PL/SQL Definitions	
1	

	--pro to fetch per and update result.
	
	set serveroutput on
	declare
	xrlno number:=&xrlno;
	p number;
	begin
	select per INTO p from result where rlno=xrlno;
	if p >= 40 then
	update result set result='PASS' where rlno=xrlno;
	end if;
	end;
	/

2	WAP that update commission of 10% in sales if total sales > 50000. Table sales: (id, sales1, sales2, commission)

	set serveroutput on
	declare
	xid number:=&xid;
	s1 number(5);
	s2 number(5);
	total number(15); --total sales
	c number; --for commission
	begin
	select sales1,sales2 INTO s1,s2 from sales where id=xid;
	total:= s1 + s2;
	if total > 50000 then
	c:=total * 0.10;
	update sales set comm=c where id=xid;
	else
	update sales set comm=0 where id=xid;
	end if;
	end;
	/

3	WAP that update commission as per given slab: • Total <= 10000 , 0 • Total >10000 and <20000 , 10% • Total >=20000 and <50000 , 15% • Total >=50000 and <100000 , 20% • Total >=100000 , 25%

	set serveroutput on
	declare
	xid number:=&xid;
	s1 number(5);

	s2 number(5);
	total number(15); --total sales
	c number; --for commission
	begin
	select sales1,sales2 INTO s1,s2 from sales where id=xid;
	total:= s1 + s2;
	if total < 10000 then
	update sales set comm=0 where id=xid;
	elsif total >=10000 and total < 20000 then
	c:=total * 0.10;
	update sales set comm=c where id=xid;
	elsif total >=20000 and total < 50000 then
	c:=total * 0.15;
	update sales set comm=c where id=xid;
	elsif total >=50000 and total < 100000 then
	c:=total * 0.20;
	update sales set comm=c where id=xid;
	else
	c:=total * 0.25;
	update sales set comm=c where id=xid;
	end if;
	end;
	/

4	Program to demonstrate the use of looping statement . print 1 to n number.

	--loop demo
	set serveroutput on
	
	declare
	i number:=1;
	n number:=&n;
	j number;
	k number:=1;
	begin
	while i <= n
	loop
	dbms_output.put_line(i);
	i:= i + 1;
	end loop;
	
	dbms_output.put_line('using for loop');
	dbms_output.put_line('----------');
	
	for j in 1..n
	loop
	dbms_output.put_line(j);
	end loop;

	dbms_output.put_line('using loop iterative control');
	dbms_output.put_line('----------');
	
	loop
	dbms_output.put_line(k);
	k:=k + 1;
	exit when (k > n);
	end loop;
	end;
	/

5	Write a program to insert square and cube into table. Temp table: Id, square, cube

	--find square and cube
	set serveroutput on
	declare
	i number:=1;
	n number:=&n;
	begin
	while i <= n
	loop
	insert into temp values(i,i*i,i*i*i);
	i:=i + 1;
	
	end loop;
	end;
	/

Procedure	
1	Create a procedure to display message on screen without any parameters.

	set serveroutput on
	
	create or replace procedure show_msg
	is
	begin
	dbms_output.put_line('procedure without paramters');
	end show_msg;
	/

2	Create a procedure to insert record in course table. Create a calling program to execute procedure.
	--procedure set serveroutput on

	create or replace procedure course_ins(xsrno IN number,xcourse IN char, xcode IN varchar2) is begin insert into course values(xsrno,xcourse,xcode); commit; dbms_output.put_line('one record stored in course table'); end course_ins; /

	--calling program
	
	--procedure call
	
	declare
	sno number(2):=&sno;
	cnm char(15):='&cnm';
	ccode varchar2(15):='&ccode';
	begin
	course_ins(sno,cnm,ccode);
	dbms_output.put_line('calling program successfully executed');
	end;
	/

3	Definition 6
	--Write a simple procedure without any parameter that updates the values in the EMP table. set serveroutput on create or replace procedure comm_update is begin update emp set comm=500; end comm_update; /
4	Definition 7
	--Write a simple procedure that increases by the salary of employees for the given department no by percentage inputted by the user using IN parameter set serverouptput on create or replace procedure inc_sal_emp(xdeptno IN number,per IN number) is begin

	update emp set salary= salary + (salary * (per/100)) where deptno=xdeptno; dbms_output.put_line('salary incremented by given per'); end inc_sal_emp; /
	--calling program

	--definition 7 calling
	
	declare
	d number(2):=&d;
	p number(2):=&p;
	begin
	inc_sal_emp(d,p);
	end;
	/

5	Definition 8

	--definition 8
	
	create or replace procedure search_Emp(id IN number, xename OUT char,xdeptno OUT number, xsalary OUT number)
	is
	begin
	select ename,deptno,salary INTO xename,xdeptno,xsalary from emp where empid=id;
	
	EXCEPTION
	WHEN NO_DATA_FOUND THEN
	dbms_output.put_line('invalid employee id');
	
	end search_Emp;
	/

	--calling program
	
	declare
	xid number(4):=&xid;
	enm char(15);
	d number(2);
	sal number(6);
	begin
	search_Emp(xid,enm,d,sal);
	dbms_output.put_line('Name of Employee:' ||enm);
	dbms_output.put_line('Deptno :' ||d);
	dbms_output.put_line('Salary:' ||sal);
	end;

	/
6	Create a function that find average of 3 inputted numbers.

	create or replace function find_Avg(x in number,y in number,z in number) RETURN number
	is
	ans number;
	begin
	ans:= (x + y + z)/3;
	return ans;
	end find_avg;
	/

	Calling function

	declare
	a number:=&a;
	b number:=&b;
	c number:=&c;
	f number;
	begin
	
	f:=find_avg(a,b,c);
	dbms_output.put_line('average='||f);
	end;
	/