```
/* Question 2.5 and Question 2.6 */

create or replace package course_registration as
    /* Triggers will be explicitly written */

    procedure showPreRequisitesforCourse(deptCode in Prerequisites.dept_code%type,
    courseNo in Prerequisites.course#%type, isFirst Boolean, result out varchar2);

    procedure enroll_student(param_B# in Enrollments.B#%type, param_classid in
    Enrollments.Classid%type, messages out varchar2);

    procedure drop_student(studentid_b# in students.b#%type,studentid_classid in
    classes.classid%type, message out varchar );

end;
/
show errors

/* Body for the package course_registration */
create or replace package body course_registration as

    procedure showPreRequisitesforCourse(deptCode in Prerequisites.dept_code%type,
    courseNo in Prerequisites.course#%type, isFirst Boolean, result out varchar2) is
       CURSOR courses_refcur is
          select pre_dept_code, pre_course# FROM Prerequisites WHERE dept_code =
          deptCode and course# = courseNo;
             course courses_refcur%rowtype;
             courseCount number;
             begin
          select count(*) into courseCount from Courses where dept_code = deptCode and
          course# = courseNo;
          open courses_refcur;
          fetch courses_refcur into course;
         if(courses_refcur%found) then
          while(courses_refcur%found) loop
           showPreRequisitesforCourse(course.pre_dept_code, course.pre_course#, FALSE,
           result);
           if(result is NULL) then
           result := course.pre_dept_code || course.pre_course#;
           else
           result := result || ',' || course.pre_dept_code || course.pre_course#;
           end if;
           fetch courses_refcur into course;
          end loop;
         elsif(courseCount < 1 and  isFirst) then
          result := deptCode || courseNo || ' does not exist';
         end if;
    end showPreRequisitesforCourse;

    procedure enroll_student(param_B# in Enrollments.B#%type, param_classid in
    Enrollments.Classid%type, messages out varchar2) is

     enrolled_prerequisites number;
     total_prerequisites number;
     not_a_student exception;
     not_a_class exception;
     not_current_semester exception;
     class_already_full exception;
     already_enrolled exception;
     max_classes exception;
     relation_prerequisites exception;
     relation_prerequisites_enroll exception;

     student_count number;
     class_count number;
     sem_class_count number;
     class_full_count number;
     student_in_class_count number;
     student_total_classes number;
     student_prerequisites_count number;
     prerequisiteClasses varchar2(200);

     cursor c1 is
     select dept_code, course# from classes where classid = param_classid;
    class c1%rowtype;

        begin
          select count(*) into student_count from Students where B# = param_B#;
          if(student_count < 1) then
           raise not_a_student;
          end if;

          select count(*) into class_count from Classes where classid = param_classid;
          if(class_count < 1) then
           raise not_a_class;
          end if;

          select count(*) into sem_class_count from Classes where classid =
          param_classid and year = 2018 and semester = 'Fall';
          if(sem_class_count < 1) then
           raise not_current_semester;
          end if;

          select count(*) into class_full_count from Classes where classid =
          param_classid and year = 2018 and semester = 'Fall' and limit <> class_size;
          if(class_full_count < 1) then
           raise class_already_full;
          end if;

          select count(*) into student_in_class_count from Enrollments where classid =
          param_classid and B# = param_B#;
          if(student_in_class_count > 0) then
           raise already_enrolled;
          end if;

          select count(*) into student_total_classes from Enrollments where B# =
          param_B#;
          if(student_total_classes >= 5) then
           raise max_classes;
          end if;

              if(student_total_classes = 4) then
               messages := 'The student will be overloaded with the new enrollment';
              end if;

              if(student_total_classes >= 5) then
               raise max_classes;
              end if;

              open c1;
              fetch c1 into class;

              showPreRequisitesforCourse(class.dept_code, class.course#, TRUE,
              prerequisiteClasses);

              select count(*) into total_prerequisites from Classes where dept_code ||
              course# in (
              select regexp_substr(prerequisiteClasses,'[^,]+', 1, level) from dual
              connect by regexp_substr(prerequisiteClasses, '[^,]+', 1, level) is not
              null );

              Select count(*) into enrolled_prerequisites from Enrollments e, Classes
              cl where e.classid = cl.classid and e.B# = param_B#  and cl.dept_code ||
