservation r = new Reservation(1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022");

Reservation r2 = new Reservation(7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022");

Reservation r3 = new Reservation(9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022");

try {

Assert.assertNotEqualsDouble(r.calculateReservationBillAmount(), r2.calculateReservationBillAmount());

Assert.assertNotEqualsDouble(r2.calculateReservationBillAmount(), r3.calculateReservationBillAmount());

Assert.assertEqualsDouble(r.calculateReservationBillAmount(), 600.00);

Assert.assertEqualsDouble(r2.calculateReservationBillAmount(), 525.00);

Assert.assertEqualsDouble(r3.calculateReservationBillAmount(), 375.00);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public static void testSettersAndGetters() {

System.out.println();

System.out.println("Testing Setters and Getters");

System.out.println("----------------------------------------");

Reservation r = new Reservation(1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022");

r.setCustomerID(2);

Assert.assertEqualsInt(r.getGuestID(), 2);

r.setRoom("RoomWView");

Assert.assertEqualsString(r.getRoomType(), "RoomWView");

r.setReservationStartDate("Jun 17, 2022");

Assert.assertEqualsString(r.getReservationStartDate(), "Jun 17, 2022");

r.setReservationEndDate("Jun 20, 2022");

Assert.assertEqualsString(r.getReservationEndDate(), r.getReservationEndDate());

}

public static void testReservationNumberOfDays() {

System.out.println();

System.out.println("Testing Calculate Reservation Number of Days");

System.out.println("----------------------------------------");

Reservation r = new Reservation(1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022");

Reservation r2 = new Reservation(7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022");

Reservation r3 = new Reservation(9, "NormalRoom", "Jun 16, 2022", "Jun 24, 2022");

try {

Assert.assertEqualsLong(r.calculateReservationNumberOfDays(), 3);

Assert.assertEqualsLong(r2.calculateReservationNumberOfDays(), r.calculateReservationNumberOfDays());

Assert.assertEqualsLong(r3.calculateReservationNumberOfDays(), 8);

Assert.assertNotEqualsLong(r.calculateReservationNumberOfDays(), r3.calculateReservationNumberOfDays());

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------

Rubric Criteria:  
Create black-box test cases to test the constructor and the getters methods of the Reservation class 8%  
Your Response:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Constructor and Getters** | | | | |
| **Test #** | **Selected Inputs**  **guestID,**  **roomType,**  **reservationStartDate,**  **reservationEndDate** | **Expected Result** | **Actual Result** | **Pass | Fail** |
| **1** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Objects 1 and 2 UUID not equal to each other | Actual: 9ce3061c-d6ef-4f22-297d-0ef2d9aca26f does not match Expected: 098c6d9d-580e-aaab-be0b-683ce1b724bc PASS | PASS |
| **2** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 getReservationDate equals new Date() | Actual: Wed Feb 21 13:22:50 EST 2024 does match Expected: Wed Feb 21 13:22:50 EST 2024 PASS | PASS |
| **3** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 and 3 reservation start dates are not equal | Actual: Jun 16, 2022 does not match Expected: Jun 17, 2022 PASS | PASS |
| **4** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 getGuestID is equal to 1 | Actual: 1 does match Expected: 1 PASS | PASS |
| **5** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 and 2 getGuestID are not equal | Actual: 1 does not match Expected: 7 PASS | PASS |
| **6** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 and 2 getRoomType are equal | Actual: RoomWBath does match Expected: RoomWBath PASS | PASS |
| **7** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 and 2 getReservationStartDate are equal | Actual: Jun 16, 2022 does match Expected: Jun 16, 2022 PASS | PASS |
| **8** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 17, 2022", "Jun 20, 2022" | Compare Reservation Object 1 and 2 getReservationEndDate are equal | Actual: Jun 19, 2022 does match Expected: Jun 19, 2022 PASS | PASS |

Rubric Criteria:  
Execute, using w6.jar, unit tests for the constructor and the getters method of the Reservation class. Document the unit tests code and results via screenshots 10%  
Your Response:

A screenshot of a computer screen

Description automatically generated

Rubric Criteria:  
Explain approach, steps, and rationale of the test cases and unit tests of testing the constructor and the getters method of the Reservation class 5%  
Your Response:

The approach I took to testing the constructor and the getters methods was to create three instances of Reservation objects two of which have the same information, save the guestID which is unique, and one instance which has different information. I ensure to check all available getter methods in the Reservation class and ensure that the information comes back as intended (i.e the UUID of the nearly identical objects are not equal to each other, but their reservation start and end dates do match).

-----------------------------------------------------------------------------------------------------------------------------------

Rubric Criteria:  
Create black-box test cases to test the setters and the getters methods of the Reservation class 8%  
Your Response:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test setters and getters** | | | | |
| **Test #** | **Selected Inputs**  **guestID,**  **roomType,**  **reservationStartDate,**  **reservationEndDate**  **selected setter method()** | **Expected Results** | **Actual Results** | **Pass | Fail** |
| **1** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  setCustomerID(2) | Change Reservation object guestID from 1 to 2. | Actual: 2 does match Expected: 2 PASS | PASS |
| **2** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  setRoom(“RoomWView”) | Change Reservation object roomType from RoomWBath to RoomWView | Actual: RoomWView does match Expected: RoomWView PASS | PASS |
| **3** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  setReservationStartDate("Jun 17, 2022") | Change Reservation start date from Jun 16, 2022 to Jun 17, 2022 | Actual: Jun 17, 2022 does match Expected: Jun 17, 2022 PASS | PASS |
| **4** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  setReservationEndDate("Jun 20, 2022") | Change Reservation end date from Jun 19, 2022 to Jun 20, 2022 | Actual: Jun 20, 2022 does match Expected: Jun 20, 2022 PASS | PASS |

