**COMP9120 Relational Database Systems**

**Tutorial Week 7 Solution: DB Application Programming**

**Exercise 1. Establishing Database Connections**

Download the solution file from Canvas

**Exercise 2. Read-Only Database Access**

a) Extend the existing listUnits() function so that it also lists for each unit the name of the faculty member who was teaching it.

    See listUnitsWithFacultyMember method of Java solution file

b) Extend the existing listUnits() function so that it takes a parameter 'name' and only lists the courses taught by a given faculty member.

    See listUnitsOfGivenFacultyMember method of Java solution file

**Exercise 3. Adding a new query**

Now try adding a new method from scratch. We wish to be able to obtain a student's transcript in the Student Registration System.

a) In pgAdmin, write and check a query to get the details of all units completed by a student. This should include details for uosCode, uosName, credits, semester, year, grade.

    See answer to 3 (b)

b) Based upon the listUnits() method, write a method listTranscript(int studentID) that performs the same query as you just wrote and uses cursor (ResultSet) to print out a student's transcript.

    See listTranscript3b method of Java solution file.

c) Create the stored procedure by executing the script below in pgAdmin, (it includes the query required in 3 (b)):

```
CREATE OR REPLACE FUNCTION listTranscript (studentID INTEGER) RETURNS REFCURSOR AS $$
DECLARE
        vCursor REFCURSOR;
BEGIN
        OPEN vCursor FOR
        SELECT uosCode, uosName, credits, semester, year, grade
        FROM Transcript JOIN UnitOfStudy USING(uosCode)
        WHERE studId=studentID
```

       **ORDER BY** uosCode,year,semester;
       **RETURN** vCursor;
**END; $$ LANGUAGE plpgsql**;

d) (Trickier) Update your listTranscript method to execute your stored procedure instead of using the query directly. Note you will need to use a CallableStatement instead of a PreparedStatement.

    See listTranscript3d method of Java solution file