PRACTICAL – 06

NAME : Aryan Kashikar
Roll n0: D2 32

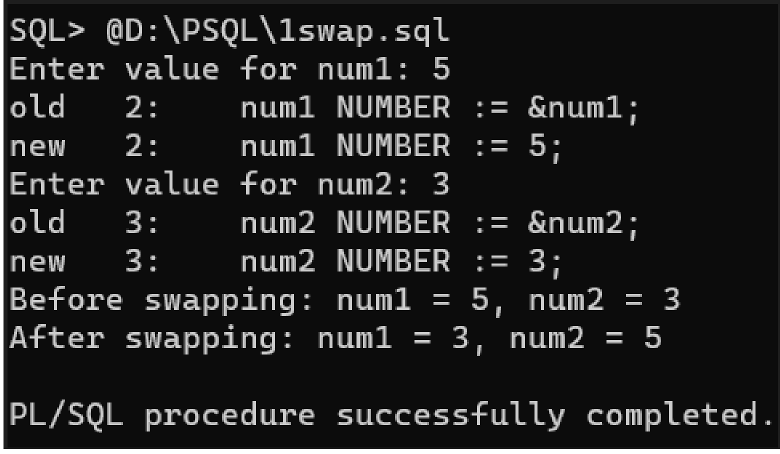
Introduction to PL/SQL and execution of PL/SQL blocks using oracle 11g

AIM: To write PL/SQL blocks using oracle 11g

Problem Statements:

1. Write a PL/SQL program for swapping 2 numbers.

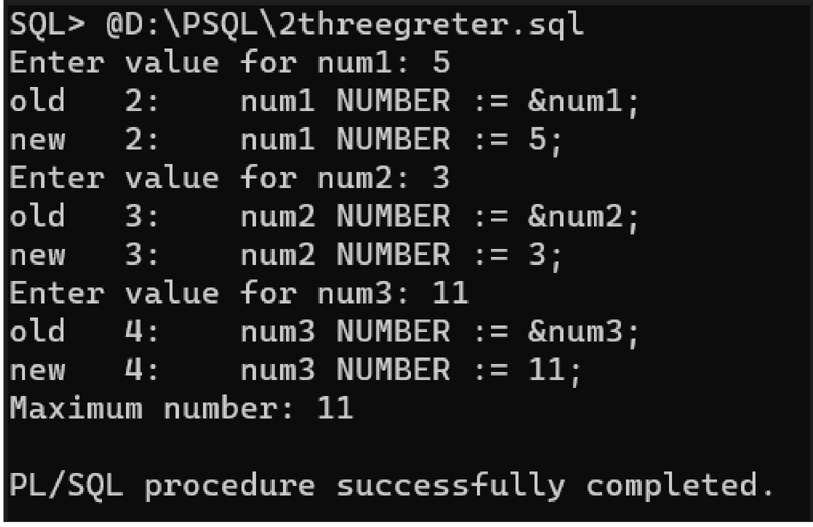
```
DECLARE
   num1 NUMBER := &num1;
   num2 NUMBER := &num2;
   temp NUMBER;
BEGIN
   DBMS_OUTPUT.PUT_LINE('Before swapping: num1 = ' || num1 || ', num2 = ' || num2);

   temp := num1;
   num1 := num2;
   num2 := temp;

   DBMS_OUTPUT.PUT_LINE('After swapping: num1 = ' || num1 || ', num2 = ' || num2);
END;
/
```



```
SQL> @D:\PSQL\1swap.sql
Enter value for num1: 5
old   2:     num1 NUMBER := &num1;
new   2:     num1 NUMBER := 5;
Enter value for num2: 3
old   3:     num2 NUMBER := &num2;
new   3:     num2 NUMBER := 3;
Before swapping: num1 = 5, num2 = 3
After swapping: num1 = 3, num2 = 5

PL/SQL procedure successfully completed.
```

