

# PL/SQL- HANDS-ON EXERCISE

## Exercise 1: Control Structures

**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

- **Question:** Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

- **Question:** Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over \$10,000.

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

- **Question:** Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

## TABLE CREATION:

### TABLE- CUSTOMERS

The screenshot displays the Oracle Live SQL web interface. On the left, the 'Navigator' pane shows 'My Schema' and a list of tables including 'CUSTOMERS'. The main editor area, titled '[SQL Worksheet]\*', contains the following SQL code:

```
1 CREATE TABLE CUSTOMERS (  
2   CUSTOMERID NUMBER PRIMARY KEY,  
3   NAME VARCHAR2(100),  
4   AGE NUMBER,  
5   BALANCE NUMBER(10, 2),  
6   ISVIP CHAR(1) DEFAULT 'N'  
7 )
```

Below the editor, the 'Script output' pane shows the execution results:

```
Elapsed: 00:00:00.032  
  
SQL> CREATE TABLE CUSTOMERS (  
      CUSTOMERID NUMBER PRIMARY KEY,  
      NAME      VARCHAR2(100),  
      AGE       NUMBER,...  
Show more...  
  
Table CUSTOMERS created.  
Elapsed: 00:00:00.018
```

At the bottom of the interface, there are links for 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Delete Your Live SQL Account', and 'Cookie Preferences'. The version number 'r31.1' is also visible.

## TABLE-LOAN

Oracle Live SQL

livesql.oracle.com/next/

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[ SQL Worksheet ]\*

1

2

3

4

5

6

7

CREATE TABLE LOANS (  
  LOANID NUMBER PRIMARY KEY,  
  CUSTOMERID NUMBER  
  REFERENCES CUSTOMERS ( CUSTOMERID ),  
  INTERESTRATE NUMBER(5, 2),  
  DUEDATE DATE  
);

Query result

Script output

DBMS output

Explain Plan

SQL history

Elapsed: 00:00:00.018

SQL> CREATE TABLE LOANS (  
  LOANID NUMBER PRIMARY KEY,  
  CUSTOMERID NUMBER  
  REFERENCES CUSTOMERS ( CUSTOMERID ),...  
Show more...

Table LOANS created.  
Elapsed: 00:00:00.015

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

## DATA INSERTION:-

## CUSTOMERS

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[ SQL Worksheet ]\*

1

2

3

4

5

6

7

8

9

10

11

12

13

14

INSERT ALL  
--INSERT INTO Customers VALUES (1, 'Alice', 65, 12000, 'N');  
--INSERT INTO Customers VALUES (2, 'Bob', 45, 9500, 'N');  
--INSERT INTO Customers VALUES (3, 'Charlie', 70, 8000, 'N');  
INTO Customers VALUES (4, 'Rahul', 61, 15000, 'N')  
INTO Customers VALUES (5, 'Priya', 58, 9800, 'N')  
INTO Customers VALUES (6, 'Suresh', 75, 11000, 'N')  
INTO Customers VALUES (7, 'Meena', 30, 13000, 'N')  
INTO Customers VALUES (8, 'Anjali', 62, 8900, 'N')  
INTO Customers VALUES (9, 'Ravi', 40, 5000, 'N')  
INTO Customers VALUES (10, 'Neha', 66, 14000, 'N')  
SELECT \* FROM DUAL;  
SELECT \* FROM CUSTOMERS;

Query result

Script output

DBMS output

Explain Plan

SQL history

Download

Execution time: 0.012 seconds

	CUSTOMERID	NAME	AGE	BALANCE	ISVIP
1	2	Bob	45	9500	N
2	4	Rahul	61	15000	N
3	5	Priya	58	9800	N
4	6	Suresh	75	11000	N
5	7	Meena	30	13000	N
6	8	Anjali	62	8900	N

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

LOANS:

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

NavigatorFiles

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[SQL Worksheet]\*

1INSERT ALL

2INTO Loans VALUES (101, 1, 7.5, SYSDATE + 20)

3INTO Loans VALUES (102, 2, 6.0, SYSDATE + 40)

4INTO Loans VALUES (103, 3, 5.5, SYSDATE + 10)

5INTO Loans VALUES (104, 4, 8.0, SYSDATE + 5)

6INTO Loans VALUES (105, 5, 7.0, SYSDATE + 31)

7INTO Loans VALUES (106, 6, 6.5, SYSDATE + 15)

8INTO Loans VALUES (107, 7, 5.0, SYSDATE + 25)

9INTO Loans VALUES (108, 8, 8.5, SYSDATE + 10)

10INTO Loans VALUES (109, 9, 9.0, SYSDATE + 45)

11INTO Loans VALUES (110, 10, 7.8, SYSDATE + 1)

12SELECT \* FROM DUAL;

13SELECT \* FROM LOANS;

Query result

Script output

DBMS output

Explain Plan

SQL history

Download

Execution time: 0.081 seconds

	LOANID	CUSTOMERID	INTERESTRATE	DUEDATE
1	101	1	7.5	7/14/2025, 2:26:03
2	102	2	6	8/3/2025, 2:26:03 P
3	103	3	5.5	7/4/2025, 2:26:03 P
4	104	4	8	6/29/2025, 2:26:03
5	105	5	7	7/25/2025, 2:26:03
6	106	6	6.5	7/9/2025, 2:26:03 P

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

SCENARIO -1

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

NavigatorFiles

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[SQL Worksheet]\*

1BEGIN

2FOR cust IN (SELECT CustomerID FROM Customers WHERE Age > 60) LOOP

3UPDATE Loans

4SET InterestRate = InterestRate - 1

5WHERE CustomerID = cust.CustomerID;

6END LOOP;

7COMMIT;

8END;

9

Query result

Script output

DBMS output

Explain Plan

SQL history

Elapsed: 00:00:00.024

SQL> BEGIN

FOR cust IN (SELECT CustomerID FROM Customers WHERE Age > 60) LOOP

UPDATE Loans

SET InterestRate = InterestRate - 1...

