1. JUnit program that demonstrates the use of lifecycle methods:

import org.junit.\*;

public class MyTest {

@BeforeClass

public static void setUpClass() {

System.out.println("Running setUpClass...");

}

@Before

public void setUp() {

System.out.println("Running setUp...");

}

@Test

public void test1() {

System.out.println("Running test1...");

}

@Test

public void test2() {

System.out.println("Running test2...");

}

@After

public void tearDown() {

System.out.println("Running tearDown...");

}

@AfterClass

public static void tearDownClass() {

System.out.println("Running tearDownClass...");

}

}

In this example, we have defined a JUnit test class called MyTest. This class contains five methods that represent the different lifecycle methods:

* setUpClass: This method is executed once before all the tests in the class are run.
* setUp: This method is executed before each test method.
* test1 and test2: These are the actual test methods that we want to run.
* tearDown: This method is executed after each test method.
* tearDownClass: This method is executed once after all the tests in the class are run.

Each of these methods simply prints a message to the console indicating that it is being executed.