**Database Systems Project 2 – Using PL/SQL and JDBC**

**to Implement Student Registration System**

**Name:** Prathyusha Vudatha

**Email:** [svudath1@binghamton.edu](mailto:svudath1@binghamton.edu)

**B.Nos:** B00708850

**Functionality Implementation**

**PL/SQL**

**Procedures used:**

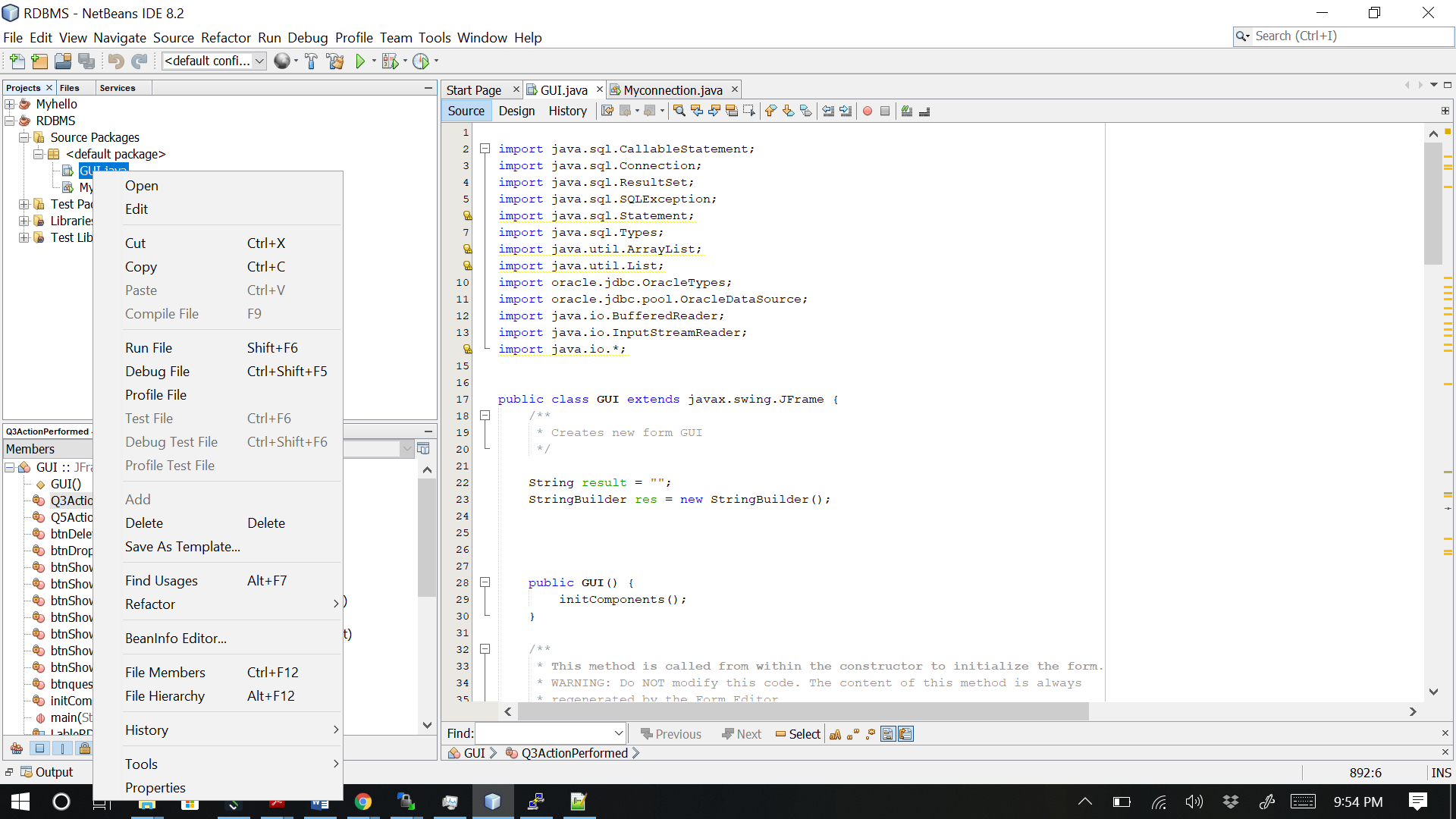
1. show\_students(): This procedure is used to show the details of the Students table.
2. show\_tas():This procedure is used to show the details of the Tas table.
3. show\_courses():This procedure is used to show the details of the Courses table.
4. show\_classes():This procedure is used to show the details of the Classes table.
5. show\_enrollments():This procedure is used to show the details of the Enrollments table.
6. show\_ta():This procedure is used to show the details of the TA of the class given the classid.
7. show\_prerequisites():This procedure is used to show the details of the Students table.
8. show\_logs():This procedure is used to show the details of the Logs table.
9. show\_prereq(): This procedure is used to return all prerequisite courses for the given course.
10. enroll\_student(): This procedure is used to enrol a student into a class.
11. drop\_class(): This procedure is used to drop a student from a class.
12. delete\_student(): This procedure is used to delete a student from a Students table.

