Exercise 4: Functions

Scenario 1:

```
 CREATE OR REPLACE FUNCTION CalculateAge (
   p_date_of_birth IN DATE
) RETURN NUMBER IS
   v_age NUMBER;
BEGIN
   v_age := TRUNC((SYSDATE - p_date_of_birth) / 365.25);
   RETURN v_age;
END CalculateAge;
/
```

Scenario 2:

```
CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (
   p_loan_amount IN NUMBER,
   p_interest_rate IN NUMBER,
   p_duration_years IN NUMBER
) RETURN NUMBER IS
   v_monthly_interest_rate NUMBER;
   v_number_of_payments NUMBER;
   v_monthly_installment NUMBER;
BEGIN
   v_monthly_interest_rate := p_interest_rate / 12 / 100;
   v_number_of_payments := p_duration_years * 12;

   IF v_monthly_interest_rate = 0 THEN
      v_monthly_installment := p_loan_amount / v_number_of_payments;
   ELSE
      v_monthly_installment := p_loan_amount * v_monthly_interest_rate /
                        (1 - POWER(1 + v_monthly_interest_rate, -v_number_of_payments));
   END IF;
   RETURN v_monthly_installment;
END CalculateMonthlyInstallment;
/
```

Scenario 3:

```
CREATE OR REPLACE FUNCTION HasSufficientBalance (
   p_account_id IN NUMBER,
   p_amount IN NUMBER
) RETURN BOOLEAN IS
   v_balance NUMBER;
BEGIN
   SELECT balance INTO v_balance
   FROM accounts
   WHERE account_id = p_account_id;

   IF v_balance >= p_amount THEN
      RETURN TRUE;
   ELSE
      RETURN FALSE;
   END IF;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RETURN FALSE;
   WHEN OTHERS THEN
      RETURN FALSE;
END HasSufficientBalance;
/
```