# Experiment 6

**Program 6.1**:write a pl/sql program to swap two numbers without taking third variable

```
DECLARE
    a NUMBER := 10;
    b NUMBER := 20;
BEGIN
    a := a + b;
    b := a - b;
    a := a - b;
    DBMS_OUTPUT.PUT_LINE('Swapped values: a=' || a || ', b=' || b);
END;
/
```

**Program 6.2:**write a pl/sql program to swap two numbers by taking third variable

```
DECLARE
    a NUMBER := 10;
    b NUMBER := 20;
    temp NUMBER;
BEGIN
    temp := a;
    a := b;
    b := temp;
    DBMS_OUTPUT.PUT_LINE('Swapped values: a=' || a || ', b=' || b);
END;
/
```

**Program 6.3:**Write a pl/sql program to find the largest of two numbers

```
DECLARE
    a NUMBER := 15;
    b NUMBER := 25;
    largest NUMBER;
BEGIN
    IF a > b THEN
        largest := a;
    ELSE
        largest := b;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Largest number: ' || largest);
END;
/
```

**Program 6.4:**write a pl/sql program to find the total and average of 6 subjects and display the grade

```
DECLARE
    s1 NUMBER := 78;
    s2 NUMBER := 85;
    s3 NUMBER := 90;
    s4 NUMBER := 72;
    s5 NUMBER := 88;
    s6 NUMBER := 91;
    total NUMBER;
    avg NUMBER;
    grade CHAR(1);
```

```
BEGIN
    total := s1 + s2 + s3 + s4 + s5 + s6;
    avg := total / 6;

    IF avg >= 90 THEN grade := 'A';
    ELSIF avg >= 80 THEN grade := 'B';
    ELSIF avg >= 70 THEN grade := 'C';
    ELSE grade := 'D';
    END IF;

    DBMS_OUTPUT.PUT_LINE('Total: ' || total || ', Average: ' || avg || ',
Grade: ' || grade);
END;
/
```

**Program 6.5:** Write a pl/sql program to find the sum of digits in a given number

```
DECLARE
    num NUMBER := 1234;
    sum_digits NUMBER := 0;
    rem NUMBER;
BEGIN
    WHILE num > 0 LOOP
        rem := MOD(num, 10);
        sum_digits := sum_digits + rem;
        num := TRUNC(num / 10);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Sum of digits: ' || sum_digits);
END;
/
```

**Program 6.6:**write a pl/sql program to display the number in reverse order

```
DECLARE
    num NUMBER := 1234;
    rev NUMBER := 0;
    rem NUMBER;
BEGIN
    WHILE num > 0 LOOP
        rem := MOD(num, 10);
        rev := rev * 10 + rem;
        num := TRUNC(num / 10);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Reversed Number: ' || rev);
END;
/
```

**Program 6.7:**Write a pl/sql program to check whether the given number is prime or not

```
DECLARE
    num NUMBER := 29;
    i NUMBER := 2;
    is_prime BOOLEAN := TRUE;
BEGIN
    WHILE i <= SQRT(num) LOOP
        IF MOD(num, i) = 0 THEN
```

```
            is_prime := FALSE;
            EXIT;
        END IF;
        i := i + 1;
    END LOOP;
    IF is_prime THEN
        DBMS_OUTPUT.PUT_LINE(num || ' is a Prime Number');
    ELSE
        DBMS_OUTPUT.PUT_LINE(num || ' is Not a Prime Number');
    END IF;
END;
/
```

**Program 6.8:**Write a pl/sql program to find the factorial of a given number

```
DECLARE
    num NUMBER := 5;
    fact NUMBER := 1;
    i NUMBER;
BEGIN
    FOR i IN 1..num LOOP
        fact := fact * i;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Factorial of ' || num || ' is ' || fact);
END;
/
```

**Program 6.9**:write a pl/sql code block to calculate the area of a circle for a value of radius varying from 3 to 7.Store the radius and the corresponding values of calculated area in an empty table named areas ,consisting of two columns radius & area

```
CREATE TABLE areas (
    radius NUMBER,
    area NUMBER
);

DECLARE
    r NUMBER;
    a NUMBER;
BEGIN
    FOR r IN 3..7 LOOP
        a := 3.14159 * r * r;
        INSERT INTO areas VALUES (r, a);
    END LOOP;
    COMMIT;
END;
/
```

**Program 6.10:**write a pl/sql code block that will accept an account number from the user,check if the users balance is less than minimum balance,only then deduct rs.100/- from the balance.this process is fired on the acct table.

```
CREATE TABLE acct (
    acc_no NUMBER PRIMARY KEY,
    balance NUMBER,
    min_balance NUMBER
);
```

```
INSERT INTO acct VALUES (101, 500, 1000);
INSERT INTO acct VALUES (102, 1500, 1000);
COMMIT;

DECLARE
    v_balance NUMBER;
    v_min_balance NUMBER;
BEGIN
    FOR rec IN (SELECT acc_no, balance, min_balance FROM acct) LOOP
        IF rec.balance < rec.min_balance THEN
            UPDATE acct SET balance = balance - 100 WHERE acc_no =
rec.acc_no;
        END IF;
    END LOOP;
    COMMIT;
END;
/
```