Lesson 01 Demo 04

Using Various Annotations in JUnit5

Objective: To use various annotations available in JUnit on methods and test cases

Tool required: Eclipse IDE

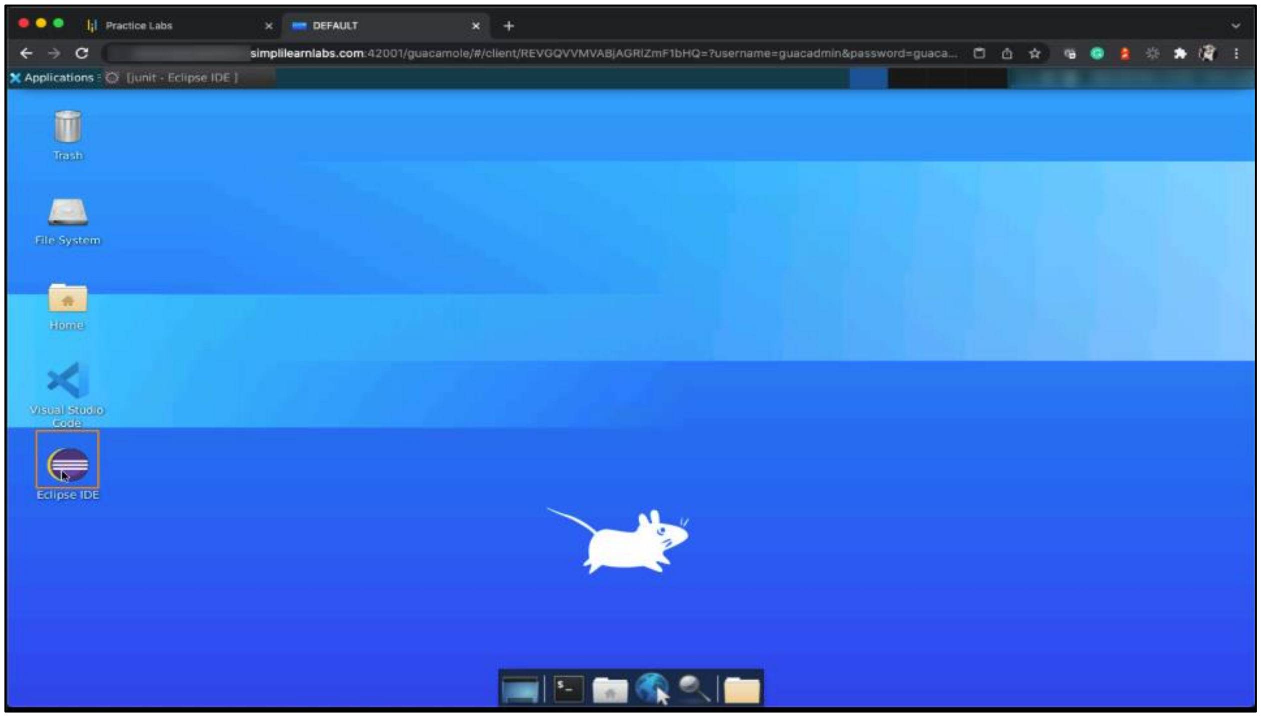
Prerequisites: None

Steps to be followed:

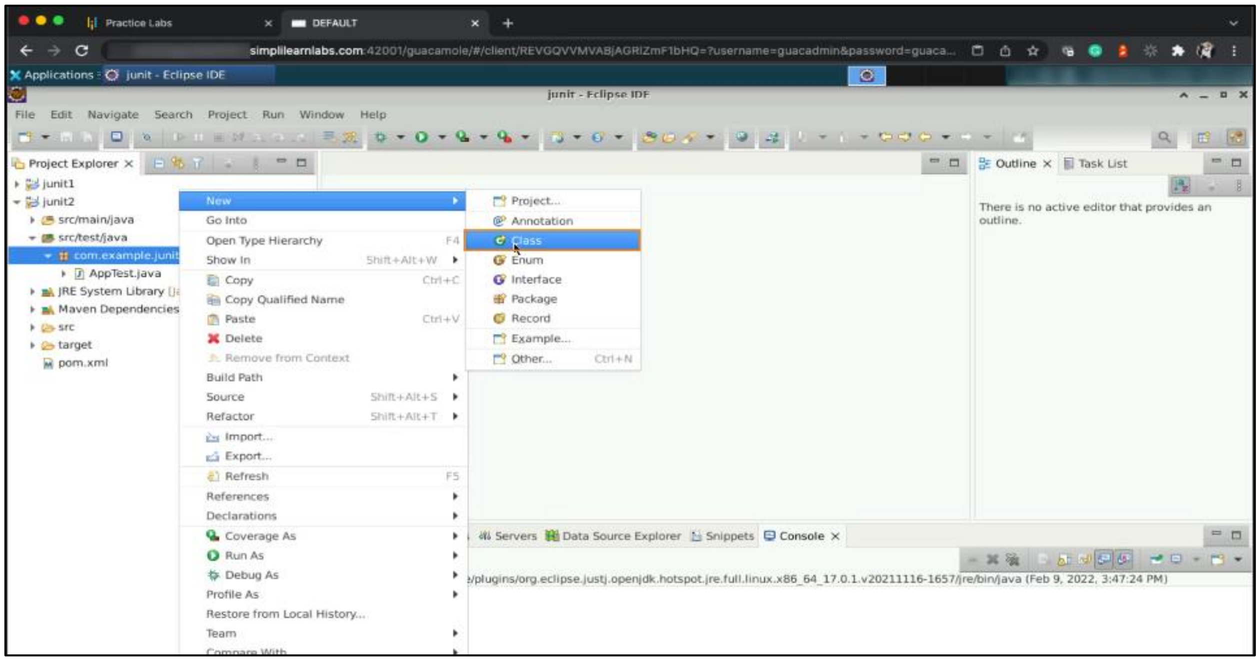
1. Executing BeforeAll, AfterAll, BeforeEach, and AfterEach annotations
2. Executing Disabled annotation
3. Executing fail() method

Step 1: Executing BeforeAll, AfterAll, BeforeEach, and AfterEach annotations

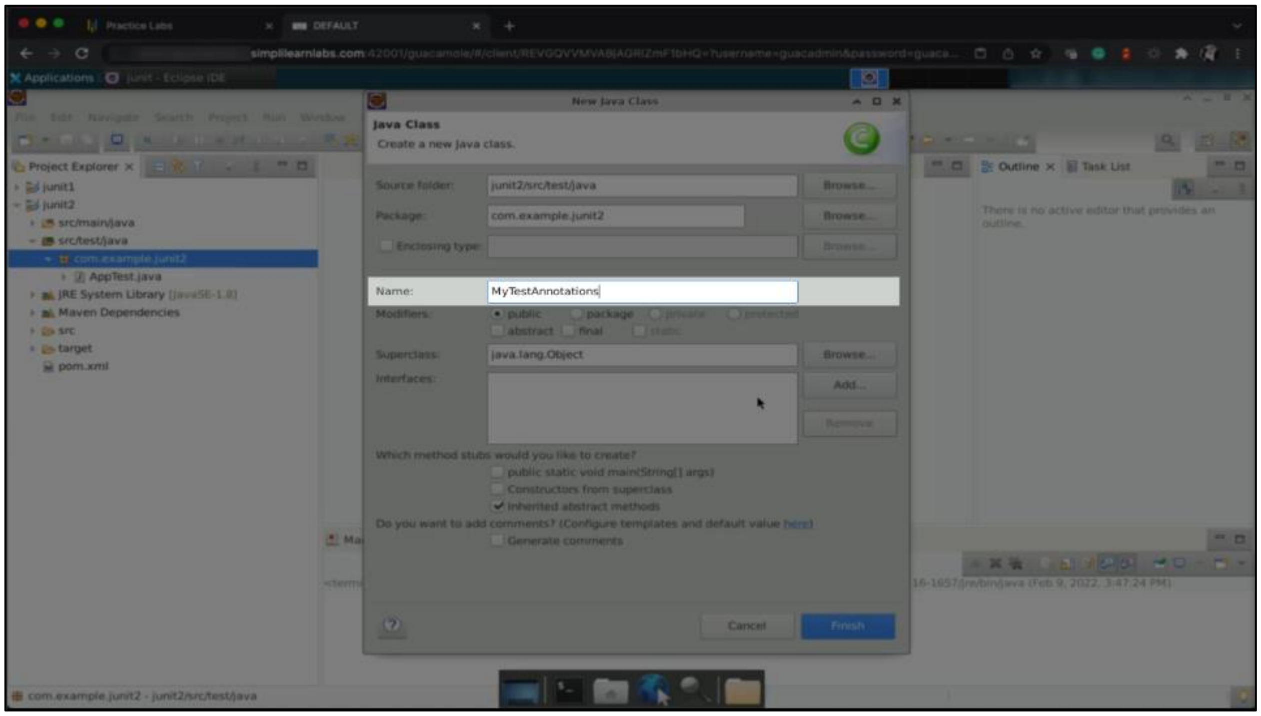
1.1 Open **Eclipse IDE**



1.2 Click the project name **junit2**. Go to **src/test/java**, right-click on **com.example.junit2,** go to **New,** and click on **Class**

