Lab TDD 4: Testing an Exception

Objectives

In this lab you will add run a test case

The lab continues from where TDD 3 lab leftoff

Instructions Project Setup

Step 1: Add a divide test method

1. You are going to add a test method to implement the divide case of 4/2. In this case 2 – 2 == 0
2. Add a test method testDiv1 as shown below

```
@Test
public void testSub2() {
    int result = TestClass.c.sub(2, 3);
    assertEquals("Error message",-1, result);
}
```

3. Run the tests and ensure the test fails

Step 3: Add the production code

1. Add the code shown
2. Rerun the tests and see them pass

```
@Override
public int div(int a, int b) {
    return a/b;
}
```

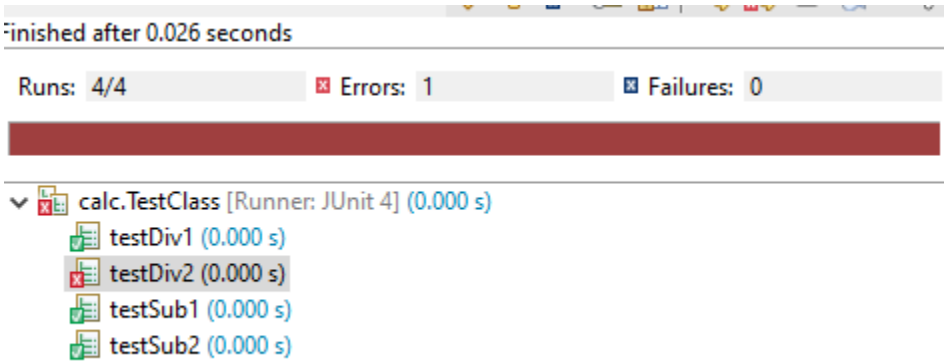
Step 4: Add a second test case for an exception

1. We want to throw an IllegalArgumentException if the user tries to divide by zero
2. Add the zero divide test case 4/0 but notice that there is nothing to test. Add a new test method testDiv2() to do this

```
@Test
public void testDiv2() {
    int result = TestClass.c.div(4,0);

}
```

3. The method throws an exception, in this case an arithmetic exception. When we run the test case, JUnit calls the test an error since an exception was thrown that was not expected.



4. What we can do is to tell JUnit we expect to throw an IllegalArgumentException.

```
@Test(expected = IllegalArgumentException.class)
public void testDiv2() {
    TestClass.c.div(4,0);
}
```

Step 5: Add the production code

1. Add the code that throws the correct exception to the implementation

```
@Override
public int div(int a, int b) {
    if (b == 0) throw new IllegalArgumentException();
    return a/b;
}
```

2. Run all the tests and ensure that they all pass