**Triggers used:**

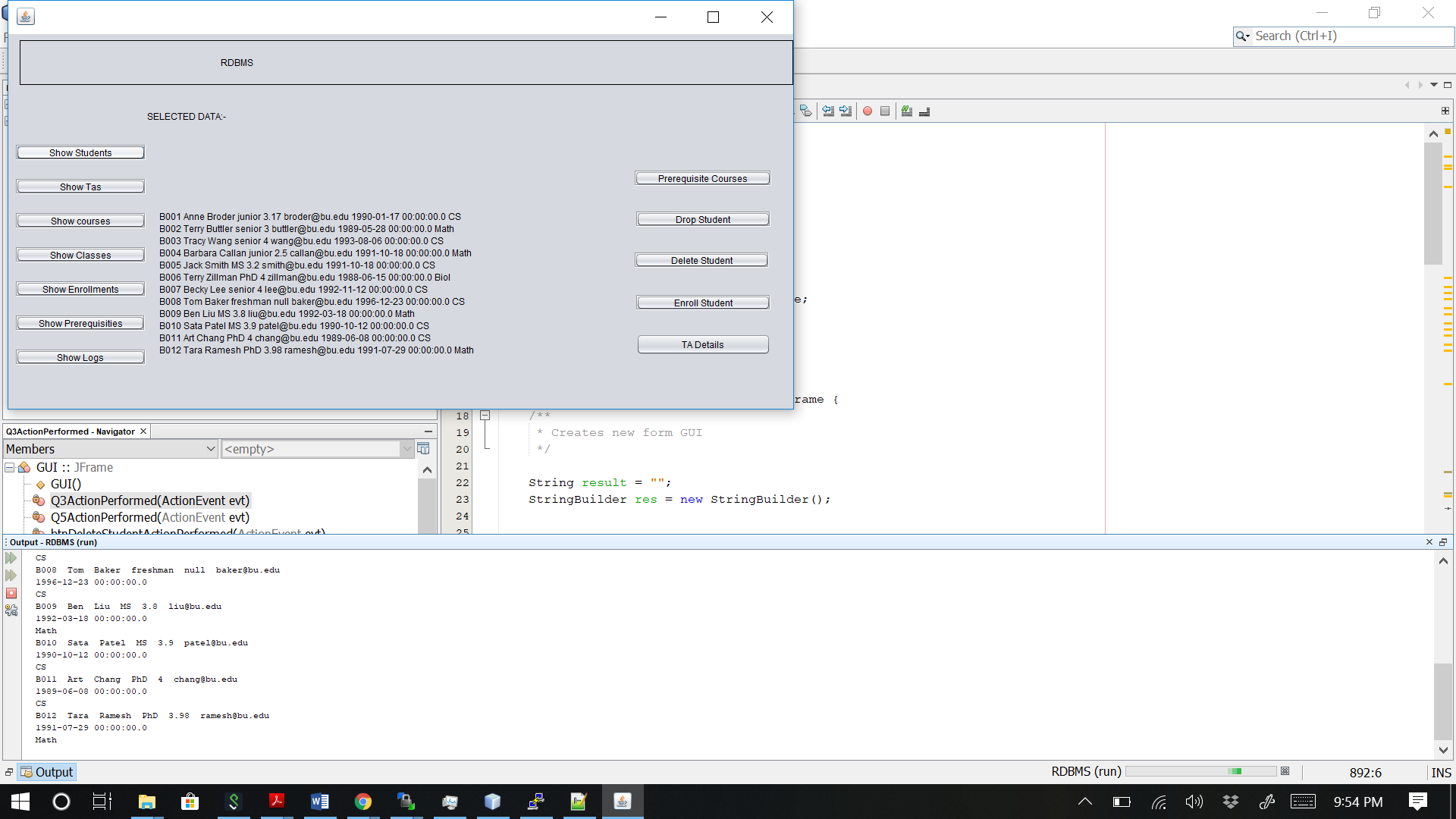
1. DELETE\_STUDENT\_ADD\_LOGS: This trigger is used to add a tuple to Logs table when a student is deleted from Students table.
2. INSERT\_ENROLLMENTS\_ADD\_LOGS: This trigger is used to add a tuple to Logs table when a tuple is inserted into the Enrollments table.
3. DELETE\_ENROLLMENTS\_ADD\_LOGS: This trigger is used to add a tuple to Logs table when a tuple is deleted from the Enrollments table.
4. DELETE\_ENROLLMENTS\_DELETE\_STUDENTS: This trigger is used to delete all tuples in the Enrollments table involving the student when a student is deleted.
5. UPDATE\_LOGS\_INSERT\_STUDENTS: This trigger is used to update the values caused by successfully enrolling a student into a class.