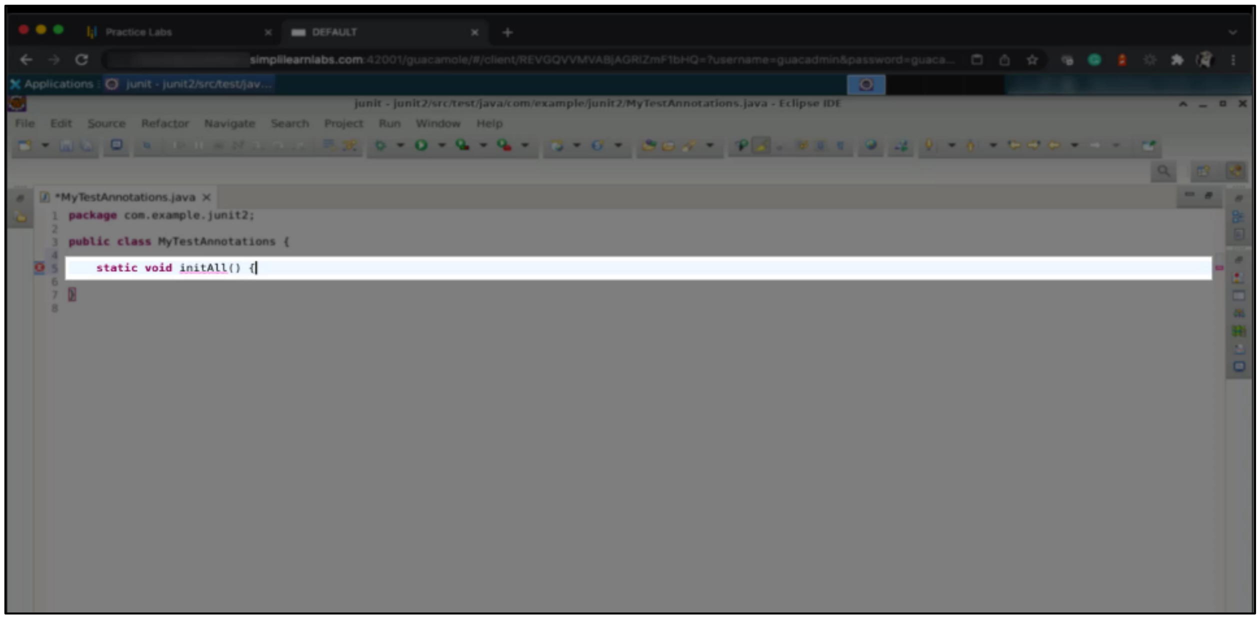


1.3 Name the class **MyTestAnnotations** and click **Finish**



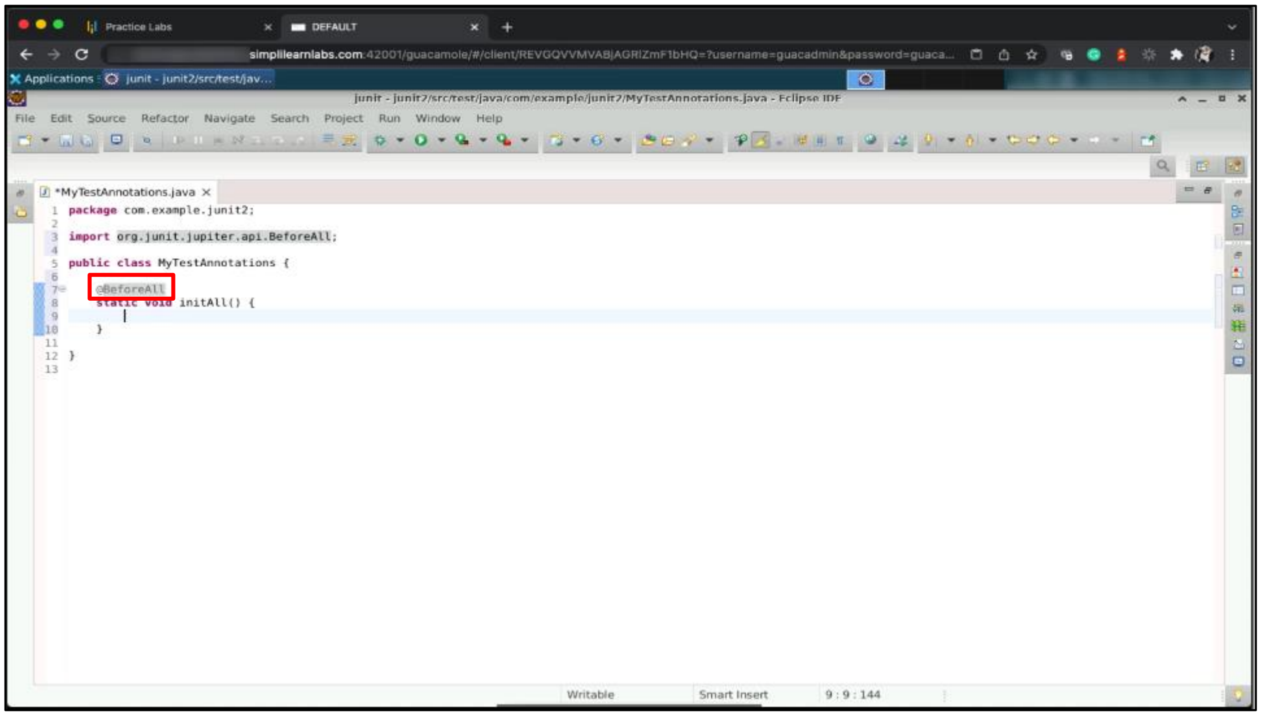
Source folder: junit2/src/test/java
Package: com.example.junit2
Name: MyTestAnnotations
Superclass: java.lang.Object

1.4 Create a new method **initAll()** as a static void function



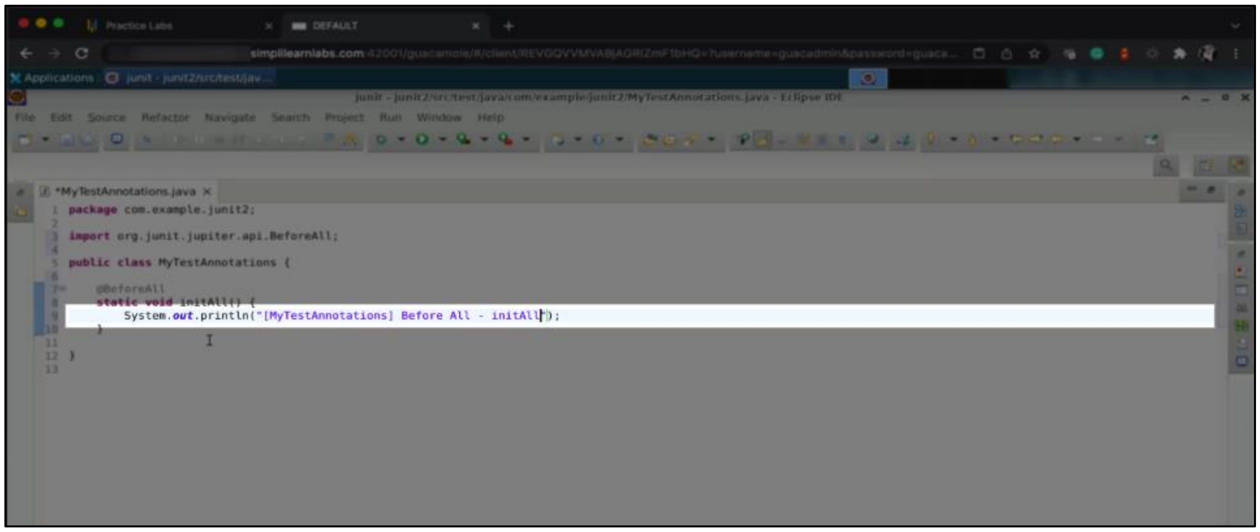
```java
package com.example.junit2;

public class MyTestAnnotations {
    static void initAll() {
```

