Practical no. 2

Aim: Writing a PL/SQL block with basic programming constructs by including the following.

1) Sequential statements
2) Unconstrained loop

1. Sequential Statement

a. Write a pl/sql block to perform arithmetic operation entered by the user.

Program:

```
set serveroutput on

accept operation char prompt "Enter the operation(+, -, *, /): "
accept n1 number prompt "Enter first number: "
accept n2 number prompt "Enter second number: "

DECLARE
    N1 NUMBER;
    N2 NUMBER;
    OPERATION VARCHAR(1);
BEGIN
    N1 := &N1;
    N2 := &N2;
    OPERATION := '&OPERATION';

    IF OPERATION = '+' THEN
        DBMS_OUTPUT.PUT_LINE(chr(10)||'The addition of '||N1||' and
'||N2||' is '||(N1+N2));
    ELSIF OPERATION = '-' THEN
        DBMS_OUTPUT.PUT_LINE('The addition of '||N1||' and '||N2||' is
'||(N1-N2));
    ELSIF OPERATION = '*' THEN
        DBMS_OUTPUT.PUT_LINE('The addition of '||N1||' and '||N2||' is
'||(N1*N2));
    elsif OPERATION = '/' THEN
        DBMS_OUTPUT.PUT_LINE('The addition of '||N1||' and '||N2||' is
'||(N1/N2));
    END if;
END;
/
```

Output:

```
Enter the operation(+, -, *, /): -
Enter first number: 10
Enter second number: 8
old   6:     N1 := &N1;
new   6:     N1 := 10;
old   7:     N2 := &N2;
new   7:     N2 := 8;
```

```
old   8:     OPERATION := '&OPERATION';
new   8:     OPERATION := '-';
The addition of 10 and 8 is 2

Enter the operation(+, -, *, /): /
Enter first number: 100
Enter second number: 2
old   6:     N1 := &N1;
new   6:     N1 := 100;
old   7:     N2 := &N2;
new   7:     N2 := 2;
old   8:     OPERATION := '&OPERATION';
new   8:     OPERATION := '/';
The addition of 100 and 2 is 50
```

2. Unconstrained loop

a. Write a pl/sql block to generate table of 20

Program:

```
set serveroutput on;
accept num number prompt "Enter the number: ";

DECLARE
    NUM NUMBER;
    I NUMBER;
BEGIN
    NUM := &NUM;
    I := 1;
    DBMS_OUTPUT.PUT_LINE(CHR(10));

    LOOP
        DBMS_OUTPUT.PUT_LINE(NUM||' * '||I||' = '||NUM*I);
        I:=I+1;
        EXIT WHEN I > 10;
    END LOOP;
END;
/
```

Output:

```
Enter the number: 20
old   5:     NUM := &NUM;
new   5:     NUM :=  20;

20 * 1 = 20
20 * 2 = 40
20 * 3 = 60
20 * 4 = 80
20 * 5 = 100
20 * 6 = 120
20 * 7 = 140
20 * 8 = 160
20 * 9 = 180
20 * 10 = 200
```

b. **To show the number between 1000-1010**

Program:

```
set serveroutput on

accept num1 number prompt "Enter the first number: "
accept num2 number prompt "Enter the second number: "

DECLARE
    NUM1 NUMBER;
    NUM2 NUMBER;

BEGIN
    num1 := &num1;
    num2 := &num2;
    DBMS_OUTPUT.PUT_LINE(CHR(10));

    LOOP
        DBMS_OUTPUT.PUT_LINE(NUM1);
        NUM1 := NUM1+1;
        EXIT WHEN NUM1>NUM2;
    END LOOP;
END;
/
```

Output:

```
Enter the first number: 1000
Enter the second number: 1010
old   6:     num1 := &num1;
new   6:     num1 := 1000;
old   7:     num2 := &num2;
new   7:     num2 := 1010;

1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
```

Practical no. 4

Aim: Writing a PL/SQL block with basic programming constructs by including the following.