**SCREENSHOTS**

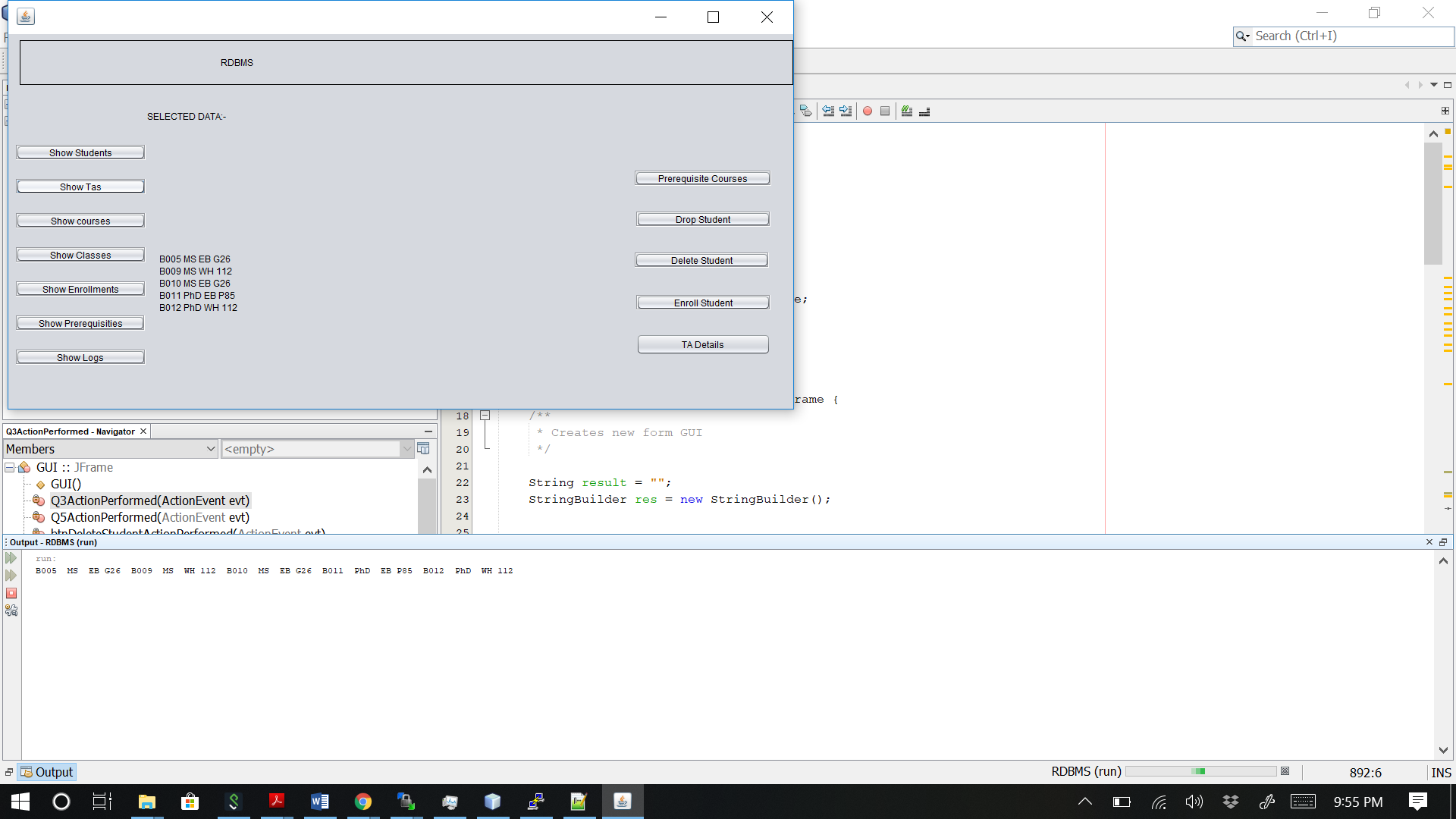
* **Run the file GUI.java**