2. Write a PL/SQL block to find the maximum number from given three numbers.

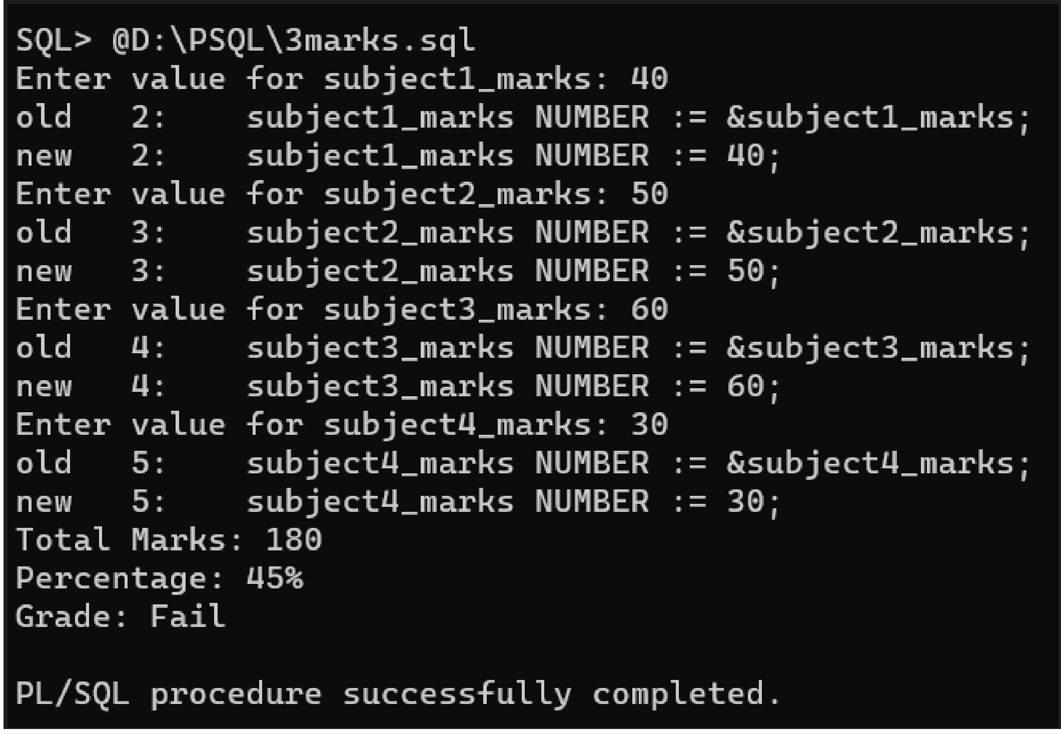
```
DECLARE
   num1 NUMBER := &num1;
   num2 NUMBER := &num2;
   num3 NUMBER := &num3;
   max_num NUMBER;
BEGIN
   IF num1 >= num2 AND num1 >= num3 THEN
      max_num := num1;
   ELSIF num2 >= num1 AND num2 >= num3 THEN
      max_num := num2;
   ELSE
      max_num := num3;
   END IF;

   DBMS_OUTPUT.PUT_LINE('Maximum number: ' || max_num);
END;
/
```



```
SQL> @D:\PSQL\2threegreter.sql
Enter value for num1: 5
old   2:     num1 NUMBER := &num1;
new   2:     num1 NUMBER := 5;
Enter value for num2: 3
old   3:     num2 NUMBER := &num2;
new   3:     num2 NUMBER := 3;
Enter value for num3: 11
old   4:     num3 NUMBER := &num3;
new   4:     num3 NUMBER := 11;
Maximum number: 11

PL/SQL procedure successfully completed.
```