Show more...

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.016

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

## SCENARIO-2

**Live SQL** > Worksheet Library 23al Return to Live SQL Classic Help and Feedback Sign Out

**Navigator** Files

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[SQL Worksheet]\*

```
1 BEGIN
2   FOR cust IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP
3     UPDATE Customers
4       SET IsVIP = 'Y'
5     WHERE CustomerID = cust.CustomerID;
6   END LOOP;
7   COMMIT;
8 END;
```

Query result Script output DBMS output Explain Plan SQL history

Elapsed: 00:00:00.016

SQL> BEGIN  
FOR cust IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP  
UPDATE Customers  
SET IsVIP = 'Y'...  
Show more...

PL/SQL procedure successfully completed.  
Elapsed: 00:00:00.012

About Oracle Contact Us Legal Notices Terms and Conditions Your Privacy Rights Delete Your Live SQL Account Cookie Preferences  
Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

## SCENARIO-3

**Live SQL** > Worksheet Library 23al Return to Live SQL Classic Help and Feedback Sign Out

**Navigator** Files

My Schema

Tables

Search objects

CUSTOMERS

LOANS

[SQL Worksheet]\*

```
1 -->
2 |<_name Customers.Name%TYPE;
3 BEGIN
4   FOR loan_rec IN (
5     SELECT L.LoanID, C.Name, L.DueDate
6     FROM Loans L
7     JOIN Customers C ON L.CustomerID = C.CustomerID
8     WHERE L.DueDate BETWEEN SYSDATE AND SYSDATE + 30
9   ) LOOP
10    DBMS_OUTPUT.PUT_LINE('Reminder: Dear ' || loan_rec.Name ||
11                          ', your loan (ID: ' || loan_rec.LoanID ||
12                          ') is due on ' || TO_CHAR(loan_rec.DueDate, 'DD-MON-YYYY'));
13   END LOOP;
14 END;
```

Query result Script output DBMS output Explain Plan SQL history

SHOW REMINDERS...

Reminder: Dear Rahul, your loan (ID: 104) is due on 29-JUN-2025  
Reminder: Dear Suresh, your loan (ID: 106) is due on 09-JUL-2025  
Reminder: Dear Meena, your loan (ID: 107) is due on 19-JUL-2025  
Reminder: Dear Anjali, your loan (ID: 108) is due on 04-JUL-2025  
Reminder: Dear Neha, your loan (ID: 110) is due on 25-JUN-2025  
Reminder: Dear Charlie, your loan (ID: 103) is due on 04-JUL-2025  
Reminder: Dear Alice, your loan (ID: 101) is due on 14-JUL-2025

PL/SQL procedure successfully completed.  
Elapsed: 00:00:00.020

About Oracle Contact Us Legal Notices Terms and Conditions Your Privacy Rights Delete Your Live SQL Account Cookie Preferences  
Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1

## Exercise 3: Stored Procedures

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

- **Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

- **Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

**Scenario 3:** Customers should be able to transfer funds between their accounts.

- **Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

### SCENARIO-1

#### TABLE CREATION: SAVINGACC

The screenshot displays the Oracle Live SQL web interface. On the left, the 'Navigator' pane shows a tree view with 'My Schema' selected, containing 'Tables'. Below this is a search bar and a list of objects: CUSTOMERS, LOANS, and SAVINGACC. The main editor area is titled 'SAVINGS.sql\*' and contains the following SQL code:

```
1 CREATE TABLE SAVINGACC(  
2   ACCOUNTID NUMBER PRIMARY KEY,  
3   CUSTOMERNAME VARCHAR2(100),  
4   BALANCE NUMBER(10,2)  
5 );
```

Below the editor, the 'Script output' tab is active, showing the execution results:

```
SQL> CREATE TABLE SAVINGACC(  
  ACCOUNTID NUMBER PRIMARY KEY,  
  CUSTOMERNAME VARCHAR2(100),  
  BALANCE NUMBER(10,2)...  
Show more...
```

Below the output, a message states: 'Table SAVINGACC created.' and 'Elapsed: 00:00:00.017'.

At the bottom of the interface, there is a footer with links: 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Delete Your Live SQL Account', and 'Cookie Preferences'. The copyright notice reads: 'Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved.' and the version is 'r31.1'.