****

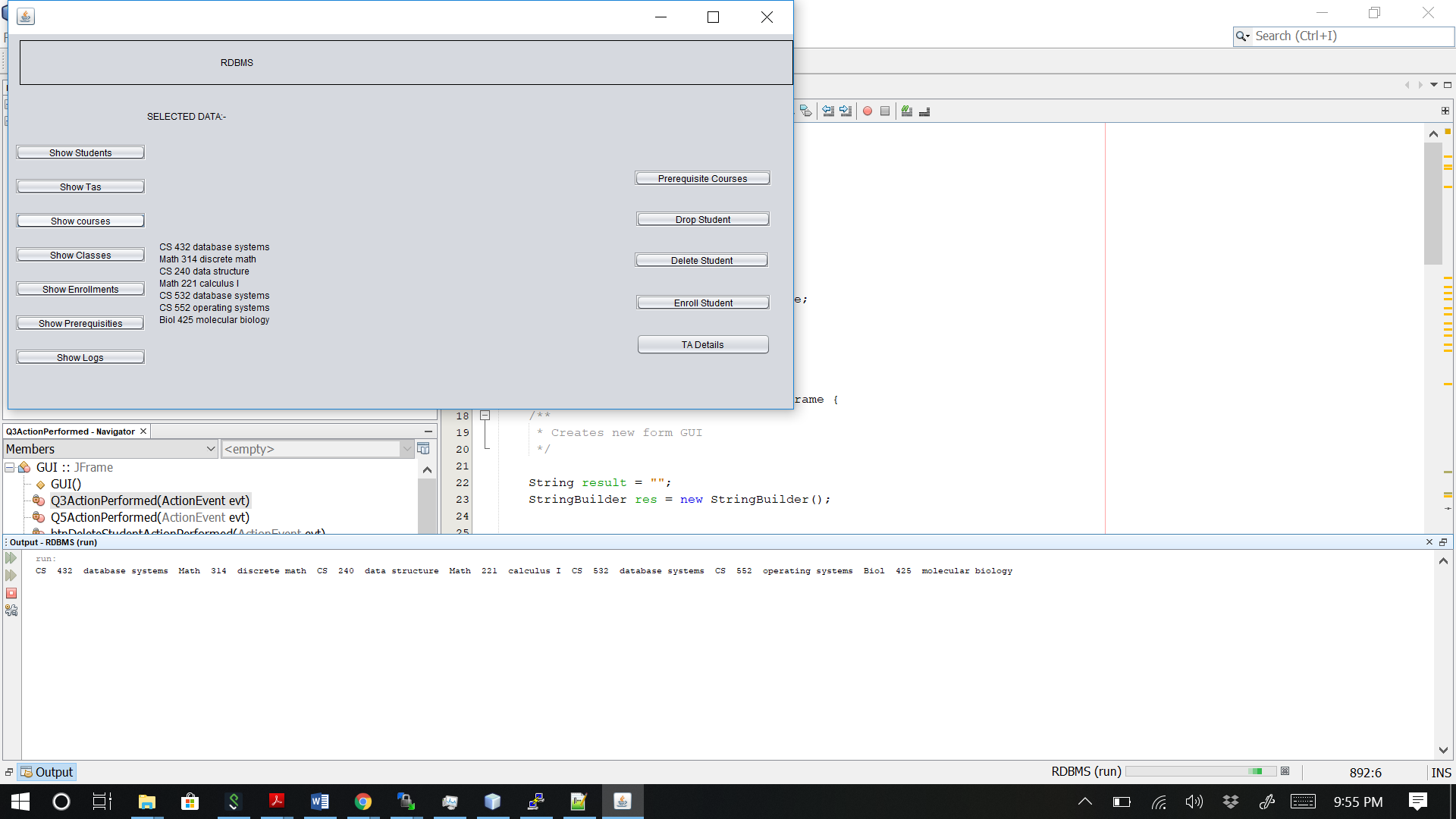
* **Click on Show students button**

