Erick Giovanni Guzmán Sierra

A01566171

Software Quality and Testing

Lab 2 Challenge

1. The purpose of the first three methods in step 1 is to check if the “expected” value (object, string, array, etc) is equal to the “actual” value. The forth method checks that the value is null. The fifth method (whenAssertingNotSameObject_thenDifferent) checks if two variables are different objects. The last method checks if an expression is true and then if a second statement is false.
2. The difference of the asserts is that one checks if both variables are equal, while the second one also checks the expression but also prints a message if the expressions aren’t equal.
3. Method:

```
      @Test
      public void whenAssertingNull_thenFalse() {
          Object longclaw = new Object();

          assertNotNull(longclaw, "The longclaw should NOT be null");
          //if we want to assert that an object should not be null we can use the 
assertNotNull assertion
      }
```

4. Method:

```
      @Test
      public void whenAssertingNotSameObject_thenSame() {
          Object oathkeeper = new Object();
          Object widowswall = oathkeeper;

          assertSame(oathkeeper, widowswall);
          // when we want to verify that two variables refer to the same object, 
we can use the assertSame assertion
      }
```

5. Method:

```
      public void methodThatShouldThrowException() {
            //throw new UnsupportedOperationException("Operation Not Supported");
      }
```

6. Method:

```
      @Test
      public void testAssertThatHasItems() {
          assertThat(Arrays.asList("Harrenhal", "Dragonstone", 
"Winterfell")).contains("Winterfell", "Dragonstone");
      }
```

7. Method:

```java
@Test
public void testMultiply() {
      int x = 998;

      assertTrue(x < 1000, "The value is less than 1000");
}
```

8. Method:

```java
@Test
public void testMultiply_ExceptionIsThrown() {
      int x = 1000;
      assertFalse(x < 1000, "The value is greater than 999");
}
```