**Exercise 1: Control Structures**

DECLARE

v\_number NUMBER := 10;

BEGIN

IF v\_number > 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Positive Number');

ELSIF v\_number = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Zero');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Negative Number');

END IF;

END;

**Output:**

Positive Number

**Exercise 3: Stored Procedures**

-- Creating the procedure

CREATE OR REPLACE PROCEDURE get\_employee\_salary (

p\_emp\_id IN EMPLOYEES.EMPLOYEE\_ID%TYPE,

p\_salary OUT EMPLOYEES.SALARY%TYPE

) AS

BEGIN

SELECT SALARY INTO p\_salary

FROM EMPLOYEES

WHERE EMPLOYEE\_ID = p\_emp\_id;

END;

-- Executing the procedure

DECLARE

v\_salary EMPLOYEES.SALARY%TYPE;

BEGIN

get\_employee\_salary(100, v\_salary);

DBMS\_OUTPUT.PUT\_LINE('Salary: ' || v\_salary);

END;

**Output:**

Salary: 24000

**Exercise 1: Setting Up Junit**

@Test

void testAddition() {

Calculator calc = new Calculator();

assertEquals(5, calc.add(2, 3));

}

**Output:**

Test passed.

**Exercise 3: Assertions in Junit**

@Test

void testAssertions() {

int result = new Calculator().add(4, 6);

assertEquals(10, result);

assertTrue(result > 5);

assertNotNull(result);

}

**Output:**

All assertions passed.

**Exercise 4: AAA Pattern, Setup, Teardown**

@BeforeEach

void setup() {

bank = new Bank();

}

@AfterEach

void teardown() {

bank = null;

}

@Test

void testWithdraw() {

bank.deposit(1000);

boolean result = bank.withdraw(500);

assertTrue(result);

assertEquals(500, bank.getBalance());

}

**Output:**

Test passed. Balance = 500

**Exercise 1: Mocking and Stubbing**

@Test

void testMocking() {

List<String> mockList = mock(List.class);

when(mockList.get(0)).thenReturn("Hello");

assertEquals("Hello", mockList.get(0));

}

**Output:**

Test passed. mockList.get(0) returned "Hello"

**Exercise 2: Verifying Interactions**

@Test

void testVerify() {

List<String> mockList = mock(List.class);

mockList.add("Test");

verify(mockList).add("Test");

verify(mockList, times(1)).add("Test");

}

**Output:**

Test passed. add("Test") was called once.

**Exercise 1: Logging Error Messages and Warning Levels**

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class LogExample {

private static final Logger logger = LoggerFactory.getLogger(LogExample.class);

public static void main(String[] args) {

logger.info("Application started");

logger.warn("This is a warning");

logger.error("An error occurred", new RuntimeException("Test error"));

}

}

**Output:**

INFO LogExample - Application started

WARN LogExample - This is a warning

ERROR LogExample - An error occurred

java.lang.RuntimeException: Test error

at LogExample.main(LogExample.java:9)