# DATA INSERTION

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

Navigator

Files

My Schema

Tables

Search objects

CUSTOMERS

LOANS

SAVINGSACC

SAVINGS.sql\*

1

2

3

4

5

6

7

8

9

10

11

INSERT ALL

INTO SAVINGSACC VALUES (5678, 'KUMAR', 100943)

INTO SAVINGSACC VALUES (8765, 'LAKSHMI', 907500)

INTO SAVINGSACC VALUES (1234, 'PRIYA', 6384000)

INTO SAVINGSACC VALUES (4321, 'LOGAN', 2600)

INTO SAVINGSACC VALUES (2345, 'SANDEEP', 4784000)

INTO SAVINGSACC VALUES (5432, 'RISHI', 957000)

INTO SAVINGSACC VALUES (3456, 'KUNAL', 1000)

INTO SAVINGSACC VALUES (6543, 'KUNJUN', 100)

SELECT \* FROM DUAL;

Query result

Script output

DBMS output

Explain Plan

SQL history

SQL> INSERT ALL

INTO SAVINGSACC VALUES (5678, 'KUMAR', 100943)

INTO SAVINGSACC VALUES (8765, 'LAKSHMI', 907500)

INTO SAVINGSACC VALUES (1234, 'PRIYA', 6384000)...

Show more...

8 rows inserted.

Elapsed: 00:00:00.020

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014-2025 Oracle and/or its affiliates. All rights reserved.

r51.1

# OUTPUT:

Live SQL

Worksheet

Library

23al

Return to Live SQL Classic

Help and Feedback

Sign Out

Navigator

Files

My Schema

Tables

Search objects

CUSTOMERS

LOANS

SAVINGSACC

SAVINGS.sql\*

1

2

3

4

5

6

7

8

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE SAVINGSACC

SET Balance = Balance + (Balance \* 0.01);

COMMIT;

END;

/

Query result

Script output

DBMS output

Explain Plan

SQL history

SQL> BEGIN

ProcessMonthlyInterest;

END;

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.007

About Oracle

Contact Us

Legal Notices

Terms and Conditions

Your Privacy Rights

Delete Your Live SQL Account

Cookie Preferences

Copyright © 2014-2025 Oracle and/or its affiliates. All rights reserved.

r51.1

## SCENARIO -2

### TABLE CREATION:- EMPLOYEES

The screenshot shows the Oracle Live SQL interface. On the left, the 'Navigator' pane shows 'My Schema' and 'Tables'. The main editor displays the SQL script to create the EMPLOYEES table:

```
1 CREATE TABLE EMPLOYEES(  
2 EMPID NUMBER PRIMARY KEY,  
3 EMPNAME VARCHAR2(100),  
4 DEPARTMENT VARCHAR2(50),  
5 SALARY NUMBER(10,2)  
6 );
```

Below the editor, the 'Script output' pane shows the execution result:

```
SQL> CREATE TABLE EMPLOYEES(  
EMPID NUMBER PRIMARY KEY,  
EMPNAME VARCHAR2(100),  
DEPARTMENT VARCHAR2(50),...  
Show more...  
  
Table EMPLOYEES created.
```

The footer includes links for 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Delete Your Live SQL Account', and 'Cookie Preferences'. The version is r31.1.

### DATA INSERTION

The screenshot shows the Oracle Live SQL interface with the 'CUST\_TABLE.sql' file open. The main editor displays the SQL script to insert data into the EMPLOYEES table:

```
1 INSERT ALL  
2 INTO Employees VALUES (105, 'Amit', 'IT', 58000)  
3 INTO Employees VALUES (106, 'Sneha', 'Sales', 47000)  
4 INTO Employees VALUES (107, 'Karan', 'Finance', 62000)  
5 INTO Employees VALUES (108, 'Neha', 'HR', 51000)  
6 INTO Employees VALUES (109, 'Ramesh', 'IT', 59000)  
7 INTO Employees VALUES (110, 'Priya', 'Finance', 55000)  
8 INTO Employees VALUES (111, 'Anil', 'Marketing', 48000)  
9 INTO Employees VALUES (112, 'Pooja', 'Marketing', 49500)  
10 INTO Employees VALUES (113, 'Deepak', 'Sales', 53000)  
11 INTO Employees VALUES (114, 'Tanya', 'HR', 52000)  
12  
13 SELECT * FROM DUAL;
```

Below the editor, the 'Script output' pane shows the execution result:

```
SQL> INSERT ALL  
INTO Employees VALUES (105, 'Amit', 'IT', 58000)  
INTO Employees VALUES (106, 'Sneha', 'Sales', 47000)  
INTO Employees VALUES (107, 'Karan', 'Finance', 62000)...  
Show more...  
  
10 rows inserted.
```

The footer includes links for 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Delete Your Live SQL Account', and 'Cookie Preferences'. The version is r31.1.

## OUTPUT:

The screenshot displays the Live SQL web application interface. On the left, the 'Navigator' pane shows a tree view of the database schema with folders for CUSTOMERS, EMPLOYEES, LOANS, and SAVINGSACC. The main editor area, titled '[SQL Worksheet]\*', contains an SQL script for creating or replacing a procedure named 'UpdateEmployeeBonus'. The script defines parameters for department name and bonus percentage, and includes logic to update employee salaries based on their department and bonus percentage. Below the editor, the 'Script output' tab is active, showing the successful execution of the procedure: 'SQL> CREATE OR REPLACE PROCEDURE UPDTEEMPLOYEEBONUS(...) ... Show more...' and 'Procedure UPDTEEMPLOYEEBONUS compiled'. The footer includes links for About Oracle, Contact Us, Legal Notices, Terms and Conditions, Your Privacy Rights, Delete Your Live SQL Account, and Cookie Preferences, along with the URL 'https://www.oracle.com/livesql/' and the version 'r31.1'.

```
1 CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (  
2   dept_name IN VARCHAR2,  
3   bonus_percent IN NUMBER  
4 ) IS  
5 BEGIN  
6   UPDATE Employees  
7     SET Salary = Salary + (Salary * bonus_percent / 100)  
8     WHERE Department = dept_name;  
9   COMMIT;  
10 END;  
11 /  
12
```

SQL> CREATE OR REPLACE PROCEDURE UPDTEEMPLOYEEBONUS(  
 DEPT\_NAME IN VARCHAR2,  
 BONUS\_PERCENT IN NUMBER  
)...  
Show more...