1.5 Add the **@BeforeAll** annotation above the method to signal that this method should be executed before all the test cases in the class



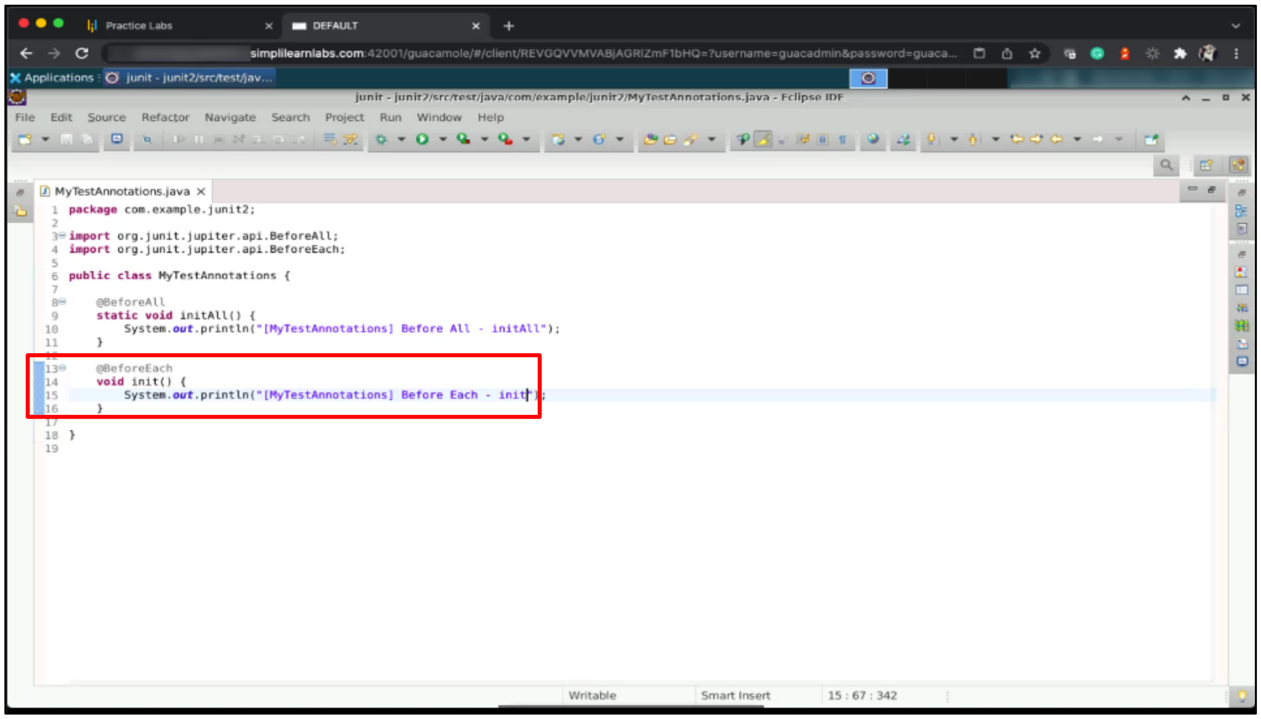
```java
package com.example.junit2;

import org.junit.jupiter.api.BeforeAll;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {

    }

}
```

1.6 Write the print statement to print **Before All** annotation



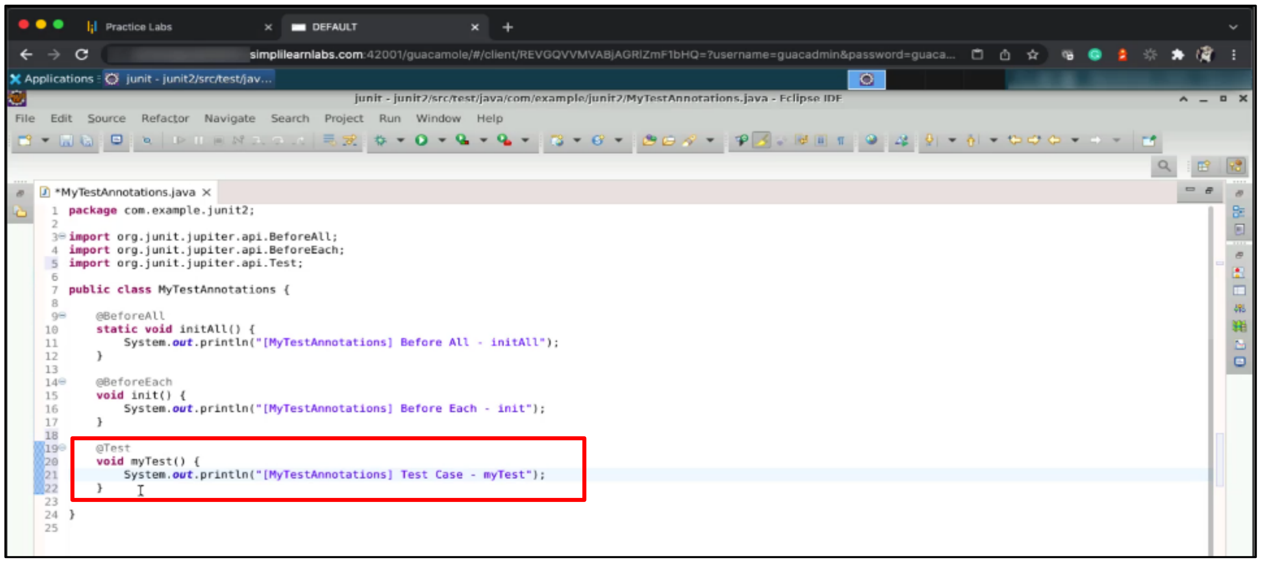
```java
package com.example.junit2;

import org.junit.jupiter.api.BeforeAll;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

}
```

1.7 Add one more method, **init()**, which is the same as above. Also, add **@BeforeEach** annotation before this method. To execute this method before each test case, add **a print statement** to print **Before Each** annotation