3. Write a PL/SQL program to input marks of 4 subjects. find the total and Percentage of 4 subjects and display the grade. [Grade “fail “ if any of subject got percentage less than 40,
Otherwise grade A (>85),B(less than 85 and >=75),C(less than 75 and >=65), D(less than 65)

```
DECLARE
   subject1_marks NUMBER := &subject1_marks;
   subject2_marks NUMBER := &subject2_marks;
   subject3_marks NUMBER := &subject3_marks;
   subject4_marks NUMBER := &subject4_marks;
   total_marks NUMBER;
   percentage NUMBER;
   grade VARCHAR2(10);
BEGIN
   total_marks := subject1_marks + subject2_marks + subject3_marks + subject4_marks;
   percentage := total_marks / 4;

   IF subject1_marks < 40 OR subject2_marks < 40 OR subject3_marks < 40 OR
subject4_marks < 40 THEN
      grade := 'Fail';
   ELSIF percentage > 85 THEN
      grade := 'A';
   ELSIF percentage >= 75 THEN
      grade := 'B';
   ELSIF percentage >= 65 THEN
      grade := 'C';
   ELSE
      grade := 'D';
   END IF;

   DBMS_OUTPUT.PUT_LINE('Total Marks: ' || total_marks);
   DBMS_OUTPUT.PUT_LINE('Percentage: ' || percentage || '%');
   DBMS_OUTPUT.PUT_LINE('Grade: ' || grade);
END;
/
```

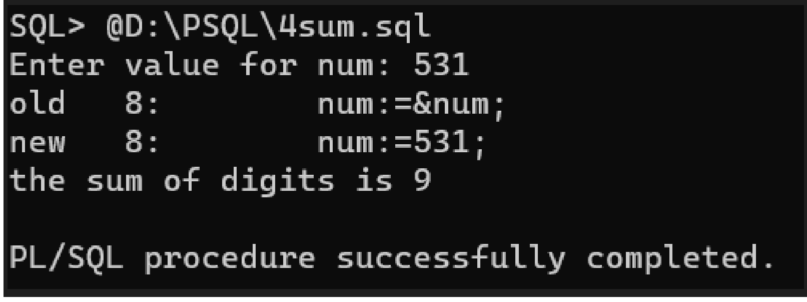


```
SQL> @D:\PSQL\3marks.sql
Enter value for subject1_marks: 40
old   2:     subject1_marks NUMBER := &subject1_marks;
new   2:     subject1_marks NUMBER := 40;
Enter value for subject2_marks: 50
old   3:     subject2_marks NUMBER := &subject2_marks;
new   3:     subject2_marks NUMBER := 50;
Enter value for subject3_marks: 60
old   4:     subject3_marks NUMBER := &subject3_marks;
new   4:     subject3_marks NUMBER := 60;
Enter value for subject4_marks: 30
old   5:     subject4_marks NUMBER := &subject4_marks;
new   5:     subject4_marks NUMBER := 30;
Total Marks: 180
Percentage: 45%
Grade: Fail

PL/SQL procedure successfully completed.
```

4. Write a program to accept a number and find the sum of its digits.

```
declare
    num int :=0;
    i int;
    s int :=0;
    r int;

begin
```

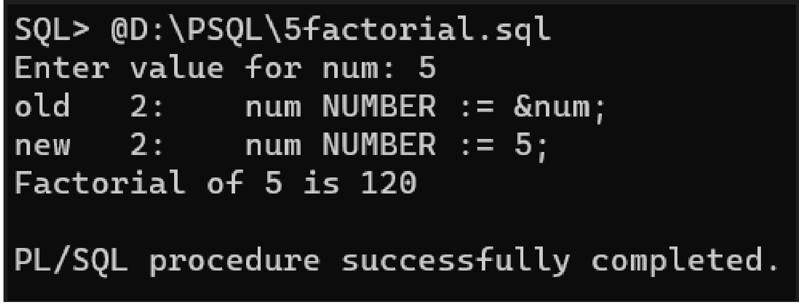
```
    num:=&num;
    while num > 0 loop
        r:= MOD(num, 10);
        s := s + r;
        num:=floor(num/10);
    end loop;

    dbms_output.put_line('  the sum of digits is '||s );

end;
/
```



```
SQL> @D:\PSQL\4sum.sql
Enter value for num: 531
old   8:         num:=&num;
new   8:         num:=531;
the sum of digits is 9

PL/SQL procedure successfully completed.
```