****

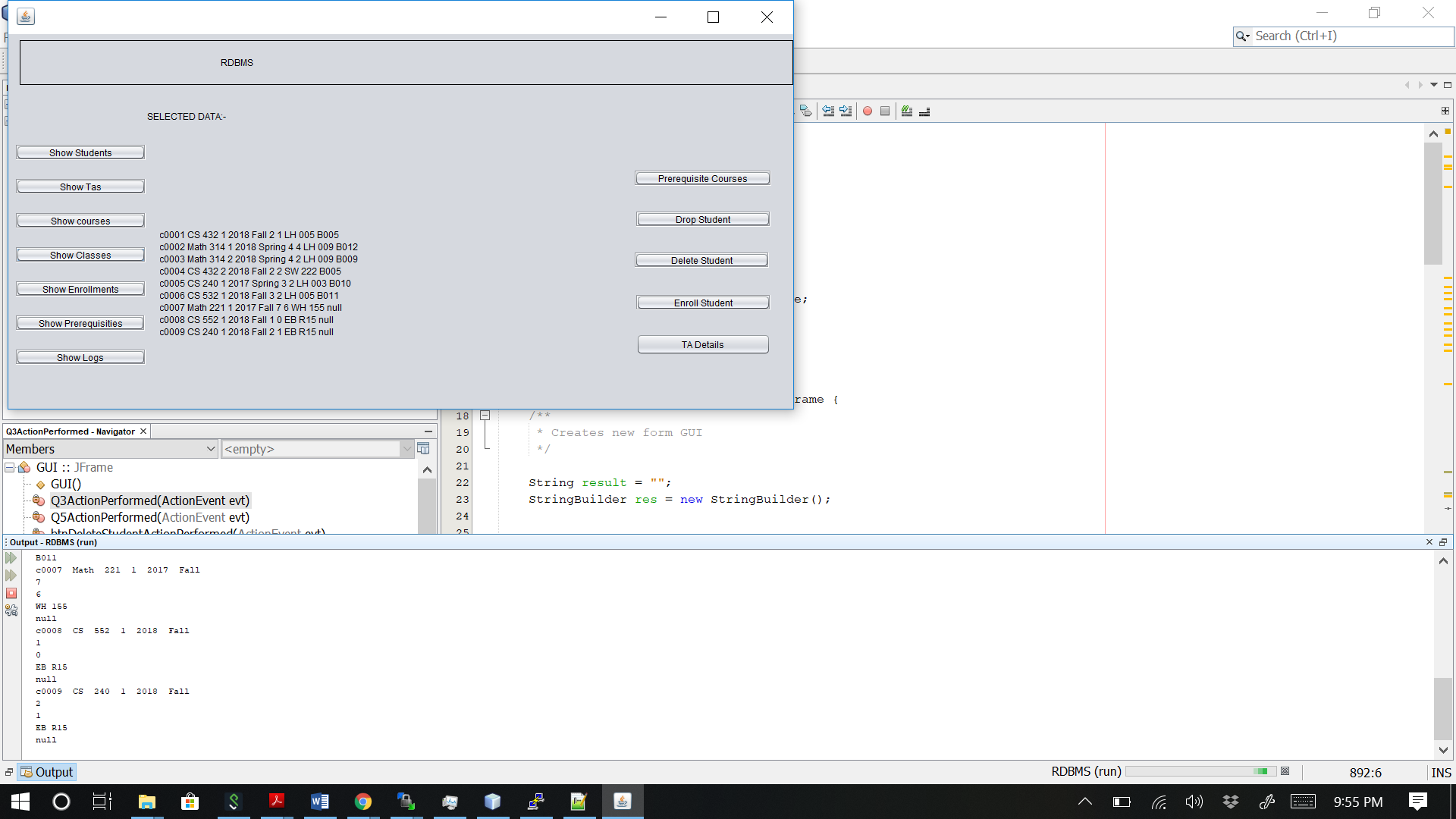
* **Click on Show Tas button**