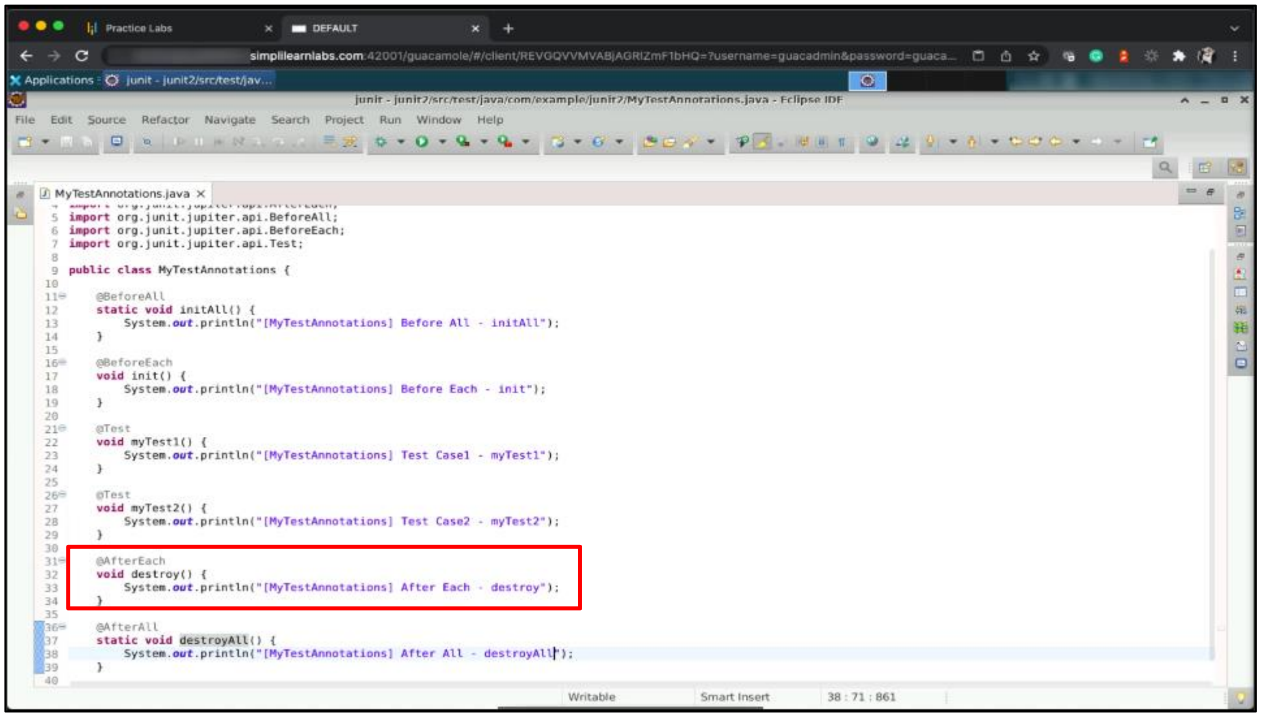
```java
package com.example.junit2;

import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

}
```

1.8 Create one test case **myTest()**. Add **@Test** annotation above the test case to inform the compiler that it is the test case and add a **print statement** to print this test case



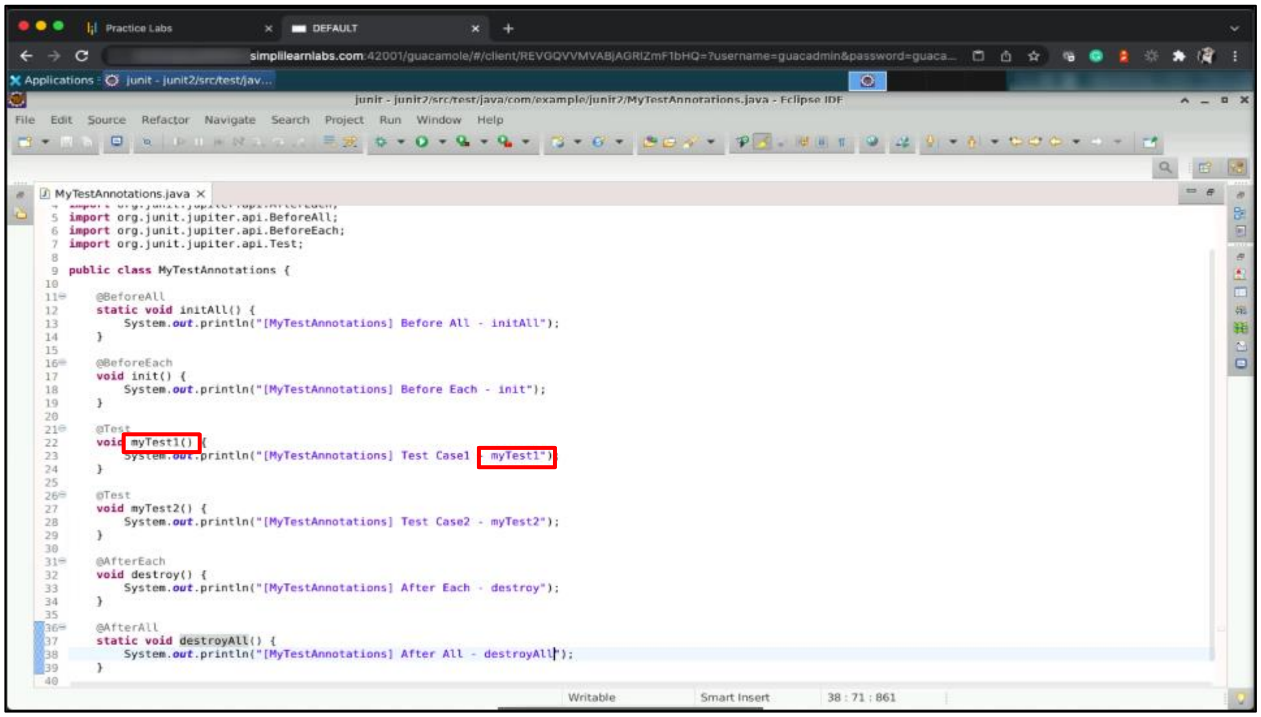
```java
package com.example.junit2;

import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest() {
        System.out.println("[MyTestAnnotations] Test Case - myTest");
    }

}
```

1.9 Create a method **destroy()** and add **@AfterEach** annotation before this method. Execute this method after each test case. Add **print statement** in this method to print **@AfterEach** annotation



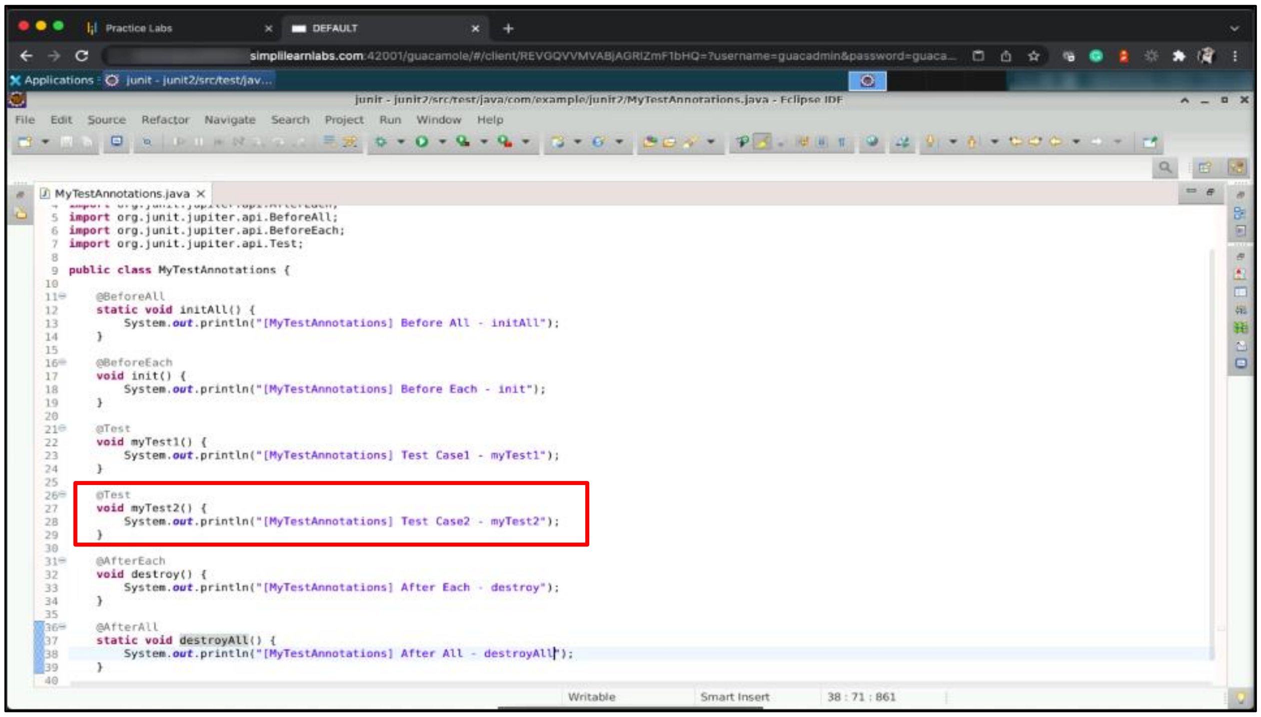
```java
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }

```

1.10 Change the name of the test case to **myTest1** to avoid confusion


Practice Labs
DEFAULT
junit - junit2/src/test/java/com/example/junit2/MyTestAnnotations.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
MyTestAnnotations.java
Writable
Smart Insert
38 : 71 : 861