5. Write a PL/SQL program to find the factorial of a given number.

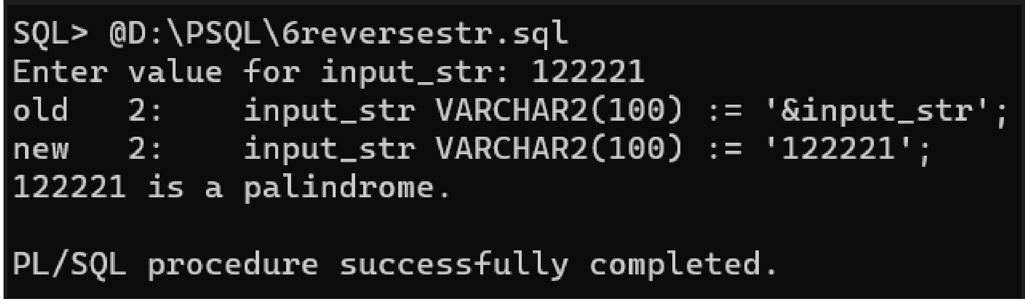
```
DECLARE
   num NUMBER := &num;
   factorial NUMBER := 1;
BEGIN
   IF num < 0 THEN
      DBMS_OUTPUT.PUT_LINE('Factorial is not defined for negative numbers.');
   ELSE
      FOR i IN 1..num LOOP
         factorial := factorial * i;
      END LOOP;

      DBMS_OUTPUT.PUT_LINE('Factorial of ' || num || ' is ' || factorial);
   END IF;
END;
/
```



```
SQL> @D:\PSQL\5factorial.sql
Enter value for num: 5
old   2:     num NUMBER := &num;
new   2:     num NUMBER := 5;
Factorial of 5 is 120

PL/SQL procedure successfully completed.
```

6. Write a program to accept a number/string and check it is palindrome or not.

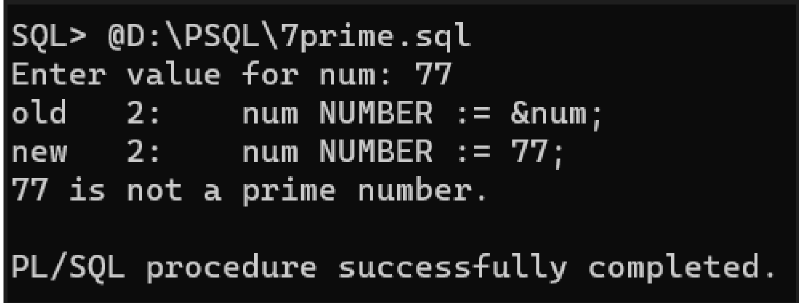
```
DECLARE
   input_str VARCHAR2(100) := '&input_str';
   reversed_str VARCHAR2(100) := '';
BEGIN
   FOR i IN REVERSE 1..LENGTH(input_str) LOOP
      reversed_str := reversed_str || SUBSTR(input_str, i, 1);
   END LOOP;

   IF input_str = reversed_str THEN
      DBMS_OUTPUT.PUT_LINE(input_str || ' is a palindrome.');
   ELSE
      DBMS_OUTPUT.PUT_LINE(input_str || ' is not a palindrome.');
   END IF;
END;
/
```



```
SQL> @D:\PSQL\6reversestr.sql
Enter value for input_str: 122221
old   2:    input_str VARCHAR2(100) := '&input_str';
new   2:    input_str VARCHAR2(100) := '122221';
122221 is a palindrome.

PL/SQL procedure successfully completed.
```

7. Write a PL / SQL program to check whether the given number is prime or not.

```
DECLARE
   num NUMBER := &num;
   is_prime BOOLEAN := TRUE;
BEGIN
   IF num <= 1 THEN
      is_prime := FALSE;
   ELSE
      FOR i IN 2..(num-1) 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;
/
```

