EXERCISE 1

AIM: Trigger code is to set a default salary of 3000 for new employees if the salary field is not provided (i.e., is NULL) during an INSERT operation on the employee table.

```
CREATE TABLE employee (
    id NUMBER PRIMARY KEY,
    name VARCHAR2(100) NOT NULL,
    salary NUMBER(10, 2),
    department VARCHAR2(50)
)

INSERT INTO employee (id, name, salary, department)
VALUES (1, 'John Doe', 50000, 'HR')

INSERT INTO employee (id, name, salary, department)
VALUES (2, 'Jane Smith', 60000, 'Finance')

INSERT INTO employee (id, name, salary, department)
VALUES (3, 'Alice Johnson', 55000, 'Engineering')

INSERT INTO employee (id, name, salary, department)
VALUES (4, 'Bob Brown', 70000, 'Marketing')

INSERT INTO employee (id, name, salary, department)
VALUES (5, 'Charlie White', 48000, 'Sales')
```

```
-------------------------------------------Output:------------------------------------
```

```
ID    NAME           SALARY  DEPARTMENT
1     John Doe       50000   HR
2     Jane Smith     60000   Finance
3     Alice Johnson  55000   Engineering
4     Bob Brown      70000   Marketing
5     Charlie White  48000   Sales
```

```
create or replace trigger trg_before_insert
before insert on employee
for each row
begin
if: NEW.salary is null then
       : NEW.salary:=3000;
       end if;
end;
```

```
-------------------------------------------Output:------------------------------------
```

Trigger created

Exercise: 2

AIM: To perform arithmetic operations (addition, subtraction, multiplication, and division) using **both local and stored functions** within a procedure called perform_operations

--STORED PROCEDURE-----------------------------------

```
create or replace function add_numbers(num1 in number, num2 in number)
        return number is
        begin
                return num1 + num2;
        end add_numbers;

create or replace function subtract_numbers(num1 in number, num2 in number)
        return number is
    begin
         return num1 - num2;
        end subtract_numbers;

create or replace function multiply_numbers(num1 in number, num2 in number)
return number is
begin
        return num1*num2;
end multiply_numbers;

create or replace function divide_numbers(num1 in number, num2 in number)
return number is
begin
        if num2=0 then
    return null;
        else
         return num1/num2;
        end if;
end divide_numbers;
```

--LOCAL PROCEDURE--------------------------------

```
CREATE OR replace procedure perform_operations(
    num1 in number,
    num2 in number
) is
function local_add(n1 in number,n2 in number)
return number is
begin
        return n1 + n2;
end local_add;

function local_subtract(n1 in number,n2 in number)
return number is
begin
        return n1 - n2;
end local_subtract;

function local_multiply(n1 in number,n2 in number)
return number is
begin
        return n1 * n2;
end local_multiply;
```

```
function local_divide(n1 in number,n2 in number)
return number is
begin
        if n2 = 0 then
return NULL;
else
       return n1 / n2;
end if;
end local_divide;
```

-----------------------------------CALLING FUNCTIONS-----------------------------

```
BEGIN
DBMS_OUTPUT.PUT_LINE('Addition(local):' || local_add(num1,num2));
DBMS_OUTPUT.PUT_LINE('Subtraction(local):' || local_subtract(num1,num2));
DBMS_OUTPUT.PUT_LINE('Multiplication(local):' || local_multiply(num1,num2));
DBMS_OUTPUT.PUT_LINE('Division(local):' || local_divide(num1,num2));

DBMS_OUTPUT.PUT_LINE('Addition(stored):' || add_numbers(num1,num2));
DBMS_OUTPUT.PUT_LINE('Subtraction(stored):' || subtract_numbers(num1,num2));
DBMS_OUTPUT.PUT_LINE('Multiplication(stored):' || multiply_numbers(num1,num2));
DBMS_OUTPUT.PUT_LINE('Division(stored):' || divide_numbers(num1,num2));
END perform_operations;
```

---Output:-------------------------------------

```
PROCEDURE CREATED
```

```
exec perform_operations(10,5)
```

---Output:-------------------------------------

```
Statement processed.
Addition(local):15
Subtraction(local):5
Multiplication(local):50
Division(local):2
Addition(stored):15
Subtraction(stored):5
Multiplication(stored):50
Division(stored):2
```