Rubric Criteria:  
Execute, using w6.jar, unit tests for the setters and the getters method of the Reservation class. Document the unit tests code and results via screenshots 10%  
Your Response:

A screenshot of a computer screen

Description automatically generated

Rubric Criteria:  
Explain approach, steps, and rationale of the test cases and unit tests of testing the setters and the getters method of the Reservation class 5%  
Your Response:

The approach to the setters and getters testing was similar to the constructor and getters method. Its aim is to test all the setter methods and ensure that the information is correctly stored back into the object. I start with a single Reservation object and change the initial information one by one with the setter methods, then test that they changed by using the getter methods to retrieve the data. I do not call every single getter method because that is for the first set of test cases whereas this is to ensure that the changed information is correctly stored by the setter methods and the getter methods just provide the proof.

-----------------------------------------------------------------------------------------------------------------------------------

Rubric Criteria:  
Create black-box test cases to test the calculateReservationNumberOfDays() method of the Reservation class 8%  
Your Response:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test calculateReservationNumberOfDays()** | | | | |
| **Test #** | **Selected Inputs**  **guestID,**  **roomType,**  **reservationStartDate,**  **reservationEndDate** | **Expected Results** | **Actual Results** | **Pass | Fail** |
| **1** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 24, 2022" | Call calculation of number of days on first Reservation object and ensure the result equal to 3 | Actual: 3 does match Expected: 3 PASS | PASS |
| **2** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 24, 2022" | Call calculation of number of days on the first and second Reservation objects and ensure that they are equal to each other. | Actual: 3 does match Expected: 3 PASS | PASS |
| **3** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 24, 2022" | Call calculate number of days on the third Reservation object and ensure it is equal to 8 | Actual: 8 does match Expected: 8 PASS | PASS |
| **4** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 24, 2022" | Call calculate number of days on the first and third Reservation objects and ensure they are not equal to each other | Actual: 3 does not match Expected: 8 PASS | PASS |

Rubric Criteria:  
Execute, using w6.jar, unit tests for the calculateReservationNumberOfDays() method of the Reservation class. Document the unit tests code and results via screenshots 10%  
Your Response:

A screenshot of a computer screen

Description automatically generated

Rubric Criteria:  
Explain approach, steps, and rationale of the test cases and unit tests of testing the calculateReservationNumberOfDays() method of the Reservation class 5%  
Your Response:

My approach to testing the calculateReservationNumberOfDays() method was to make three objects, two of which had the same reservation start and end dates and a third that had a different start and end date and a longer period of time. I check first that the function returns the correct amount of days with the first object and then check that the result of calling it on object one and two are equal. I then check that the third object returns the correct days and then check that the result of calling the method on object one and three are not equal.

-----------------------------------------------------------------------------------------------------------------------------------

Rubric Criteria:  
Create black-box test cases to test the calculateReservationBillAmount() method of the Reservation class 8%  
Your Response:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test calculateReservationBillAmount()** | | | | |
| **Test #** | **Selected Inputs**  **guestID,**  **roomType,**  **reservationStartDate,**  **reservationEndDate** | **Expected Results** | **Actual Results** | **Pass | Fail** |
| **1** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022" | Calculate the bill amount of Reservation object one and two and ensure they are not equal | Actual: 600.0 does not match Expected: 525.0 PASS | PASS |
| **2** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022" | Calculate the bill amount of Reservation object two and three and ensure they are not equal | Actual: 525.0 does not match Expected: 375.0 PASS | PASS |
| **3** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022" | Calculate the bill amount of Reservation object one is equal to 600.00 | Actual: 600.0 does match Expected: 600.0 PASS | PASS |
| **4** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022" | Calculate the bill amount of Reservation object two is equal to 525.00 | Actual: 525.0 does match Expected: 525.0 PASS | PASS |
| **5** | 1, "RoomWBath", "Jun 16, 2022", "Jun 19, 2022"  &&  7, "RoomWView", "Jun 16, 2022", "Jun 19, 2022"  &&  9, "NormalRoom", "Jun 16, 2022", "Jun 19, 2022" | Calculate the bill amount of Reservation object three and ensure it is equal to 375.00 | Actual: 375.0 does match Expected: 375.0 PASS | PASS |

Rubric Criteria:  
Execute, using w6.jar, unit tests for the calculateReservationBillAmount() method of the Reservation class. Document the unit tests code and results via screenshots 10%  
Your Response:

A screenshot of a computer program

Description automatically generated

Rubric Criteria:  
Explain approach, steps, and rationale of the test cases and unit tests of testing the calculateReservationBillAmount() method of the Reservation class 5%  
Your Response:

My approach to testing the calculateReservationBillAmount() method was to create three Reservation objects with the same stay period but had varied room types. This would ensure that all the different room types and their cost would be tested. I test that none of the objects bill total equal each other first. Then I move to test each Reservation object returns the correct amount when the method is called.

-----------------------------------------------------------------------------------------------------------------------------------

Rubric Criteria:  
Reflect on the learning experience and lessons learned 8%  
Your Response:

This was a fairly straightforward project to work on, I breezed through most of the coding portions of the project without issue. Most of the lessons learned come from choosing which tests to run and if my test cases were sufficiently testing the Reservation class. I have a decent amount of experience with running test files on classes so I felt comfortable along the way. I did struggle with the w6.jar file and spent many hours attempting to run the program from the command line which I eventually was unsuccessful in doing. I had the jar file imported correctly but still need to investigate further why the command line running of the program failed. The overall lesson here is that often times the coding will be the easiest portion of the work, while the crafting of test cases and the complication of the runtime environment will often cause the biggest headaches.