**Phase-3 Practice Project: Assisted Practice -**

**5.Write a program to demonstrate dynamic tests.**

**Source Code:**

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com</groupId>

<artifactId>junit-tests</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-engine -->

<dependency>

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

<artifactId>junit-jupiter-engine</artifactId>

<version>5.9.3</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.platform/junit-platform-runner -->

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-runner</artifactId>

<version>1.9.3</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>maven-war-plugin</artifactId>

<version>3.3.1</version>

</plugin>

</plugins>

<finalName>JunitDemo</finalName>

</build>

</project>

**DynamicTests.java:**

**package** com;

**import** java.util.ArrayList;

**import** java.util.Collection;

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

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

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

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

**class** DynamicTests {

**private** CalculateAreaApp calculateAreaApp;

@BeforeEach

**void** init() {

**this**.calculateAreaApp = **new** CalculateAreaApp();

}

@TestFactory

Collection<DynamicTest> dynamicTests() {

Collection<DynamicTest> tests = **new** ArrayList<>();

tests.add(DynamicTest.*dynamicTest*("both arguments positive", ()-> { Assertions.*assertEquals*(2, calculateAreaApp.calcArea(2, 1)); } ));

tests.add(DynamicTest.*dynamicTest*("one positive and one zero", ()-> { Assertions.*assertEquals*(0, calculateAreaApp.calcArea(10, 0)); } ));

tests.add(DynamicTest.*dynamicTest*("both arguments negative", ()-> { Assertions.*assertEquals*(1, calculateAreaApp.calcArea(-1, -1)); } ));

tests.add(DynamicTest.*dynamicTest*("one positive and one negative", ()-> { Assertions.*assertEquals*(-1, calculateAreaApp.calcArea(1, -1)); } ));

**return** tests;

}

}

**OUTPUT:**

