**Exercise 6: Verifying Interaction Order**

**Scenario:** You need to ensure that methods are called in a specific order.

**LifecycleManager :-**

package org.example1;  
  
public interface LifecycleManager {  
 void initialize();  
 void process();  
 void cleanup();  
}

**LifecycleManagerTest.java :-**

package org.example1;  
  
  
import org.mockito.InOrder;  
import org.junit.jupiter.api.Test;  
import static org.mockito.Mockito.\*;  
  
public class LifecycleManagerTest {  
  
 @Test  
 public void testMethodCallOrder() {  
 // Step 1: Create mock  
 LifecycleManager mockManager = *mock*(LifecycleManager.class);  
  
 // Step 2: Call methods in a specific order  
 mockManager.initialize();  
 mockManager.process();  
 mockManager.cleanup();  
  
 // Step 3: Verify the order of interaction  
 InOrder inOrder = *inOrder*(mockManager);  
 inOrder.verify(mockManager).initialize();  
 inOrder.verify(mockManager).process();  
 inOrder.verify(mockManager).cleanup();  
 }  
}

**pom.xml :-**

<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  
 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>org.example1</groupId>  
 <artifactId>method-call-order</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>17</maven.compiler.source> <!-- or 11/1.8 -->  
 <maven.compiler.target>17</maven.compiler.target>  
 </properties>  
  
 <dependencies>  
 <!-- JUnit 5 -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>5.10.0</version>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- Mockito Core -->  
 <dependency>  
 <groupId>org.mockito</groupId>  
 <artifactId>mockito-core</artifactId>  
 <version>5.11.0</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <!-- Use Surefire Plugin for test execution -->  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>3.1.2</version>  
 </plugin>  
 </plugins>  
 </build>  
</project>

**Output :-**