Procedure UPDTEEMPLOYEEBONUS compiled

[About Oracle](#) [Contact Us](#) [Legal Notices](#) [Terms and Conditions](#) [Your Privacy Rights](#) [Delete Your Live SQL Account](#) [Cookie Preferences](#)  
<https://www.oracle.com/livesql/> 25 Oracle and/or its affiliates All rights reserved. r31.1

## SCENARIO-3

### TABLE CREATION: ACCOUNTS

The screenshot displays the Live SQL web application interface. On the left, the 'Navigator' pane shows a tree view of the database schema with folders for ACCOUNTS, CUSTOMERS, EMPLOYEES, LOANS, and SAVINGSACC. The main editor area, titled 'SC-3\_1.sql', contains an SQL script for creating a table named 'Accounts'. The script defines columns for AccountID (primary key), CustomerName, and Balance. Below the editor, the 'Script output' tab is active, showing the successful execution of the table creation: 'SQL> CREATE TABLE Accounts ( AccountID NUMBER PRIMARY KEY, CustomerName VARCHAR2(100), Balance NUMBER(10, 2)... Show more...' and 'Table ACCOUNTS created.' The footer includes links for About Oracle, Contact Us, Legal Notices, Terms and Conditions, Your Privacy Rights, Delete Your Live SQL Account, and Cookie Preferences, along with the copyright notice 'Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved.' and the version 'r31.1'.

```
1 CREATE TABLE Accounts (  
2   AccountID NUMBER PRIMARY KEY,  
3   CustomerName VARCHAR2(100),  
4   Balance NUMBER(10, 2)  
5 );
```

SQL> CREATE TABLE Accounts (  
 AccountID NUMBER PRIMARY KEY,  
 CustomerName VARCHAR2(100),  
 Balance NUMBER(10, 2)...  
Show more...

Table ACCOUNTS created.

[About Oracle](#) [Contact Us](#) [Legal Notices](#) [Terms and Conditions](#) [Your Privacy Rights](#) [Delete Your Live SQL Account](#) [Cookie Preferences](#)  
Copyright © 2014, 2025 Oracle and/or its affiliates All rights reserved. r31.1



## DATA INSERTION:

The screenshot shows the Live SQL interface with a script titled "SC\_3-2.sql". The script contains an `INSERT ALL` statement with 10 rows of data into the `Accounts` table. The data includes account IDs from 2001 to 2010, names, and balances. The script ends with `SELECT * FROM DUAL;`. The "Script output" tab is active, showing the execution result: "10 rows inserted."

```
1  INSERT ALL
2    INTO Accounts VALUES (2001, 'Rajesh', 12000)
3    INTO Accounts VALUES (2002, 'Kavya', 7500)
4    INTO Accounts VALUES (2003, 'Manoj', 9800)
5    INTO Accounts VALUES (2004, 'Divya', 13400)
6    INTO Accounts VALUES (2005, 'Arjun', 9400)
7    INTO Accounts VALUES (2006, 'Ansha', 11200)
8    INTO Accounts VALUES (2007, 'Nikhil', 8700)
9    INTO Accounts VALUES (2008, 'Ishita', 6300)
10   INTO Accounts VALUES (2009, 'Vikram', 15000)
11   INTO Accounts VALUES (2010, 'Ritika', 4500)
12  SELECT * FROM DUAL;
```

Query result | **Script output** | DBMS output | Explain Plan | SQL history

SQL> INSERT ALL  
INTO Accounts VALUES (2001, 'Rajesh', 12000)  
INTO Accounts VALUES (2002, 'Kavya', 7500)  
INTO Accounts VALUES (2003, 'Manoj', 9800)...

10 rows inserted.

