CSSE376 – Lab 3 Unit testing – Caio Balthazar – CM 1478

**[Question 1 (1 pt)] In what project is the source code found? In what project is the test code found?**

In the Expedia project. In the Expedia Test project.

**[Question 2 (2 pts)] What classes exist in the project?**

Booking, Car, Flight, Hotel and User (There are also test classes for each of these in the ExpediaTest project).

**[Question 3 (4 pts)] Explain, in your own words, the functionality that is supported by the Flight class.**

It holds the data and procedures necessary for calculating the base price of a given flight. This calculation method is required by the interface Booking.

**[Question 4 (2 pts)] What are the test classes in the project?**

BookingTest, CarTest, FlightTest, HotelTest and UserTest.

**[Question 5 (4 pts)] Identify the various test methods in the UserTest class.**

TestThatUserInitializes, TestThatUserHasZeroFrequentFlierMilesOnInit, TestThatUserCanBookEverything, TestThatUserHasFrequentFlierMilesAfterBooking, TestThatUserCanBookAFlight, TestThatUserCanBookAHotelAndACar and TestThatUserHasCorrectNumberOfFrequentFlyerMilesAfterOneFlight.

**[Question 6 (4 pts)] Identify at least three different functions that are supported by the Assert class. (Hint: type in \Assert" followed by a period in one of the test classes in the \ExpediaTest" project and use the documentation.)**

AreEqual – Verifies that two values are equal.

Are Same – Asserts that two objects refer to the same object.

DoesNotThrow – Verifies that a delegate does not throw an exception.

**[Question 7 (4 pts)] Please explain the functions that you identified briefly.**

AreEqual – Verifies that two values are equal. In other words, checks for the equality in the values of the two objects, ignoring the fact that they might be different object references.

Are Same – Asserts that two objects refer to the same object. In other words, verifies that these objects point to the same place and/or have the same native object id, thus they are the same object.

DoesNotThrow – Verifies that a delegate does not throw an exception. In other words, verifies that the execution of a determined portion of code (in this case the testing method) does not throw this exception while running.

**[Question 8 (4 pts)] You may have noticed that there are two methods on the assert class named AreEqual and AreSame. These methods may seem identical, but there is a critical difference between them. Please explain the difference.**

AreEqual – Verifies that two values are equal. In other words, checks for the equality in the values of the two objects, ignoring the fact that they might be different object references.

Are Same – Asserts that two objects refer to the same object. In other words, verifies that these objects point to the same place and/or have the same native object id, thus they are the same object.

**[Question 9 (4 pts)] What is the unit test TestThatHotelInitializes verifying?**

Verifies that the constructor is working, the object is being properly initialized and is not null as consequence of any initialization errors. In other words, checks if the constructor is giving an valid object.

**([Question 10 (4 pts)] What is the generic algorithm for calculating getBasePrice? (You may**

**look at Hotel.cs.)**

It corresponds to 45 times number Of Nights To Rent.

**[Question 11 (4 pts)] What cases are tested with these new tests?**

The 1 day, 2 days and 10 days cases.

**[Question 12 (4 pts)] Why don't we have to include Assert.IsNotNull(target) in each of**

**these tests?**

Because if target was null, calling target.getBasePrice would rise an exception and since no exceptions were allowed in the test cases this would make the test fail anyway. Additionally, we already have one test for checking if the Hotel initializes, in order to avoid code duplication, initialization related tests should be covered in this mentioned test method.

**[Question 13 (4 pts)] Why does this test expect an exception? (Please look at Hotel.cs.)**

Because -5 is an invalid amount of days to rent a hotel and, for this, in the constructor of the Hotel class there is an if statement throwing an exception in case invalid arguments like this are passed. As we want to be sure that the class is shielding itself from invalid parameters, we pass the parameter and expect it to be properly detected and the method to throw the correct exception.

**[Question 14 (4 pts)] Please provide the attribute declaration to accompany a test case that**

**expects an OutOfMemoryException.**

```
[ExpectedException(typeof(OutOfMemoryException))]
```