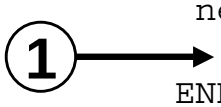
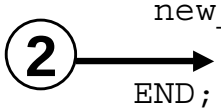
Practice 3

PL/SQL Block

```
DECLARE
 weight    NUMBER(3) := 600;
 message   VARCHAR2(255) := 'Product 10012';
BEGIN
  DECLARE
   weight   NUMBER(3) := 1;
   message   VARCHAR2(255) := 'Product 11001';
   new_locn  VARCHAR2(50) := 'Europe';
  BEGIN
   weight := weight + 1;
   new_locn := 'Western ' || new_locn;
 END;
 weight := weight + 1;
 message := message || ' is in stock';
 new_locn := 'Western ' || new_locn;
END;
/
```





1. Evaluate the PL/SQL block given above and determine the data type and value of each of the following variables according to the rules of scoping.
   a. The value of `weight` at position 1 is:

   b. The value of `new_locn` at position 1 is:

   c. The value of `weight` at position 2 is:

   d. The value of `message` at position 2 is:

   e. The value of `new_locn` at position 2 is:

Practice 3 (continued)

Scope Example

```
DECLARE
   customer        VARCHAR2(50) := 'Womansport';
   credit_rating   VARCHAR2(50) := 'EXCELLENT';
BEGIN
    DECLARE
       customer     NUMBER(7) := 201;
       name    VARCHAR2(25) := 'Unisports';
    BEGIN
       credit_rating :='GOOD';
       …
    END;
  …
END;
/
```

2. In the PL/SQL block shown above, determine the values and data types for each of the following cases.
   a. The value of `customer` in the nested block is:

   b. The value of `name` in the nested block is:

   c. The value of `credit_rating` in the nested block is:

   d. The value of `customer` in the main block is:

   e. The value of `name` in the main block is:

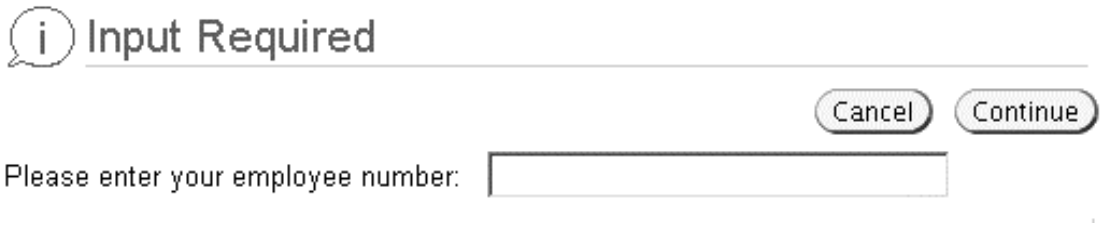
   f. The value of `credit_rating` in the main block is:

Practice 3 (continued)

3. Use the same session that you used to execute the practices in Lesson 2. If you have opened a new session, then execute `lab_02_05_soln.sql`. Edit `lab_02_05_soln.sql`.
   a. Use single line comment syntax to comment the lines that create the bind variables.
   b. Use multiple line comments in the executable section to comment the lines that assign values to the bind variables.
   c. Declare two variables: `fname` of type `VARCHAR2` and size 15, and `emp_sal` of type `NUMBER` and size 10.
   d. Include the following SQL statement in the executable section:
      ```
      SELECT first_name, salary
         INTO fname, emp_sal FROM employees
         WHERE employee_id=110;
      ```
   e. Change the line that prints 'Hello World' to print 'Hello' and the first name. You can comment the lines that display the dates and print the bind variables, if you want to.
   f. Calculate the contribution of the employee towards provident fund (PF). PF is 12% of the basic salary and basic salary is 45% of the salary. Use the bind variables for the calculation. Try and use only one expression to calculate the PF. Print the employee's salary and his contribution towards PF.
   g. Execute and save your script as `lab_03_03_soln.sql`. Sample output is shown below.

      ```
      Hello John
      YOUR SALARY IS : 8200
      YOUR CONTRIBUTION TOWARDS PF: 442.8
      PL/SQL procedure successfully completed.
      ```

4. Accept a value at run time using the substitution variable. In this practice, you will modify the script that you created in exercise 3 to accept user input.
   a. Load the script `lab_03_04.sql` file.
   b. Include the `PROMPT` command to prompt the user with the following message: 'Please enter your employee number.'
   c. Modify the declaration of the `empno` variable to accept the user input.
   d. Modify the select statement to include the variable `empno`.
   e. Execute and save your script as `lab_03_04_soln.sql`. Sample output is shown below.



Enter 100 and click the Continue button.

```
Hello Steven
YOUR SALARY IS : 24000
YOUR CONTRIBUTION TOWARDS PF: 1296
PL/SQL procedure successfully completed.
```

5. Execute the script `lab_03_05.sql`. This script creates a table called `employee_details`.
    a. The `employee` and `employee_details` tables have the same data. You will update the data in the `employee_details` table. Do not update or change the data in the `employees` table.
    b. Open the script `lab_03_05b.sql` and observe the code in the file. Note that the code accepts the employee number and the department number from the user.
    c. You will use this as the skeleton script to develop the application, which was discussed in the lesson titled “Introduction.”