## OUTPUT:

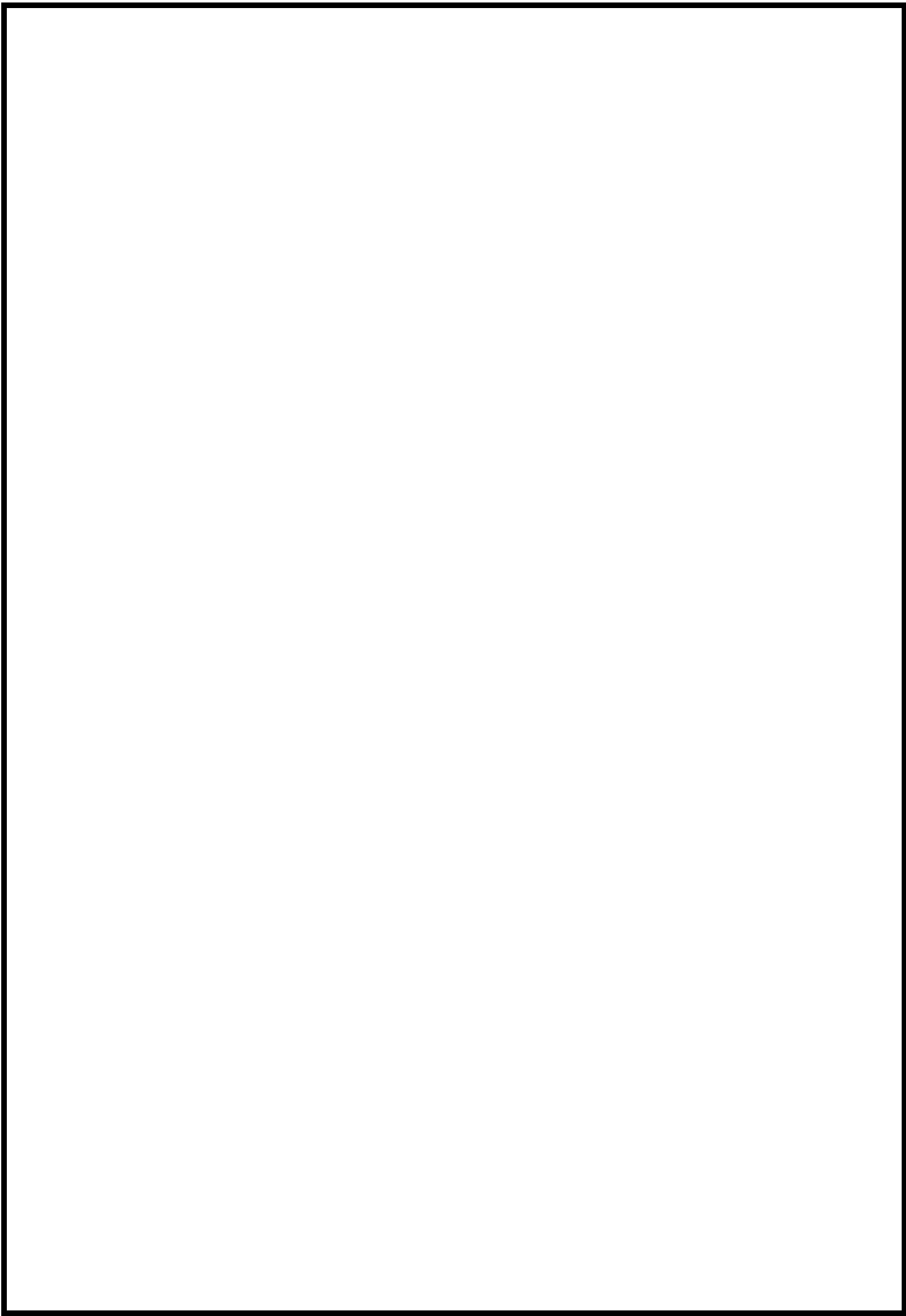
The screenshot shows the Live SQL interface with a script titled "[SQL Worksheet]\*". The script creates a stored procedure named `TransferFunds`. It takes three parameters: `from_acc` (number), `to_acc` (number), and `amount` (number). The procedure checks if the `from_acc` balance is sufficient. If not, it raises an `insufficient_balance` exception. If sufficient, it updates the `from_acc` balance by subtracting the `amount` and the `to_acc` balance by adding the `amount`. The script ends with `COMMIT;`. The "Script output" tab is active, showing the compilation message: "Procedure TRANSFERFUNDS compiled".

```
1  CREATE OR REPLACE PROCEDURE TransferFunds (
2    from_acc IN NUMBER,
3    to_acc IN NUMBER,
4    amount IN NUMBER
5  ) IS
6    insufficient_balance EXCEPTION;
7    v_balance NUMBER;
8  BEGIN
9    SELECT Balance INTO v_balance FROM Accounts WHERE AccountID = from_acc;
10
11    IF v_balance < amount THEN
12      RAISE insufficient_balance;
13    END IF;
14
15    UPDATE Accounts SET Balance = Balance - amount WHERE AccountID = from_acc;
16    UPDATE Accounts SET Balance = Balance + amount WHERE AccountID = to_acc;
17
18    COMMIT;
```

Query result | **Script output** | DBMS output | Explain Plan | SQL history

to\_acc IN NUMBER,  
amount IN NUMBER...

