Last Updated: 3/13/15

Topic Name: JUnit Tests

Brief Introduction:

JUnit is a testing framework that's used for Java code testing. It is used in development testing. One purpose for using JUnit it to test code as you develop it as you go so you're your code is a little more complete at the end. JUnit makes very nice for developers to create test classes that will be able to test the code more specifically of a function or process.
JUnit Testing allows us to:

- Find problems early
- Facilitate change
- Simplify integration
- Documentation
- Design

There are a few limitations to JUnit Testing such as not being able to catch every error. It also can take a longer time to write all the tests for the application. For each line of code there may be 2, 3…5 or more lines of code that must be written to test the code with depending on all the variables that are being used.

@Test public void method()
    The `@Test` annotation identifies a method as a test method.

@Test (expected = Exception.class)
    Fails if the method does not throw the named exception.

@Test(timeout=100)
    Fails if the method takes longer than 100 milliseconds.

@Before public void method()
    This method is executed before each test. It is used to prepare the test environment (e.g., read input data, initialize the class).

@After public void method()
    This method is executed after each test. It is used to cleanup the test environment (e.g., delete temporary data, restore defaults). It can also save memory by cleaning up expensive memory structures.

@BeforeClass public static void method()
    This method is executed once, before the start of all tests. It is used to perform time intensive activities, for example, to connect to a database. Methods marked with this annotation need to be defined as `static` to work with JUnit.

@AfterClass public static void method()

This method is executed once, after all tests have been finished. It is used to perform clean-up activities, for example, to disconnect from a database. Methods annotated with this annotation need to be defined as `static` to work with JUnit.

@Ignore

Ignores the test method. This is useful when the underlying code has been changed and the test case has not yet been adapted. Or if the execution time of this test is too long to be included.

```java
import org.junit.*;

public class TestFoobar {
    @BeforeClass
    public static void setUpClass() throws Exception {
        // Code executed before the first test method
    }

    @Before
    public void setUp() throws Exception {
        // Code executed before each test
    }

    @Test
    public void testOneThing() {
        // Code that tests one thing
    }

    @Test
    public void testAnotherThing() {
        // Code that tests another thing
    }

    @Test
    public void testSomethingElse() {
        // Code that tests something else
    }

    @After
    public void tearDown() throws Exception {
        // Code executed after each test
    }

    @AfterClass
    public static void tearDownClass() throws Exception {
        // Code executed after the last test method
    }
}
```

Teaching Description:
None yet.

Teaching Examples:
None yet.

Files to View:
None yet.