Basic Looping and Conditionals

1. **Question:** Write a PL/SQL block that prints the numbers from 1 to 10 using a `FOR` loop.

   SQL

   ```
   BEGIN
     FOR i IN 1..10 LOOP
       DBMS_OUTPUT.PUT_LINE(i);
     END LOOP;
   END;
   /
   ```

2. **Question:** Write a PL/SQL block that prints the even numbers from 1 to 20 using a `WHILE` loop.

   SQL

   ```
   DECLARE
     i NUMBER := 2;
   BEGIN
     WHILE i <= 20 LOOP
       DBMS_OUTPUT.PUT_LINE(i);
       i := i + 2;
     END LOOP;
   END;
   /
   ```

3. **Question:** Write a PL/SQL block that prints "Odd" or "Even" for numbers 1 to 5 using an `IF` statement within a loop.

   SQL

   ```
   BEGIN
     FOR i IN 1..5 LOOP
       IF MOD(i, 2) = 0 THEN
         DBMS_OUTPUT.PUT_LINE(i || ' is Even');
       ELSE
         DBMS_OUTPUT.PUT_LINE(i || ' is Odd');
       END IF;
     END LOOP;
   END;
   /
   ```

More Complex Scenarios

4. **Question:** Write a PL/SQL block that calculates the factorial of a given number (e.g., 5).

   SQL

   ```
   DECLARE
     n NUMBER := 5;
     factorial NUMBER := 1;
   ```

```
BEGIN
  FOR i IN 1..n LOOP
    factorial := factorial * i;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Factorial of ' || n || ' is: ' || factorial);
END;
/
```

5. **Question:** Write a PL/SQL block that finds the largest number in a given set of numbers (e.g., 10, 5, 20, 8).

   SQL

```
DECLARE
  TYPE num_array IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
  numbers num_array;
  largest NUMBER;
BEGIN
  numbers(1) := 10;
  numbers(2) := 5;
  numbers(3) := 20;
  numbers(4) := 8;

  largest := numbers(1);
  FOR i IN 2..numbers.COUNT LOOP
    IF numbers(i) > largest THEN
      largest := numbers(i);
    END IF;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Largest number is: ' || largest);
END;
/
```

6. **Question:** Write a PL/SQL block that prints the Fibonacci sequence up to a given number of terms (e.g., 6 terms).

   SQL

```
DECLARE
  terms NUMBER := 6;
  a NUMBER := 0;
  b NUMBER := 1;
  c NUMBER;
BEGIN
  DBMS_OUTPUT.PUT_LINE(a);
  DBMS_OUTPUT.PUT_LINE(b);
  FOR i IN 3..terms LOOP
    c := a + b;
    DBMS_OUTPUT.PUT_LINE(c);
    a := b;
    b := c;
  END LOOP;
END;
/
```

7. **Question:** Write a PL/SQL block that uses a `LOOP ... EXIT WHEN` structure to sum numbers entered until a negative number is entered.

SQL

```
DECLARE
  num NUMBER;
  sum NUMBER := 0;
BEGIN
  LOOP
    num := &Enter_Number;
    EXIT WHEN num < 0;
    sum := sum + num;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Sum: ' || sum);
END;
/
```

8. **Question:** Create a PL/SQL block that determines if a provided number is a prime number.

SQL

```
DECLARE
  num NUMBER := &Enter_Number;
  is_prime BOOLEAN := TRUE;
BEGIN
  IF num <= 1 THEN
    is_prime := FALSE;
  ELSE
    FOR i IN 2..TRUNC(SQRT(num)) LOOP
      IF MOD(num, i) = 0 THEN
        is_prime := FALSE;
        EXIT;
      END IF;
    END LOOP;
  END IF;
  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;
/
```