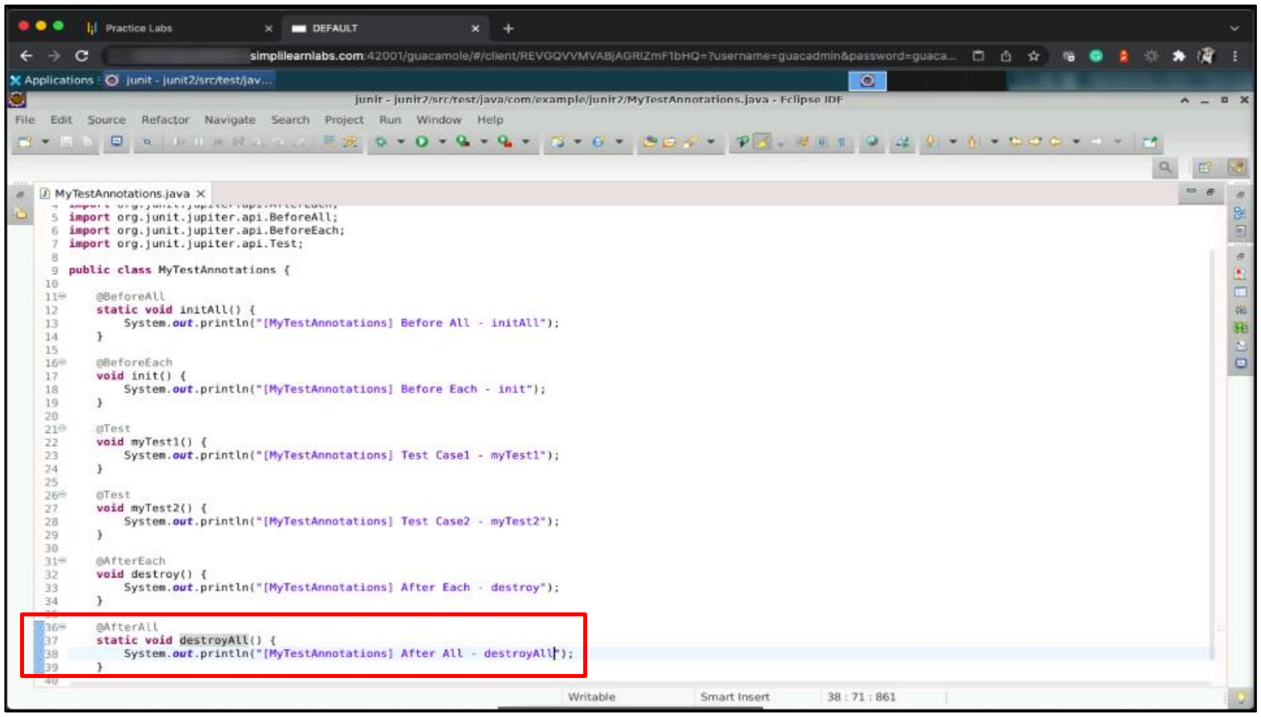
```java
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }
```

1.11 Create another test case same as the previous test case with the name **myTest2**


Practice Labs
DEFAULT
junit - junit2/src/test/java/com/example/junit2/MyTestAnnotations.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
MyTestAnnotations.java
Writable
Smart Insert
38 : 71 : 861

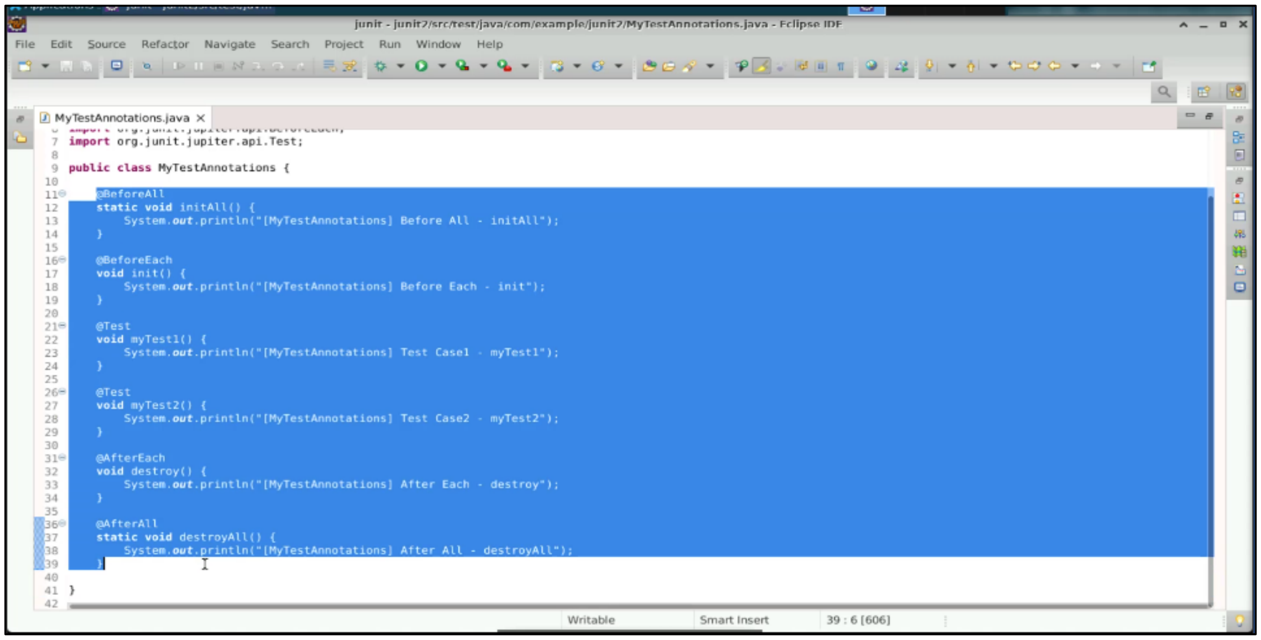
```java
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }
```

1.12 Now, create one more method **destroyAll()** and add **@AfterAll** annotation before this method. Execute this method after every test case. Add **print statement** in this method to print **After All** annotation


junit - junit2/src/test/java/com/example/junit2/MyTestAnnotations.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
MyTestAnnotations.java
Writable
Smart Insert
38 : 71 : 861

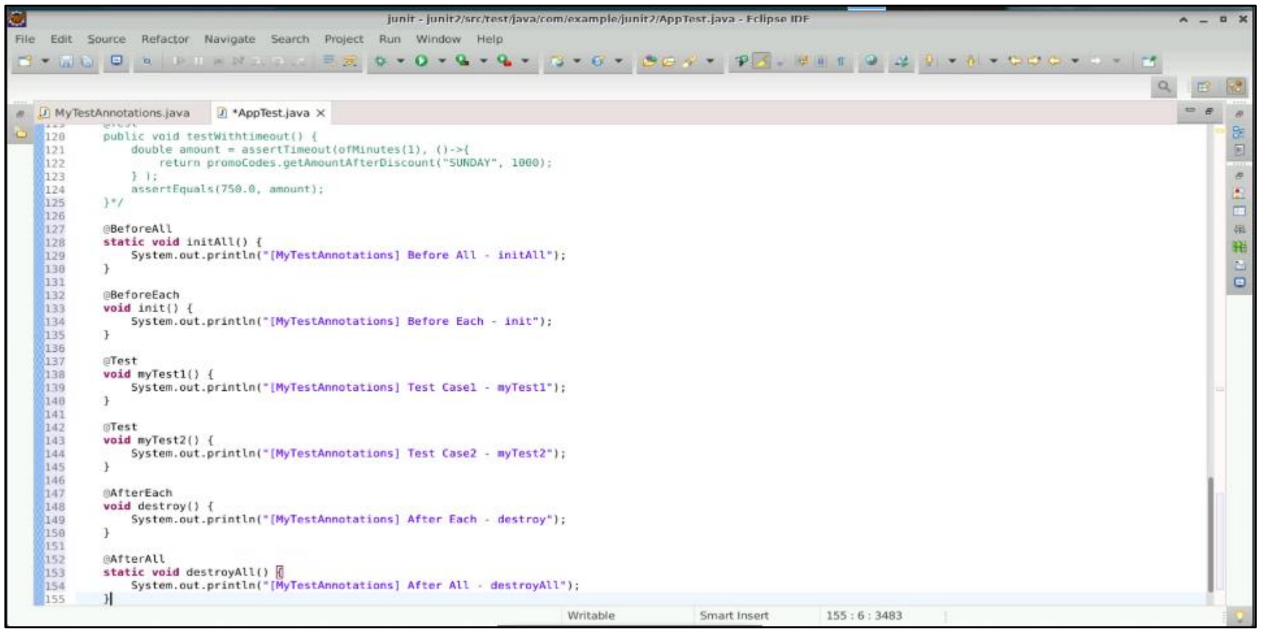
```java
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }
```

