Assignment 1

```
/*
Check greatest oh=f three numbers

DECLARE
   a NUMBER := &a ;
   b NUMBER := &b ;
    c NUMBER := &c;
BEGIN
      IF a > b
      AND a > c THEN
           dbms_output.Put_line('Greatest number is ' || a);
   ELSIF b > a AND b > c THEN
          dbms_output.Put_line('Greatest number is '|| b);
   ELSE
          dbms_output.Put_line('Greatest number is '|| c);
   END IF;
END;

*/

/*
Number is odd or even

DECLARE
   n NUMBER := 4;
   r NUMBER;
BEGIN
   r := MOD(n, 2);

   IF r = 0 THEN
     dbms_output.Put_line('Even');
   ELSEIF r = 1 THEN
     dbms_output.Put_line('Odd');
   ELSE
   dbms_output.put_line('not a integer');
   END IF;
END;

*/
```

```
/*
Check Grade for a specific score


DECLARE
  score NUMBER := 75;      -- Change this value to the actual score
  grade CHAR(1);           -- Grade variable to store the result
BEGIN
  -- Use a CASE statement to determine the grade based on the score
  CASE
    WHEN score >= 90 THEN grade := 'A';
    WHEN score >= 80 THEN grade := 'B';
    WHEN score >= 70 THEN grade := 'C';
    WHEN score >= 60 THEN grade := 'D';
    ELSE grade := 'F';
  END CASE;

  DBMS_OUTPUT.PUT_LINE('Score: ' || score || ', Grade: ' || grade);
END;
/

*/

/*
write a table for the given number

DECLARE
n number;
i number;

BEGIN
n:=5;
for i in 1..10
loop
dbms_output.put_line(n||' x '||i||' = '||n*i);
end loop;
end;

*/
```

```
/*
factorial of a number

DECLARE
fac number :=1;
 n number :=7;

BEGIN
while n > 0 loop
fac:=n*fac;
n:=n-1;
end loop;
dbms_output.put_line(fac);
end;

*/

/*
Fibonacci Series

DECLARE
  n NUMBER := 8;  -- Input limit of Fibonacci series
  a NUMBER := 0;   -- First no. in series
  b NUMBER := 1;   -- Second no. in series
  c NUMBER;        -- Next no. in series
BEGIN
  DBMS_OUTPUT.PUT_LINE('Fibonacci series up to ' || n || ':');
  DBMS_OUTPUT.PUT_LINE(a);  -- Display first no.

  WHILE b <= n LOOP
    DBMS_OUTPUT.PUT_LINE(b);  -- Display current no.
    c := a + b;               -- Calculate next no.
    a := b;                   -- Update first no.
    b := c;                   -- Update second no.
  END LOOP;
END;
/

*/
```

```
/*
Reverse Number

DECLARE
  num NUMBER := &num;      -- Input the number
  original_num NUMBER := num;  -- Store the original number for comparison
  reverse_num NUMBER := 0;      -- Variable to store the reversed number
  digit NUMBER;          -- Variable to store the current digit
BEGIN
  WHILE num > 0 LOOP
    digit := num mod 10;                -- Get the last digit of the number
    reverse_num := reverse_num * 10 + digit;  -- Add the digit to the reversed number
    num := num / 10;                   -- Remove the last digit from the number
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Original number: ' || original_num);
  DBMS_OUTPUT.PUT_LINE('Reverse number: ' || reverse_num);
END;
/
*/
```

Assignment 2

```
/*
Write a PL/SQL program to demonstrate following exceptions
too many rows
DECLARE
  v_value1 VARCHAR2(10);
BEGIN
  SELECT column_name INTO v_value1
  FROM your_table
  WHERE condition; -- Replace 'your_table' and 'condition' with the appropriate values

  DBMS_OUTPUT.PUT_LINE('Value 1: ' || v_value1);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No data found.');
  WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE('Too many rows.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
Write a PL/SQL program to demonstrate following exceptions
no data found
DECLARE
  v_value1 VARCHAR2(10);
BEGIN
  SELECT column_name INTO v_value1
  FROM your_table
  WHERE condition; -- Replace 'your_table' and 'condition' with the appropriate values

  DBMS_OUTPUT.PUT_LINE('Value 1: ' || v_value1);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No data found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

Write a PL/SQL code to display a message to check whether the record is deleted or not.

```
-- Declare variables
DECLARE
  record_exists BOOLEAN := FALSE;  -- Flag to check if record exists
BEGIN
  -- Check if record exists in the table
  SELECT COUNT(*) INTO record_exists FROM your_table_name WHERE your_condition;

  -- Display message based on record existence
  IF record_exists THEN
    DBMS_OUTPUT.PUT_LINE('Record exists and has not been deleted.');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Record does not exist or has been deleted.');
  END IF;
END;
/
```

Write a PL/SQL code to display a message to provide the information about the number of records deleted by the delete statement issued in a PL/SQL block.

