Tiffany Pohl
CM 784
Lab 4

1. In order to test the getRoomOccupant() function, a fake database must be created. Once the fake database is created, it is used to determine if the function is returning the correct data (this is done in the mocks.Record()). Once the mocking is handled, regular testing can be continued.
2. To throw an exception, use LastCall.Throw(Exception exception) which throws the specified exception with called.
3. You do not need to use a stub if the mocked object does not return a value; you can use a DynamicMock instead.
4. To test the AvailableRooms property, a fake database is created. The fake database then gets a fake list of rooms that it has. Once this is all created, the method can be tested normally with a nunit test.
5. In order to ensure that the service locator removes a car from its available cars when a car is booked, first a ServiceLocator was set up and a few sample cars were added to it. The value of the ServiceLocator.Instance was then set using reflection. Finally, the test case was finished with regular nunit assertions.