              cl.course# in (
              select regexp_substr(prerequisiteClasses,'[^,]+', 1, level) from dual
              connect by regexp_substr(prerequisiteClasses, '[^,]+', 1, level) is not
              null );

```

```
             if(total_prerequisites != enrolled_prerequisites) then
             raise relation_prerequisites_enroll;
             end if;

             showPreRequisitesforCourse(class.dept_code, class.course#, TRUE,
             prerequisiteClasses);
             select count(*) into student_prerequisites_count from Enrollments e,
             Classes c where B# = param_B# and e.Classid = c.classid and
             INSTR(prerequisiteClasses, c.dept_code || c.course#) > 0 and e.lgrade not
             in ('A','A-','B+','B','B-','C+','C');
             close c1;
             if(student_prerequisites_count > 0) then
              raise relation_prerequisites;
             end if;

 insert into Enrollments values (param_B#, param_classid, NULL);
   messages := 'Student added successfully';

exception
 when not_a_student then
  messages := 'The B# is invalid';
 when not_a_class then
  messages := 'The Classid is invalid';
 when not_current_semester then
  messages := 'Cannot enroll into a class from a previous semester';
 when class_already_full then
  messages := 'The class is already full';
 when already_enrolled then
  messages := 'The student is already in the class';
 when max_classes then
  messages := 'Students cannot be enrolled in more than five classes in the same
  semester';
 when relation_prerequisites then
  messages := 'Prerequisite not satisfied';
 when relation_prerequisites_enroll then
  messages := 'Prerequisite Issue : Student has not completed the Prerequisites';
end enroll_student;


       procedure drop_student(studentid_b# in students.b#%type,studentid_classid in
       classes.classid%type, message out varchar )
   is
       message1 varchar(50);
       studentid_class_enroll_cnt int;
       studentid_dept_code classes.dept_code%type;
       message2 varchar(50);
       studentid_prereq_cnt int;
       studentid_class_size int;
       studentid_course# classes.course#%type;
       studentid_class_size_cnt int;
       studentid_count int;
       studentid_class_cnt int;

   begin
   select count(*) into studentid_count from students where b# = studentid_b#;
   if studentid_count=0
   then
       message:='The B# is invalid';
       else
       select count(*) into studentid_class_cnt from classes where
       classid=studentid_classid;
           if studentid_class_cnt=0
           then
               message:='The classid is invalid';
               else
                   select count(*) into studentid_class_enroll_cnt from enrollments
                   where classid=studentid_classid and b#=studentid_b#;
                   if studentid_class_enroll_cnt=0
                   then
```

```
                        message:= 'The student is not enrolled in the class';
                    else
                        select dept_code,course# into
                        studentid_dept_code,studentid_course# from classes where
                        classid=studentid_classid;

                        select count(*) into studentid_prereq_cnt from classes
                        cl,prerequisites p where classid in (select classid from
                        enrollments where classid<>studentid_classid and
                        b#=studentid_b#) and
                        cl.dept_code=p.dept_code and cl.course# = p.course# and
                        p.pre_dept_code=studentid_dept_code and
                        p.pre_course#=studentid_course#;

                    if studentid_prereq_cnt>0
                    then
                        message:='The drop is not permitted because another class uses
                        it as a prerequisite';

                    else
                        delete from enrollments where b#=studentid_b# and
                        classid=studentid_classid;
                        message:='Student dropped successfully';

                        select class_size into studentid_class_size from classes where
                        classid=studentid_classid;

                        select count(*) into studentid_class_size_cnt from enrollments
                        where b#=studentid_b#;

                        if studentid_class_size=0
                        then
                            message1:='The class now has no students.';
                        end if;

                        if studentid_class_size_cnt=0
                        then
                            message2:='This student is not enrolled in any classes.';
                        end if;
                        message:=message||' '||message1||' '||message2;



                    end if;
                end if;
            end if;
        end if;
     end drop_student ;

end course_registration;
/
show errors
```