1) IF .. THEN .. ELSE
2) IF .. ELSIF .. ELSE .. END IF
3) CASE

1. IF .. THEN .. ELSE

a. Write a pl/sql block to check whether number is less than 50

Program:

```
set serveroutput on

DECLARE
    NUM NUMBER := 9;
BEGIN
    IF (NUM < 50) THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10)||NUM ||' is less than 50.');
    END IF;
    DBMS_OUTPUT.PUT_LINE(NUM ||' is entered.');
END;
/
```

Output:

```
9 is less than 50.
9 is entered.
```

b. Write a pl/sql block to check number entered by user is less than 50

Program:

```
set serveroutput on

accept num number prompt "Enter the number: "

DECLARE
    NUM NUMBER := &NUM;
BEGIN
    IF (NUM < 50) THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10) ||NUM ||' is less than 50.');
    END IF;
    DBMS_OUTPUT.PUT_LINE(NUM ||' is entered.');
END;
/
```

Output:

```
Enter the number: 49
old   2:     NUM NUMBER := &NUM;
new   2:     NUM NUMBER :=         49;
```

```
49 is less than 50.
49 is entered.
```

2. IF .. ELSIF .. ELSE .. END IF

a. Write a pl/sql program to update salary of employee by 2000 for eid = 1 if salary is less than or equal to 20000

Queries:

```
SQL> CREATE TABLE EMP(eid number, ename varchar2(20), salary number);

Table created.

SQL> INSERT INTO EMP VALUES(&eid,'&ename',&salary);
Enter value for eid: 1
Enter value for ename: Jayesh
Enter value for salary: 19900
old   1: insert into emp values(&eid,'&ename',&salary)
new   1: insert into emp values(1,'Jayesh',19900)

1 row created.

SQL> /
Enter value for eid: 2
Enter value for ename: Jay
Enter value for salary: 22099
old   1: insert into emp values(&eid,'&ename',&salary)
new   1: insert into emp values(2,'Jay',22099)

1 row created.

SQL> /
Enter value for eid: 3
Enter value for ename: Yash
Enter value for salary: 15000
old   1: insert into emp values(&eid,'&ename',&salary)
new   1: insert into emp values(3,'Yash',15000)

1 row created.

SQL> /
Enter value for eid: 4
Enter value for ename: Om
Enter value for salary: 9000
old   1: insert into emp values(&eid,'&ename',&salary)
new   1: insert into emp values(4,'Om',9000)

1 row created.

SQL> /
Enter value for eid: 5
Enter value for ename: Nilesh
Enter value for salary: 35000
old   1: insert into emp values(&eid,'&ename',&salary)
```

```
new   1: insert into emp values(5,'Nilesh',35000)

1 row created.

SQL> SELECT * FROM EMP;

       EID ENAME                    SALARY
---------- -------------------- ----------
         1 Jayesh                    19900
         2 Jay                       22099
         3 Yash                      15000
         4 Om                         9000
         5 Nilesh                    35000
```

Program:

```
SET SERVEROUTPUT ON

DECLARE
    ID  EMP.EID%TYPE:=1;
    SAL EMP.SALARY%TYPE;

BEGIN
    SELECT SALARY INTO SAL FROM EMP WHERE EID = ID;
    IF (SAL <= 20000) THEN
        UPDATE EMP SET SALARY=SALARY+2000 WHERE EID = ID;
        DBMS_OUTPUT.PUT_LINE('Salary is updated');
    END IF;
END;
/
```

Output:

```
SQL> @emp_query
Salary is updated

PL/SQL procedure successfully completed.

SQL> select * from emp;

       EID ENAME                    SALARY
---------- -------------------- ----------
         1 Jayesh                    21900
         2 Jay                       22099
         3 Yash                      15000
         4 Om                         9000
         5 Nilesh                    35000
```

b. **Write a pl/sql program to update salary of employee by 2000 for user entered eid if salary is less than or equal to 20000**

Program:

```
set serveroutput on

DECLARE
```

```
    ID  EMP.EID%TYPE;
    SAL EMP.SALARY%TYPE;

BEGIN
    ID:=&ID;
    SELECT SALARY INTO SAL FROM EMP WHERE EID=ID;

    IF (SAL <= 20000) THEN
        UPDATE EMP SET SALARY=SALARY+2000 WHERE EID=ID;
        DBMS_OUTPUT.PUT_LINE('Salary is updated!!');
    END IF;
END;
/
```