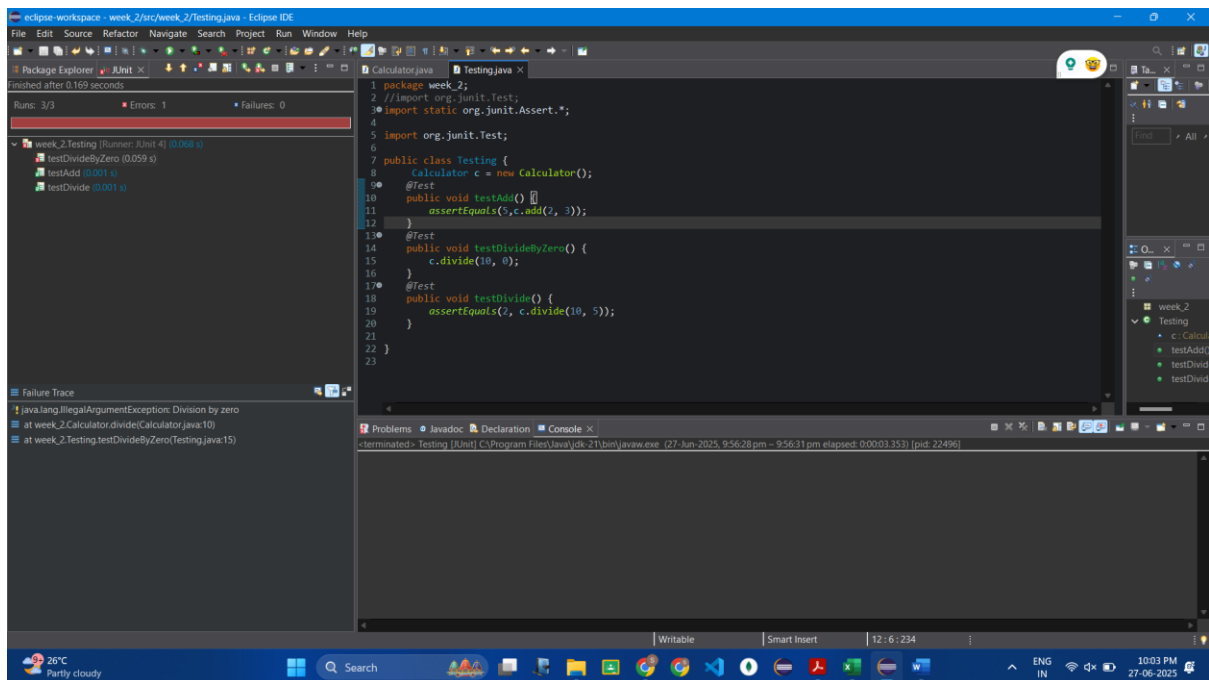
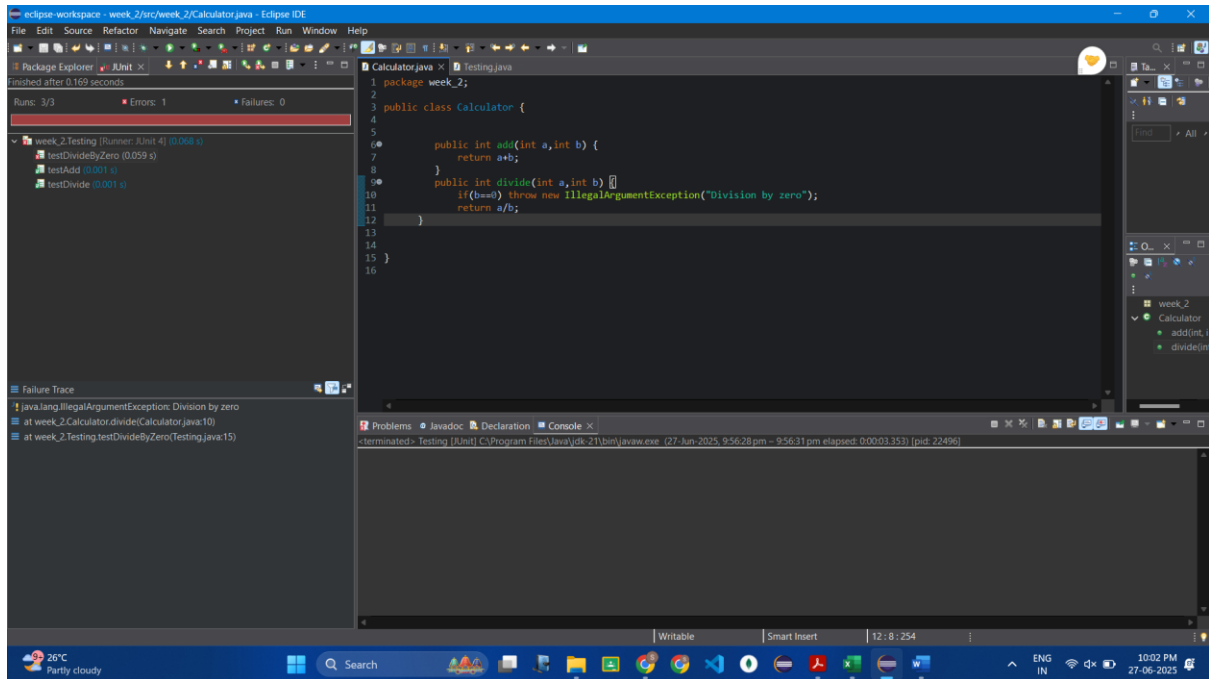
Procedure TRANSFERFUNDS compiled  
Elapsed: 00:00:00.018



