**ADVANCED MOCKITO EXERCISES**

**PROGRAM STRUCTURE:**

AdvancedMockitoExercises/

├── pom.xml

└── src/

├── main/java/com/example/

│ ├── EvenChecker.java

│ ├── ExceptionThrower.java

│ └── PerformanceTester.java

└── test/java/com/example/

├── EvenCheckerTest.java

├── ExceptionThrowerTest.java

├── PerformanceTesterTest.java

├── OrderedTests.java

**EvenChecker.java**

package com.example;

public class EvenChecker {

    public boolean isEven(int number) {

        return number % 2 == 0;

    }

}

**ExceptionThrower.java**

package com.example;

public class ExceptionThrower {

    public void throwException() {

        throw new IllegalArgumentException("Invalid input");

    }

}

**PerformanceTester.java**

package com.example;

public class PerformanceTester {

    public void performTask() throws InterruptedException {

        Thread.sleep(400); // simulate work

    }

}

**EvenCheckerTest.java**

package com.example;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.params.ParameterizedTest;

import org.junit.jupiter.params.provider.ValueSource;

public class EvenCheckerTest {

    EvenChecker checker = new EvenChecker();

    @ParameterizedTest

    @ValueSource(ints = {2, 4, 6, 8, 10})

    void testEvenNumbers(int number) {

        assertTrue(checker.isEven(number));

    }

    @ParameterizedTest

    @ValueSource(ints = {1, 3, 5, 7, 9})

    void testOddNumbers(int number) {

        assertFalse(checker.isEven(number));

    }

}

**ExceptionThrowerTest.java**

package com.example;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

public class ExceptionThrowerTest {

    @Test

    void testThrowException() {

        ExceptionThrower thrower = new ExceptionThrower();

        assertThrows(IllegalArgumentException.class, thrower::throwException);

    }

}

OrderedTests.java

package com.example;

import org.junit.jupiter.api.\*;

@TestMethodOrder(MethodOrderer.OrderAnnotation.class)

public class OrderedTests {

    @Test

    @Order(1)

    void firstTest() {

        System.out.println("First Test");

    }

    @Test

    @Order(2)

    void secondTest() {

        System.out.println("Second Test");

    }

    @Test

    @Order(3)

    void thirdTest() {

        System.out.println("Third Test");

    }

}

**PerformanceTesterTest.java**

package com.example;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

import org.junit.jupiter.api.Timeout;

import java.util.concurrent.TimeUnit;

public class PerformanceTesterTest {

    @Test

    @Timeout(value = 1, unit = TimeUnit.SECONDS)

    void testPerformTaskWithinTime() throws InterruptedException {

        PerformanceTester tester = new PerformanceTester();

        tester.performTask(); // Should complete in < 1 second

    }

}

**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>com.example</groupId>

    <artifactId>AdvancedJUnitExercises</artifactId>

    <version>1.0-SNAPSHOT</version>

    <dependencies>

        <dependency>

            <groupId>org.junit.jupiter</groupId>

            <artifactId>junit-jupiter</artifactId>

            <version>5.10.0</version>

            <scope>test</scope>

        </dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>

                <groupId>org.apache.maven.plugins</groupId>

                <artifactId>maven-surefire-plugin</artifactId>

                <version>3.1.2</version>

            </plugin>

        </plugins>

    </build>

</project>

**OUTPUT:**