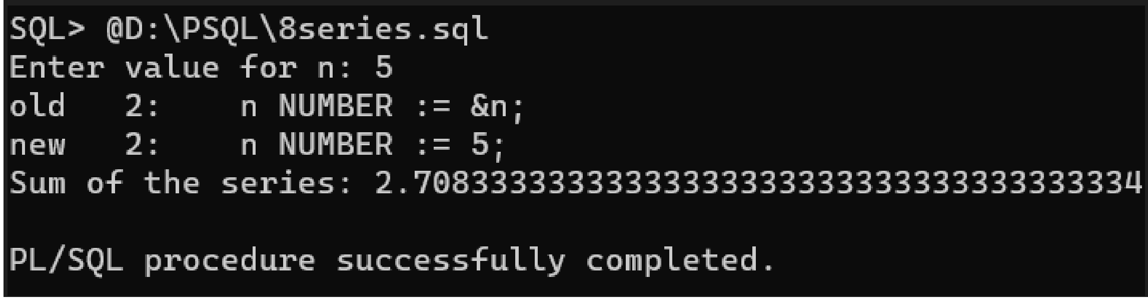


```
SQL> @D:\PSQL\7prime.sql
Enter value for num: 77
old   2:     num NUMBER := &num;
new   2:     num NUMBER := 77;
77 is not a prime number.

PL/SQL procedure successfully completed.
```

8. Write a PL / SQL code to find sum of below series.

S=1/1!+2/2!+3/3!........n/n!

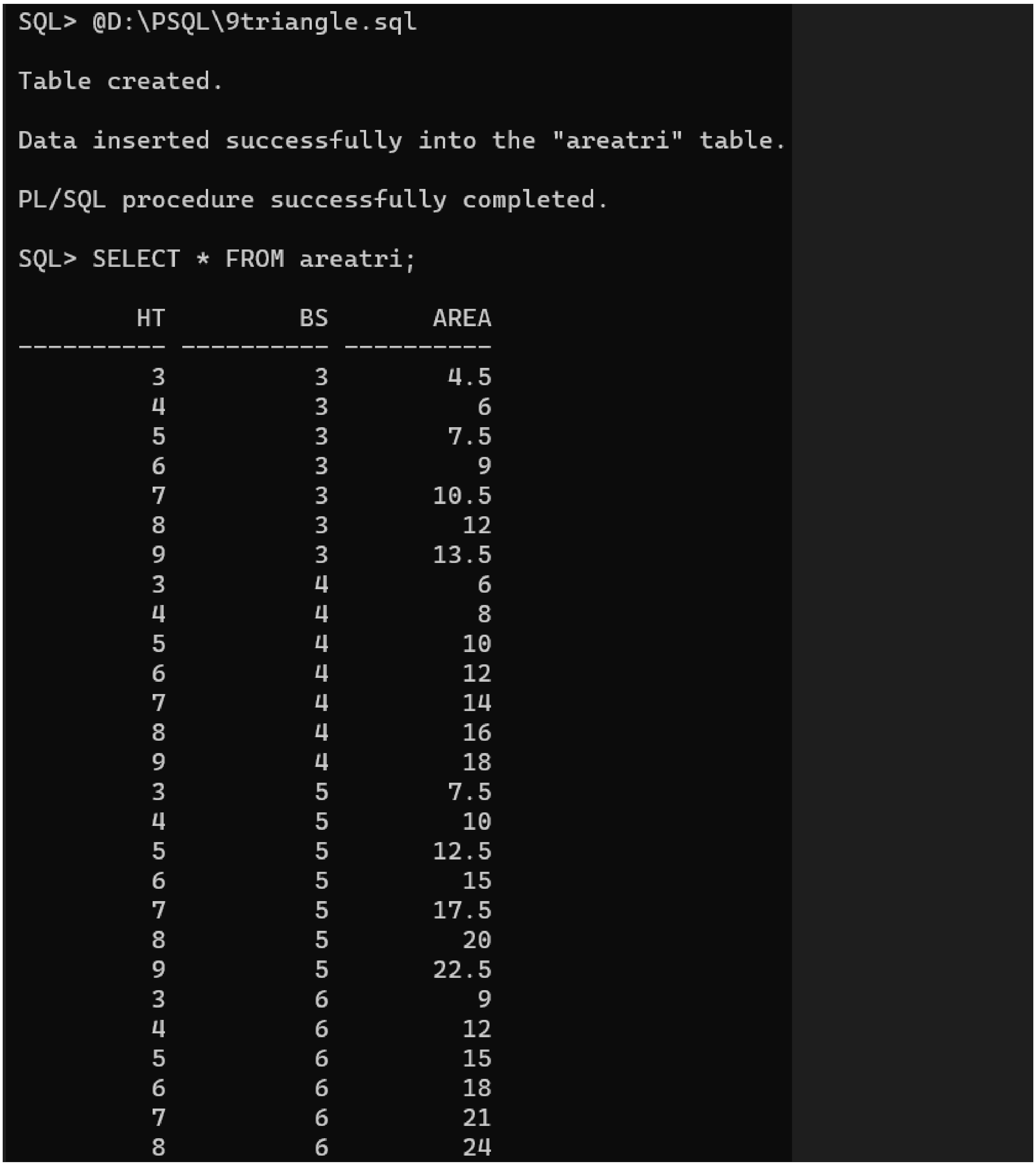
```
DECLARE
   n NUMBER := &n;
   s NUMBER := 0;
   factorial NUMBER := 1;
BEGIN
   FOR i IN 1..n LOOP
      factorial := factorial * i;
      s := s + (i / factorial);
   END LOOP;

   DBMS_OUTPUT.PUT_LINE('Sum of the series: ' || s);
END;
/
```