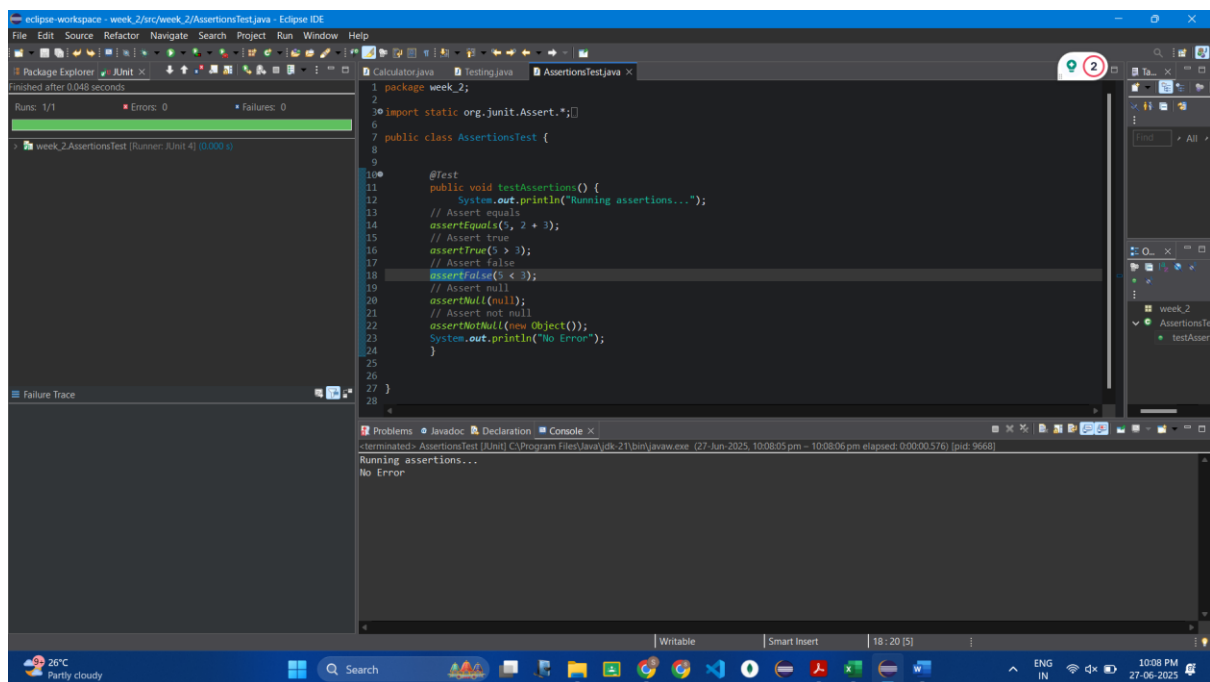
## WEEK-2

### MODULE-4 TDD using JUnit5 and Mockito

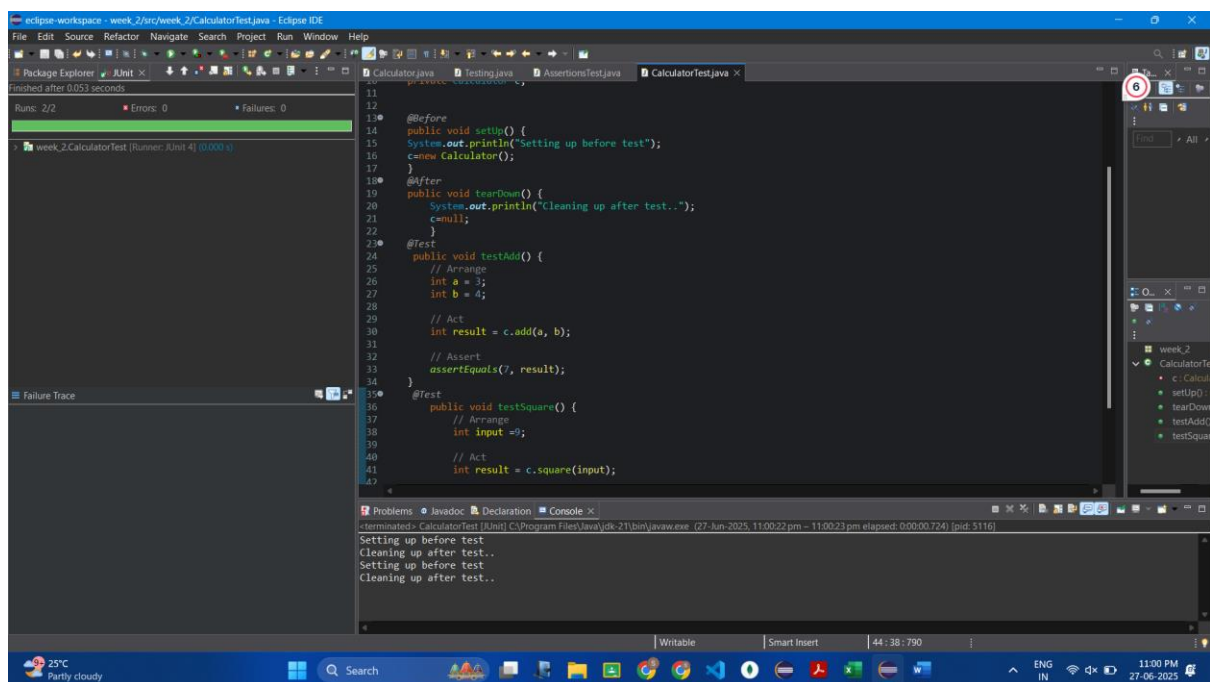
#### Exercise 1: Setting Up Junit



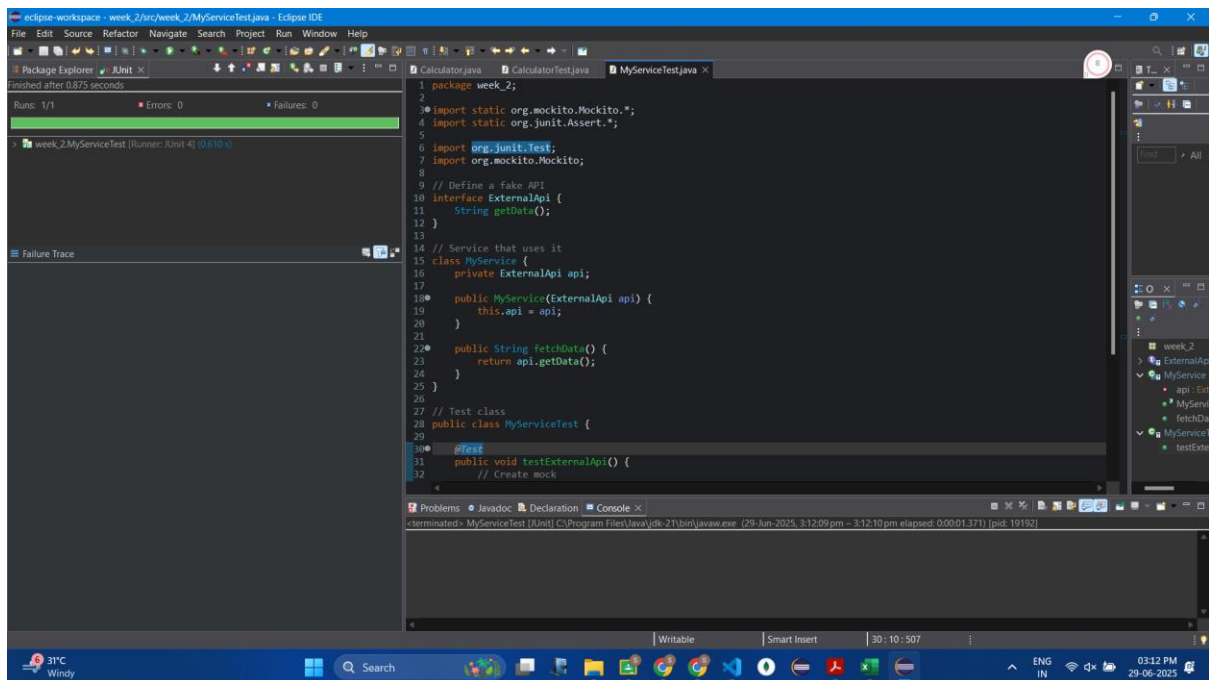