Output:

```
SQL> @emp_query2
Enter value for id: 3
old   6:     id:=&id;
new   6:     id:=3;
Salary is updated!!

PL/SQL procedure successfully completed.

SQL> select * from emp;

       EID ENAME                    SALARY
---------- -------------------- ----------
         1 Jayesh                    21900
         2 Jay                       22099
         3 Yash                      17000
         4 Om                         9000
         5 Nilesh                    35000
```

c. **Write a pl/sql program to update salary of employee by 2000 for all eid if salary is less than or equal to 20000**

Program:

```
set serveroutput on

DECLARE
    ID  EMP.EID%TYPE;
    SAL EMP.SALARY%TYPE;
BEGIN
    FOR I IN (SELECT EID, SALARY FROM EMP) LOOP
        ID := I.EID;
        SAL := I.SALARY;
        IF SAL <= 20000 THEN
            UPDATE EMP SET SALARY=SALARY+2000 WHERE EID=ID;
            DBMS_OUTPUT.PUT_LINE('Salary updated for employee '||ID);
        END IF;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Salaries are updated');
END;
```

```
/
```

Output:

```
SQL> @c_table_query
Salary updated for employee 3
Salary updated for employee 4
Salaries are updated

PL/SQL procedure successfully completed.

SQL> select * from emp;

       EID ENAME                    SALARY
---------- -------------------- ----------
         1 Jayesh                    21900
         2 Jay                       22099
         3 Yash                      19000
         4 Om                        11000
         5 Nilesh                    35000
```

3. CASE STATEMENT

a. Write a pl/sql program to display which remark got

Program:

```
set serveroutput on

accept grade char prompt "Enter your grades: "

DECLARE
    GRADE CHAR;
BEGIN
    GRADE := '&grade';
    CASE GRADE
        WHEN 'A' THEN
            DBMS_OUTPUT.PUT_LINE('Excellent');
        WHEN 'B' THEN
            DBMS_OUTPUT.PUT_LINE('Very good');
        WHEN 'C' THEN
            DBMS_OUTPUT.PUT_LINE('Well done');
        WHEN 'D' THEN
            DBMS_OUTPUT.PUT_LINE('You passed');
        WHEN 'F' THEN
            DBMS_OUTPUT.PUT_LINE('Better try again');
        ELSE
            DBMS_OUTPUT.PUT_LINE('No such grade');
    END CASE;
END;
/
```

Output:

```
SQL> @case
```

```
Enter your grades: A
old   4:     grade := '&grade';
new   4:     grade := 'A';
Excellent

PL/SQL procedure successfully completed.

SQL> @case
Enter your grades: G
old   4:     grade := '&grade';
new   4:     grade := 'G';
No such grade

PL/SQL procedure successfully completed.

SQL> @case
Enter your grades: F
old   4:     grade := '&grade';
new   4:     grade := 'F';
Better try again

PL/SQL procedure successfully completed.
```

4. IF .. ELSIF .. ELSE .. END IF

Program:

```
set serveroutput on

accept A number prompt "Enter the number: "

DECLARE
    A NUMBER;
BEGIN
    A := &A;
    IF (A=10) THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10)||'Value of a is 10');
    ELSIF (A=20) THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10)||'Value of a is 20');
    ELSIF (A=30) THEN
        DBMS_OUTPUT.PUT_LINE(CHR(10)||'Value of a is 30');
    ELSE
        DBMS_OUTPUT.PUT_LINE(CHR(10)||'None of the values is
        matching');
    END IF;
    DBMS_OUTPUT.PUT_LINE('Exact value of a is '||A);
END;
/
```

Output:

```
SQL> @match
Enter the number: 10
old   5:     a := &a;
new   5:     a :=         10;
```

```
Value of a is 10
Exact value of a is 10

PL/SQL procedure successfully completed.

SQL> @match
Enter the number: 100
old   5:     a := &a;
new   5:     a :=        100;

None of the values is matching
Exact value of a is 100

PL/SQL procedure successfully completed.
```