```
-- Declare variables
DECLARE
  num_deleted NUMBER := 0;  -- Number of records deleted
BEGIN
  -- Delete records using DELETE statement
  DELETE FROM your_table_name WHERE your_condition RETURNING COUNT(*) INTO
num_deleted;

  -- Display message with number of records deleted
  IF num_deleted > 0 THEN
    DBMS_OUTPUT.PUT_LINE('Number of records deleted: ' || num_deleted);
  ELSE
    DBMS_OUTPUT.PUT_LINE('No records deleted.');
  END IF;
END;
/
```

Write a PL/SQL code to demonstrate %TYPE and %ROWTYPE to display details of employees in EMP table.

```
DECLARE
  -- Declare variables using %TYPE for data type inference
  v_empno emp.empno%TYPE;
  v_ename emp.ename%TYPE;
  v_job emp.job%TYPE;
  v_sal emp.sal%TYPE;

  -- Declare record variable using %ROWTYPE for data type inference
  v_emp_rec emp%ROWTYPE;
BEGIN
  -- Fetch employee details using %TYPE
  SELECT empno, ename, job, sal
  INTO v_empno, v_ename, v_job, v_sal
  FROM emp
  WHERE empno = 7369; -- Replace with the appropriate empno value

  -- Display employee details
  DBMS_OUTPUT.PUT_LINE('Employee Details using %TYPE:');
  DBMS_OUTPUT.PUT_LINE('Empno: ' || v_empno);
  DBMS_OUTPUT.PUT_LINE('Ename: ' || v_ename);
  DBMS_OUTPUT.PUT_LINE('Job: ' || v_job);
  DBMS_OUTPUT.PUT_LINE('Sal: ' || v_sal);

  -- Fetch employee details using %ROWTYPE
  SELECT *
  INTO v_emp_rec
  FROM emp
  WHERE empno = 7369; -- Replace with the appropriate empno value

  -- Display employee details
  DBMS_OUTPUT.PUT_LINE('Employee Details using %ROWTYPE:');
  DBMS_OUTPUT.PUT_LINE('Empno: ' || v_emp_rec.empno);
  DBMS_OUTPUT.PUT_LINE('Ename: ' || v_emp_rec.ename);
  DBMS_OUTPUT.PUT_LINE('Job: ' || v_emp_rec.job);
  DBMS_OUTPUT.PUT_LINE('Sal: ' || v_emp_rec.sal);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No data found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
```

/
Write a PL/SQL code to display the empno, ename, job of employees of department number 10 for EMP table (using Cursor).

```
DECLARE
  -- Declare cursor
  CURSOR emp_cursor IS
    SELECT empno, ename, job
    FROM emp
    WHERE deptno = 10;

  -- Declare variables to store column values
  v_empno emp.empno%TYPE;
  v_ename emp.ename%TYPE;
  v_job emp.job%TYPE;
BEGIN
  -- Open cursor
  OPEN emp_cursor;

  -- Fetch and display employee details
  DBMS_OUTPUT.PUT_LINE('Empno | Ename | Job');
  DBMS_OUTPUT.PUT_LINE('----------------------');

  LOOP
    FETCH emp_cursor INTO v_empno, v_ename, v_job;
    EXIT WHEN emp_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(v_empno || ' | ' || v_ename || ' | ' || v_job);
  END LOOP;

  -- Close cursor
  CLOSE emp_cursor;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

Write a PL/SQL code to display the employee number and name of top 5 highest paid Employees (using Cursor).

```
DECLARE
  -- Declare cursor
  CURSOR emp_cursor IS
    SELECT empno, ename
    FROM emp
    ORDER BY sal DESC
    FETCH FIRST 5 ROWS ONLY; -- Fetch only top 5 highest paid employees

  -- Declare variables to store column values
  v_empno emp.empno%TYPE;
  v_ename emp.ename%TYPE;
BEGIN
  -- Open cursor
  OPEN emp_cursor;

  -- Fetch and display employee details
  DBMS_OUTPUT.PUT_LINE('Employee Number | Employee Name');
  DBMS_OUTPUT.PUT_LINE('-------------------------------');

  LOOP
    FETCH emp_cursor INTO v_empno, v_ename;
    EXIT WHEN emp_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE(v_empno || ' | ' || v_ename);
  END LOOP;

  -- Close cursor
  CLOSE emp_cursor;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
```

Write a PL/SQL code to calculate the total salary of first n records of emp table. The value of n Is passed to cursor as parameter

```
DECLARE
  -- Declare cursor with parameter
  CURSOR emp_cursor(p_n IN NUMBER) IS
    SELECT sal
    FROM emp
    WHERE ROWNUM <= p_n; -- Limit the number of rows to fetch

  -- Declare variable to store total salary
  v_total_salary NUMBER := 0;
BEGIN
  -- Fetch and calculate total salary
  FOR emp_rec IN emp_cursor(5) -- Pass the value of n as 5
  LOOP
    v_total_salary := v_total_salary + emp_rec.sal;
  END LOOP;

  -- Display total salary
  DBMS_OUTPUT.PUT_LINE('Total Salary of First ' || 5 || ' Employees: ' || v_total_salary);
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END;
/
*/
```