****

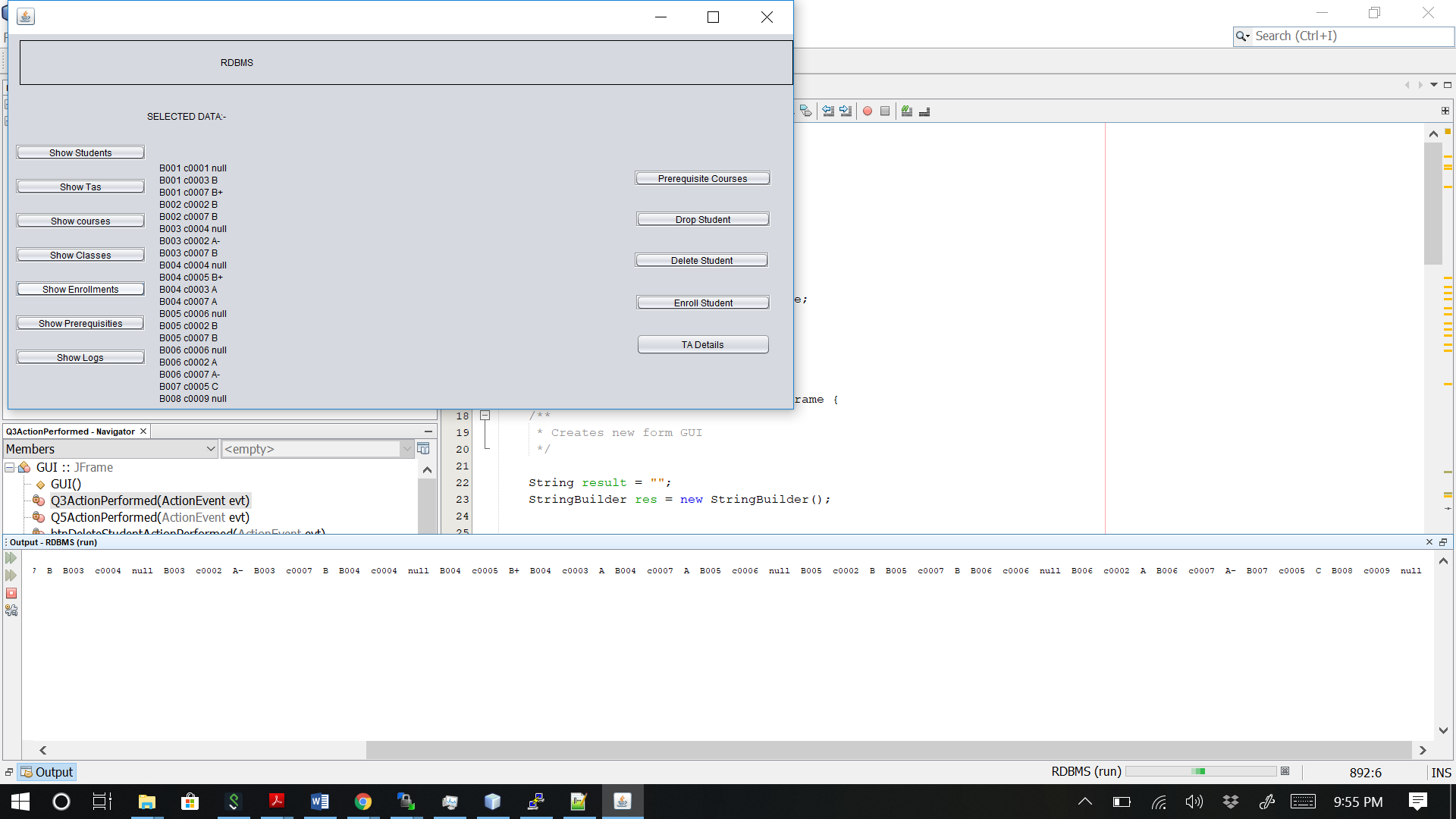
* **Click on Show Courses button**

