Lab 04

Michael Eaton

1. The database is created via a stub, then record is used to show that calls to it from getRoomOccupant for specific rooms should return the predefined values. The mock database is then queried for those rooms, and the result of the queries is returned.
2. Call a method, then use LastCall.throw(exception)
3. No need for a stub if no value is returned. You can replace the stub with a DynamicMock.

4. The database is created via a stub, then a list of 0-99 is created and assigned to it as rooms. The Hotel object the retrieves the list of rooms from the database and the number of rooms in the list that the hotel receives is tested against the number of items in the list that was passed to the database.

5. A ServiceLocator and two Cars are created. The Cars are added to the ServiceLocator. Then, all instances of a ServiceLocator are made to point to the ServiceLocator that we created. The User then books his Car, and the ServiceLocator is tested to see if it removes the Car from its list properly.