1.13 Copy the entire code within the class **MyTestAnnotations.** Cut the entire code


junit - junit2/src/test/java/com/example/junit2/MyTestAnnotations.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
MyTestAnnotations.java
Writable
Smart Insert
39 : 6 [606]

```java
import org.junit.jupiter.api.Test;

public class MyTestAnnotations {

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }

}
```

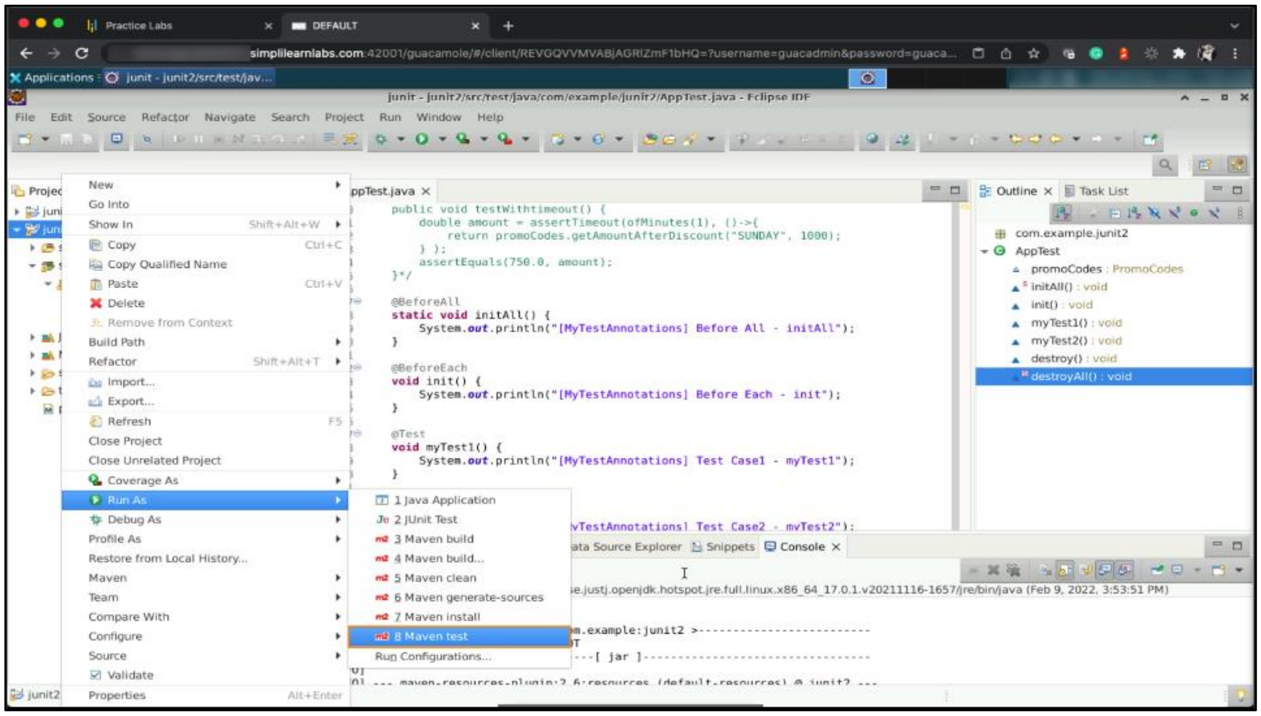
1.14 Go to the **AppTest.java** file and comment on all the test cases created in that file. Then, paste the code copied in the last step into this file



```
    public void testWithtimeout() {
        double amount = assertTimeout(ofMinutes(1), ()->{
            return promoCodes.getAmountAfterDiscount("SUNDAY", 1000);
        } );
        assertEquals(750.0, amount);
    }*/

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }
```

junit - junit2/src/test/java/com/example/junit2/AppTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
MyTestAnnotations.java
*AppTest.java
Writable
Smart Insert
155 : 6 : 3483

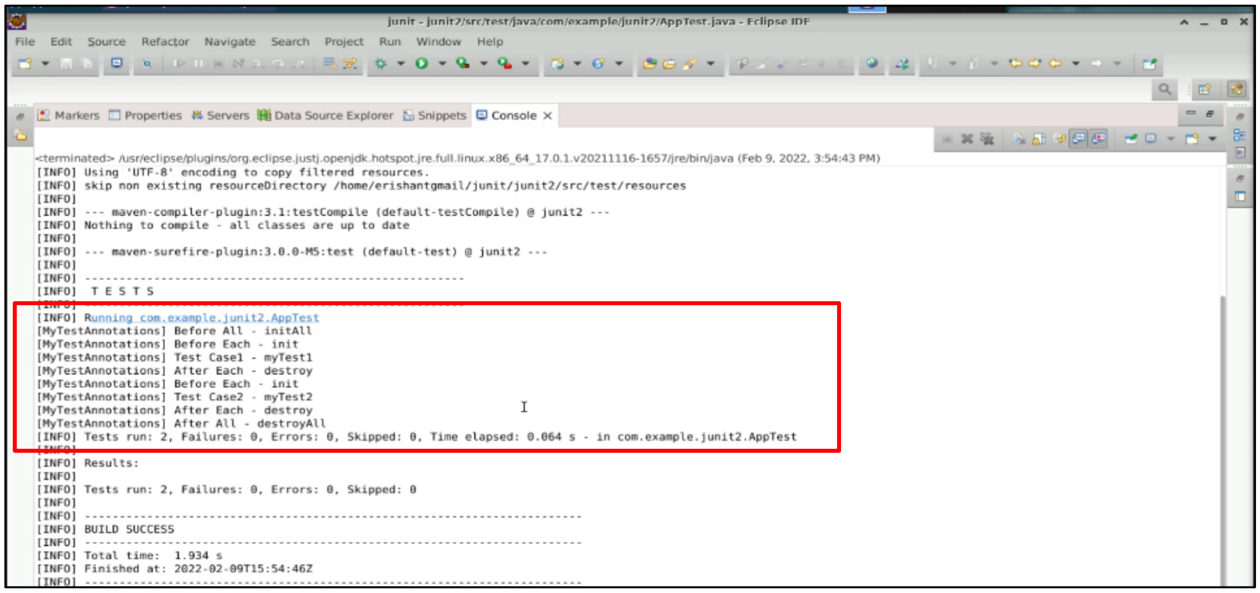
1.15 Open **Project Explorer**, right-click on project name **junit2,** click on **Run As,** and run the project as a **Maven test**



```
    public void testWithtimeout() {
        double amount = assertTimeout(ofMinutes(1), ()->{
            return promoCodes.getAmountAfterDiscount("SUNDAY", 1000);
        } );
        assertEquals(750.0, amount);
    }*/

    @BeforeAll
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }
```

```
e.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Feb 9, 2022, 3:53:51 PM)
m.example:junit2 >------------------------
T
---[ jar ]-----------------------------
--- maven-resources-plugin:2.6:resources (default-resources) @ junit2 ---
```

