OUTPUT:

CALCULATE SUM OF TWO NUMBER

```
SQL> DECLARE
 2          A NUMBER(2);
 3          B NUMBER(2);
 4          C NUMBER(2);
 5    BEGIN
 6       A:=17;
 7       B:=19;
 8       C:=A+B;
 9    DBMS_OUTPUT.PUT_LINE(C);
10    DBMS_OUTPUT.PUT_LINE('SUM OF TWO NUMBER  ' || C);
11  END;
12  /

36
SUM OF TWO NUMBER  36
PL/SQL procedure successfully completed.
```

FIBONACCI SERIES

```
SQL> DECLARE
 2          A NUMBER(3);
 3          B NUMBER(3);
 4          C NUMBER(3);
 5          N NUMBER(3) := &N;
 6          NEGATIVE EXCEPTION;
 7   BEGIN
 8      IF N<2 THEN
 9          RAISE NEGATIVE;
10     END IF;
11     A:=0;
12     B:=1;
13     DBMS_OUTPUT.PUT_LINE('Fibonacci Series is...');
14     DBMS_OUTPUT.PUT_LINE(A);
15     DBMS_OUTPUT.PUT_LINE(B);
16     FOR I IN 3 ..
17     N LOOP
18         C:=A+B;
19         DBMS_OUTPUT.PUT_LINE(C);
20         A:=B;
21         B:=C;
22    END LOOP;
23    EXCEPTION
24    WHEN NEGATIVE THEN
25       DBMS_OUTPUT.PUT_LINE('N Should be greater than 1');
```

```
 26  END;
 27  /

Enter value for n: 5
old   5: N NUMBER(3) := &N;
new   5: N NUMBER(3) := 5;
Fibonacci Series is...
0
1
1
2
3

PL/SQL procedure successfully completed.
```

CHECK NUMBER IS ARMSTRONG OR NOT

```
SQL> DECLARE
  2        N NUMBER;
  3        S NUMBER;
  4        D NUMBER;
  5        DUP NUMBER;
  6  BEGIN
  7      N := &N;
  8      DUP := N;
  9      S := 0;
 10     WHILE (N!=0)
 11     LOOP
 12         D := MOD(N,10);
 13         S := S+(D*D*D);
 14         N := FLOOR(N/10);
 15     END LOOP;
 16     IF (DUP=S) THEN
 17         DBMS_OUTPUT.PUT_LINE('NUMBER IS ARMSTRONG');
 18    ELSE
 19         DBMS_OUTPUT.PUT_LINE('NUMBER IS NOT ARMSTRONG');
 20    END IF;
 21  END;
 22  /

Enter value for n: 17
old   7: N := &N;
new   7: N := 17;
NUMBER IS NOT ARMSTRONG

PL/SQL procedure successfully completed.
```

GENERATE ALL PRIME NUMBER BELOW 100

```
SQL> DECLARE
  2    I NUMBER;
  3    J NUMBER;
  4    C NUMBER;
  5  BEGIN
  6    DBMS_OUTPUT.PUT_LINE('The Prime Numbers are...');
  7    I := 2;
  8    WHILE (I<=100)
  9    LOOP
 10      C := 0;
 11      J := 1;
 12      WHILE (J<=I)
 13      LOOP
 14        IF (MOD(I,J)=0) THEN
 15          C := C+1;
 16        END IF;
 17        J := J+1;
 18      END LOOP;
 19      IF (C=2) THEN
 20        DBMS_OUTPUT.PUT_LINE(I);
 21      END IF;
 22      I := I+1;
 23    END LOOP;
 24  END;
 25  /

The Prime Numbers are...
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
```

```
71
73
79
83
89
97

PL/SQL procedure successfully completed.
```

INCREMENT SALARY BY 1000 FOR SPECIFIED EMP-CODE

```
SQL> CREATE TABLE EMP(EMPNO NUMBER(3),NAME VARCHAR(15),SAL NUMBER(5));

Table created.

SQL> INSERT INTO EMP VALUES(17,'KARTHI',9117);

1 row created.

SQL> DECLARE
  2    A NUMBER(4);
  3   BEGIN
  4      A := & EMPNO;
  5      UPDATE EMP SET SAL = SAL + 1000 WHERE EMPNO = A;
  6  END;
  7  /
Enter value for empno: 17
old   4:      A := & EMPNO;
new 4:      A := 17;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP;

EMPNO     NAME       SAL
----------- --------------- ----------
     17       KARTHI    10117
```

DELETE ROW FROM TABLE FOR SPECIFIED EMP-CODE

```
SQL> CREATE TABLE EMP(EMPNO NUMBER(3),NAME VARCHAR(15),DEPT VARCHAR(3));

Table created.

SQL> INSERT INTO EMP VALUES(21,'BOLT',’ECE’);

1 row created.
```

```
SQL> DECLARE
  2     A NUMBER(4);
  3    BEGIN
  4       A := & EMPNO;
  5       DELETE FROM EMP WHERE EMPNO = A;
  6  END;
  7  /
Enter value for empno: 21
old   4:      A := & EMPNO;
new 4:      A := 21;

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMP;

No rows selected
```