## Exercise 3: Assertions in Junit



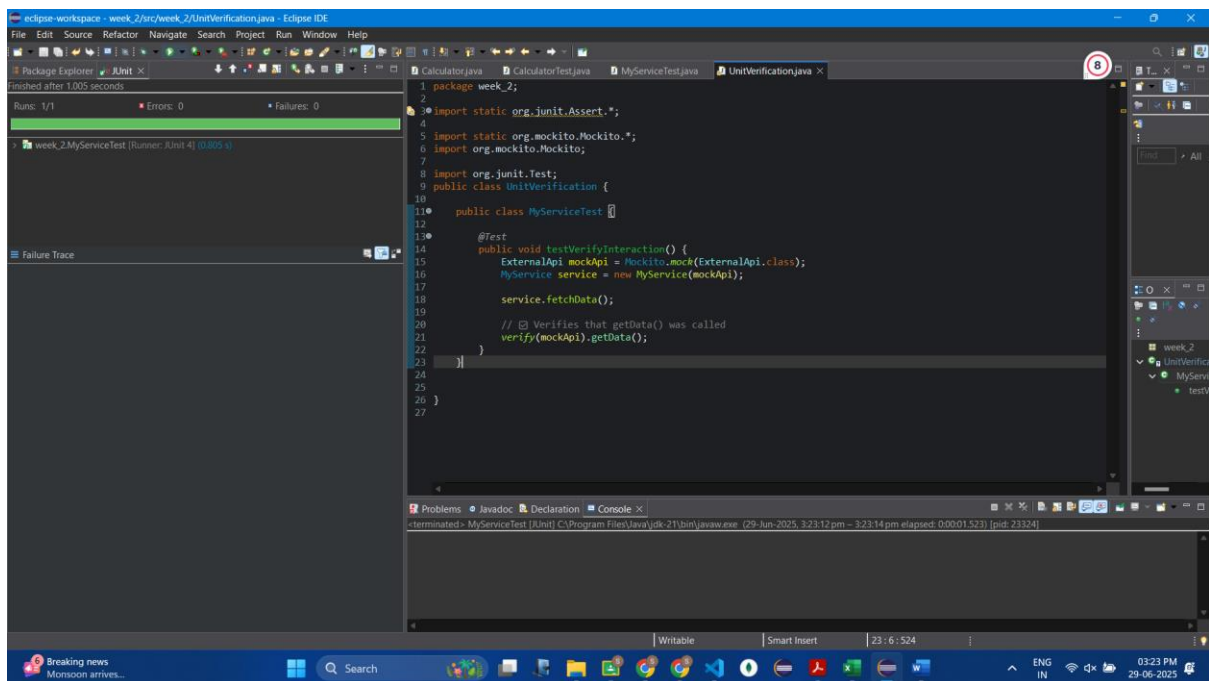
## Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit



## Exercise 1: Mocking and Stubbing



## Exercise 2: Verifying Interactions



## Exercise 1: Logging Error Messages and Warning Levels