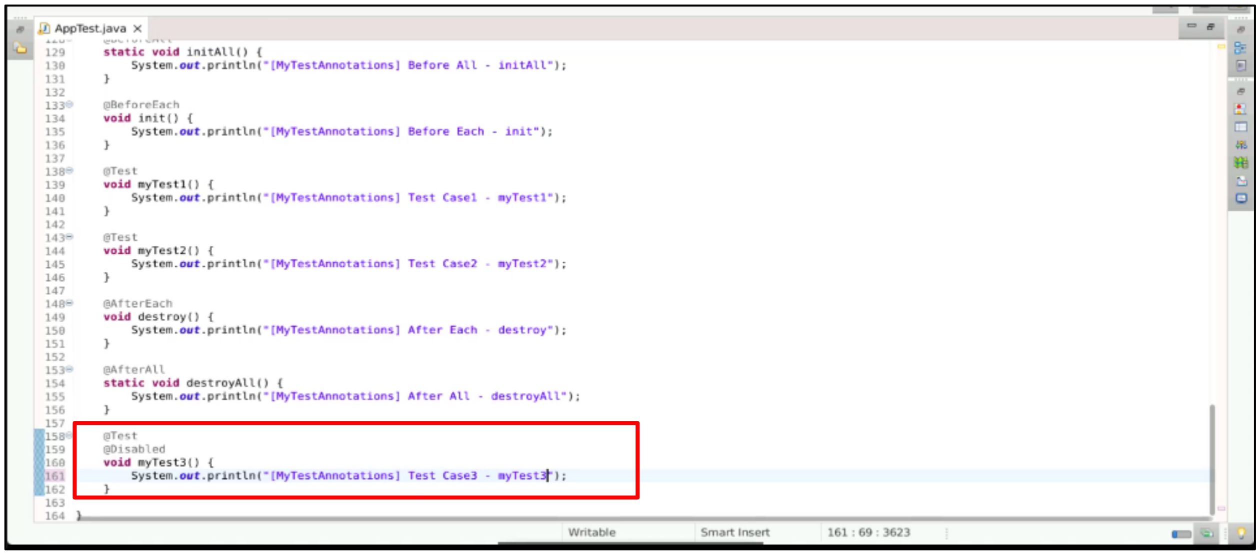
1.16 In the console, and you can see that all the methods and test cases are successfully executed. You can also see the sequence in which they are executed.



```
<terminated> /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Feb 9, 2022, 3:54:43 PM)
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/erishantgmail/junit/junit2/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ junit2 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:3.0.0-M5:test (default-test) @ junit2 ---
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running com.example.junit2.AppTest
[MyTestAnnotations] Before All - initAll
[MyTestAnnotations] Before Each - init
[MyTestAnnotations] Test Case1 - myTest1
[MyTestAnnotations] After Each - destroy
[MyTestAnnotations] Before Each - init
[MyTestAnnotations] Test Case2 - myTest2
[MyTestAnnotations] After Each - destroy
[MyTestAnnotations] After All - destroyAll
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.064 s - in com.example.junit2.AppTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.934 s
[INFO] Finished at: 2022-02-09T15:54:46Z
[INFO] ------------------------------------------------------------------------
```

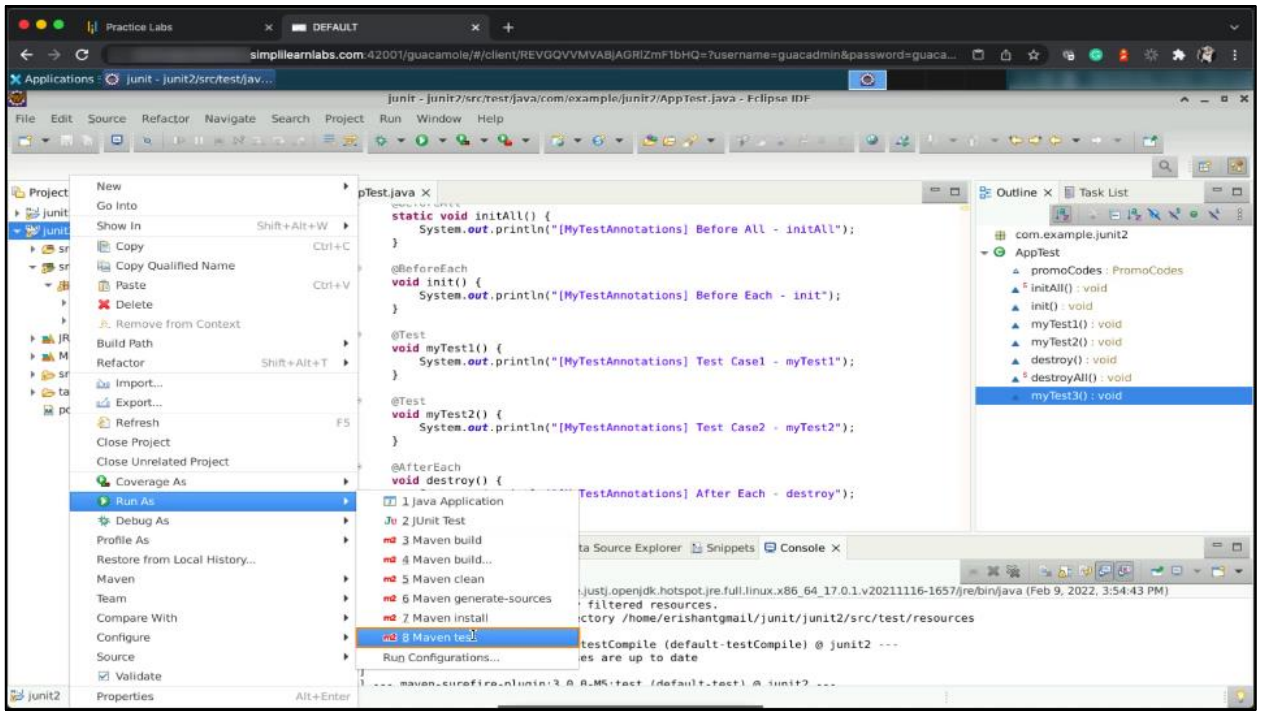
Step 2: Executing Disabled annotation

2.1 Come back to the **AppTest.java** file. Create another test case with the name **myTest3**. This time add one more annotation as **@Disabled** above this test case. Do not execute this test case. Add print statement



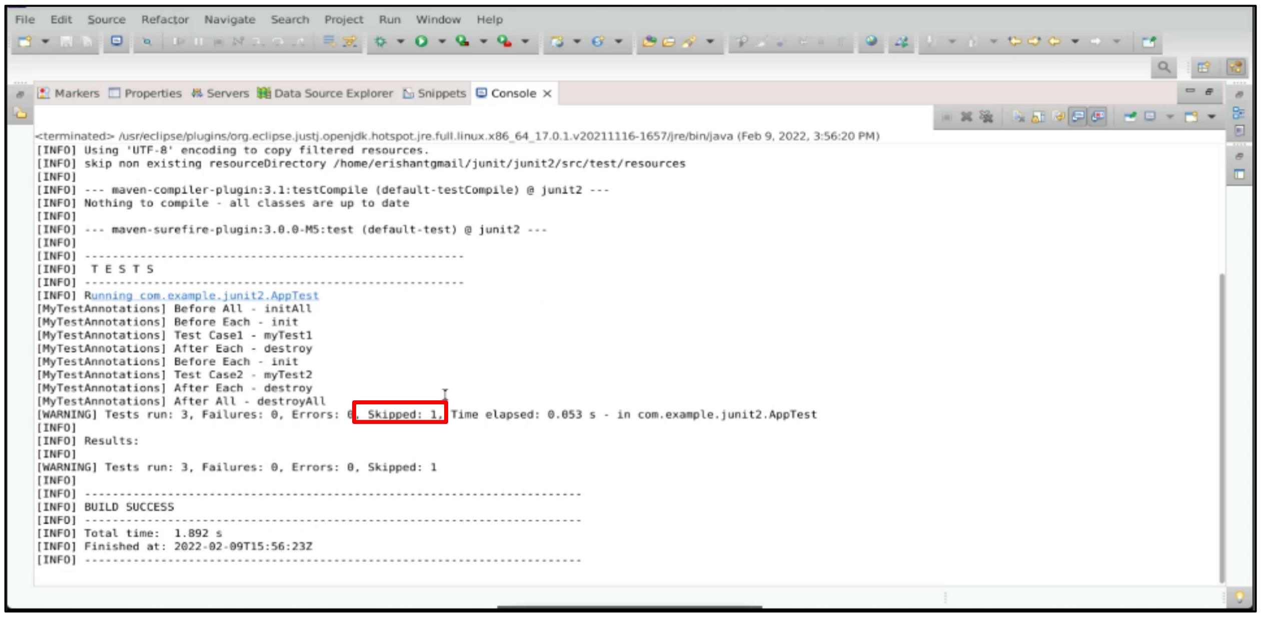
```java
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }

    @AfterAll
    static void destroyAll() {
        System.out.println("[MyTestAnnotations] After All - destroyAll");
    }

    @Test
    @Disabled
    void myTest3() {
        System.out.println("[MyTestAnnotations] Test Case3 - myTest3");
    }

}
```