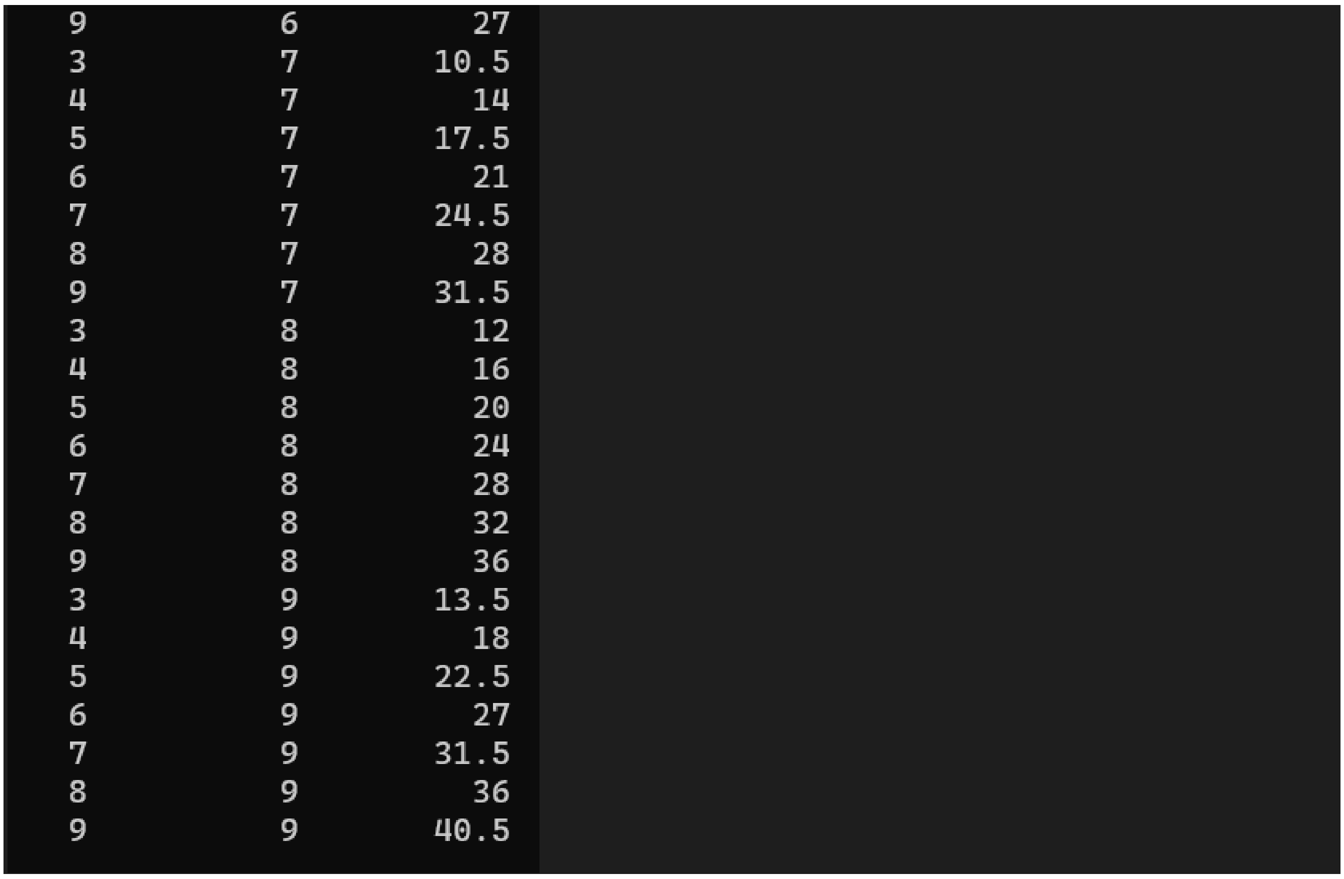
```
SQL> @D:\PSQL\8series.sql
Enter value for n: 5
old   2:     n NUMBER := &n;
new   2:     n NUMBER := 5;
Sum of the series: 2.70833333333333333333333333333333333334

PL/SQL procedure successfully completed.
```

9. Calculate the area of a triangle for a value of base varying from 3 to 9 and for each base value height varying from 3 to 9. Store the base, height and the corresponding values of calculated area in table areatri. Consisting of three columns ht, bs and area.

```
CREATE TABLE areatri (
   ht NUMBER,
   bs NUMBER,
   area NUMBER
);

DECLARE
   base_start NUMBER := 3;
   base_end NUMBER := 9;
   height_start NUMBER := 3;
   height_end NUMBER := 9;
   area NUMBER;
BEGIN
   FOR base IN base_start..base_end LOOP
      FOR height IN height_start..height_end LOOP
         area := (base * height) / 2;
         INSERT INTO areatri (ht, bs, area) VALUES (height, base, area);
      END LOOP;
   END LOOP;

   COMMIT;
   DBMS_OUTPUT.PUT_LINE('Data inserted successfully into the "areatri" table.');
END;
/
```



```
SQL> @D:\PSQL\9triangle.sql

Table created.

Data inserted successfully into the "areatri" table.

PL/SQL procedure successfully completed.

SQL> SELECT * FROM areatri;

        HT         BS       AREA
---------- ---------- ----------
         3          3        4.5
         4          3          6
         5          3        7.5
         6          3          9
         7          3       10.5
         8          3         12
         9          3       13.5
         3          4          6
         4          4          8
         5          4         10
         6          4         12
         7          4         14
         8          4         16
         9          4         18
         3          5        7.5
         4          5         10
         5          5       12.5
         6          5         15
         7          5       17.5
         8          5         20
         9          5       22.5
         3          6          9
         4          6         12
         5          6         15
         6          6         18
         7          6         21
         8          6         24
```



```
         9              6          27
         3              7        10.5
         4              7          14
         5              7        17.5
         6              7          21
         7              7        24.5
         8              7          28
         9              7        31.5
         3              8          12
         4              8          16
         5              8          20
         6              8          24
         7              8          28
         8              8          32
         9              8          36
         3              9        13.5
         4              9          18
         5              9        22.5
         6              9          27
         7              9        31.5
         8              9          36
         9              9        40.5
```