2.2 Rerun the project as a **Maven test** again



```
static void initAll() {
    System.out.println("[MyTestAnnotations] Before All - initAll");
}

@BeforeEach
void init() {
    System.out.println("[MyTestAnnotations] Before Each - init");
}

@Test
void myTest1() {
    System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
}

@Test
void myTest2() {
    System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
}

@AfterEach
void destroy() {
```

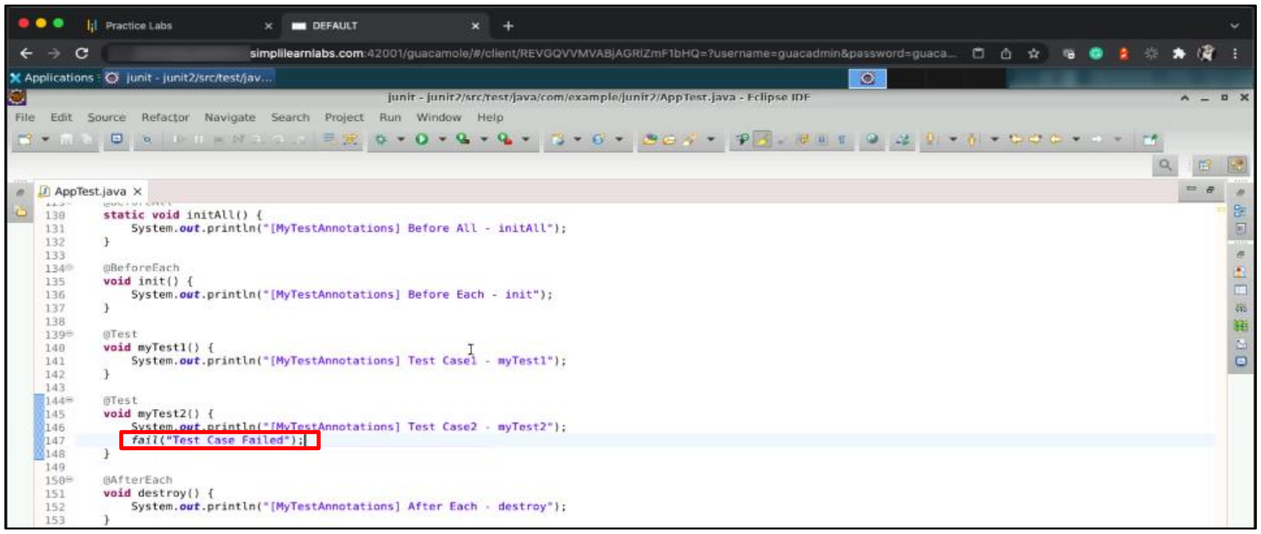
You can see that all three test cases are executed, but only two of them are visible in the output, and one test case is skipped.



```
<terminated> /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Feb 9, 2022, 3:56:20 PM)
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/erishantgmail/junit/junit2/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ junit2 ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:3.0.0-M5:test (default-test) @ junit2 ---
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running com.example.junit2.AppTest
[MyTestAnnotations] Before All - initAll
[MyTestAnnotations] Before Each - init
[MyTestAnnotations] Test Case1 - myTest1
[MyTestAnnotations] After Each - destroy
[MyTestAnnotations] Before Each - init
[MyTestAnnotations] Test Case2 - myTest2
[MyTestAnnotations] After Each - destroy
[MyTestAnnotations] After All - destroyAll
[WARNING] Tests run: 3, Failures: 0, Errors: 0, Skipped: 1, Time elapsed: 0.053 s - in com.example.junit2.AppTest
[INFO]
[INFO] Results:
[INFO]
[WARNING] Tests run: 3, Failures: 0, Errors: 0, Skipped: 1
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.892 s
[INFO] Finished at: 2022-02-09T15:56:23Z
[INFO] ------------------------------------------------------------------------
```

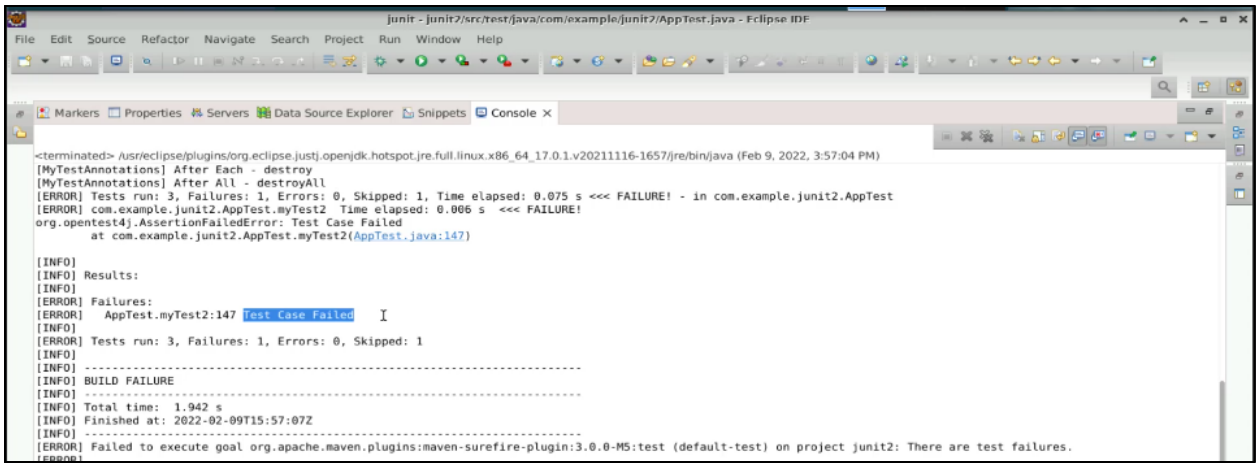
Step 3: Executing the fail() method

3.1 Go back to the **AppTest.java** file. In **myTest2,** add the **fail()** method with the message **Test Case Failed,** so that after compilation this test fails and gives the error



```java
    static void initAll() {
        System.out.println("[MyTestAnnotations] Before All - initAll");
    }

    @BeforeEach
    void init() {
        System.out.println("[MyTestAnnotations] Before Each - init");
    }

    @Test
    void myTest1() {
        System.out.println("[MyTestAnnotations] Test Case1 - myTest1");
    }

    @Test
    void myTest2() {
        System.out.println("[MyTestAnnotations] Test Case2 - myTest2");
        fail("Test Case Failed");
    }

    @AfterEach
    void destroy() {
        System.out.println("[MyTestAnnotations] After Each - destroy");
    }
```

3.2 Now, run the project as a **Maven test** again. In the output, you can see that there is 1 failed test with the message **Test Case Failed,** which is throwing the **AssertionFailedError**



```
<terminated> /usr/eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.linux.x86_64_17.0.1.v20211116-1657/jre/bin/java (Feb 9, 2022, 3:57:04 PM)
[MyTestAnnotations] After Each - destroy
[MyTestAnnotations] After All - destroyAll
[ERROR] Tests run: 3, Failures: 1, Errors: 0, Skipped: 1, Time elapsed: 0.075 s <<< FAILURE! - in com.example.junit2.AppTest
[ERROR] com.example.junit2.AppTest.myTest2  Time elapsed: 0.006 s  <<< FAILURE!
org.opentest4j.AssertionFailedError: Test Case Failed
        at com.example.junit2.AppTest.myTest2(AppTest.java:147)

[INFO]
[INFO] Results:
[INFO]
[ERROR] Failures:
[ERROR]   AppTest.myTest2:147 Test Case Failed
[INFO]
[ERROR] Tests run: 3, Failures: 1, Errors: 0, Skipped: 1
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.942 s
[INFO] Finished at: 2022-02-09T15:57:07Z
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-surefire-plugin:3.0.0-M5:test (default-test) on project junit2: There are test failures.
```

This is how you can work with different annotations in JUnit. You can also use multiple annotations on a single method or a test case.