10. Print below pattern

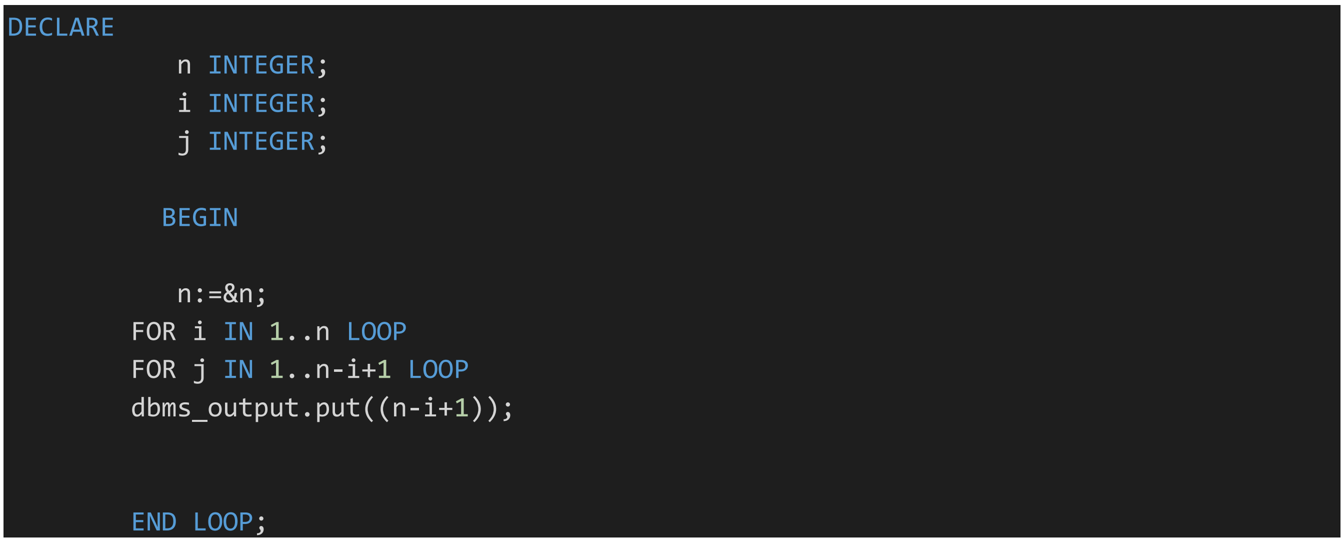
55555

4444

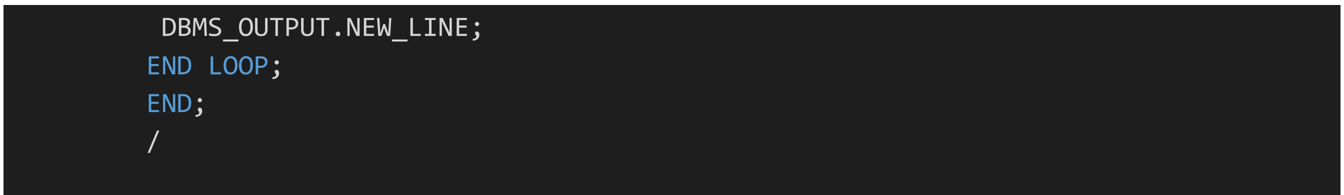
333

22

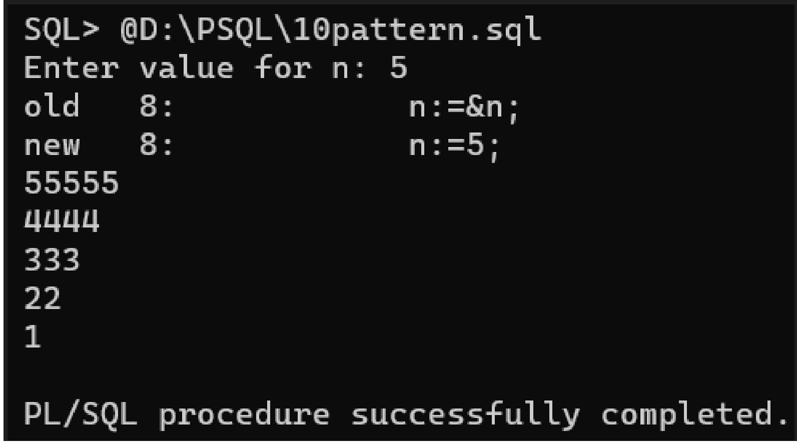
1



```
DECLARE
           n INTEGER;
           i INTEGER;
           j INTEGER;

          BEGIN

           n:=&n;
        FOR i IN 1..n LOOP
        FOR j IN 1..n-i+1 LOOP
        dbms_output.put((n-i+1));


        END LOOP;
```



```
         DBMS_OUTPUT.NEW_LINE;
        END LOOP;
        END;
        /
```



```
SQL> @D:\PSQL\10pattern.sql
Enter value for n: 5
old   8:            n:=&n;
new   8:            n:=5;
55555
4444
333
22
1

PL/SQL procedure successfully completed.
```