****

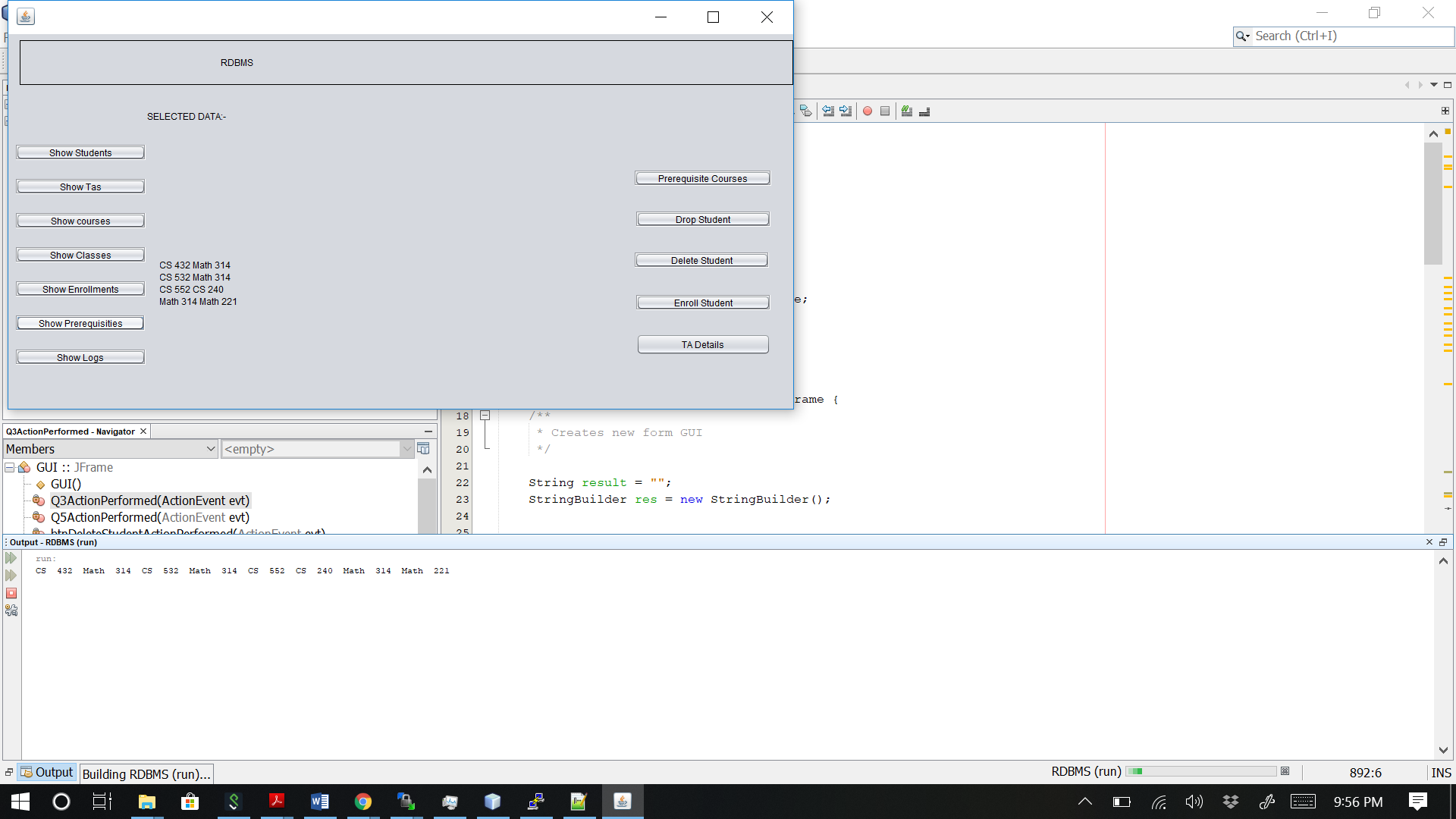
* **Click on Show Classes button**

****

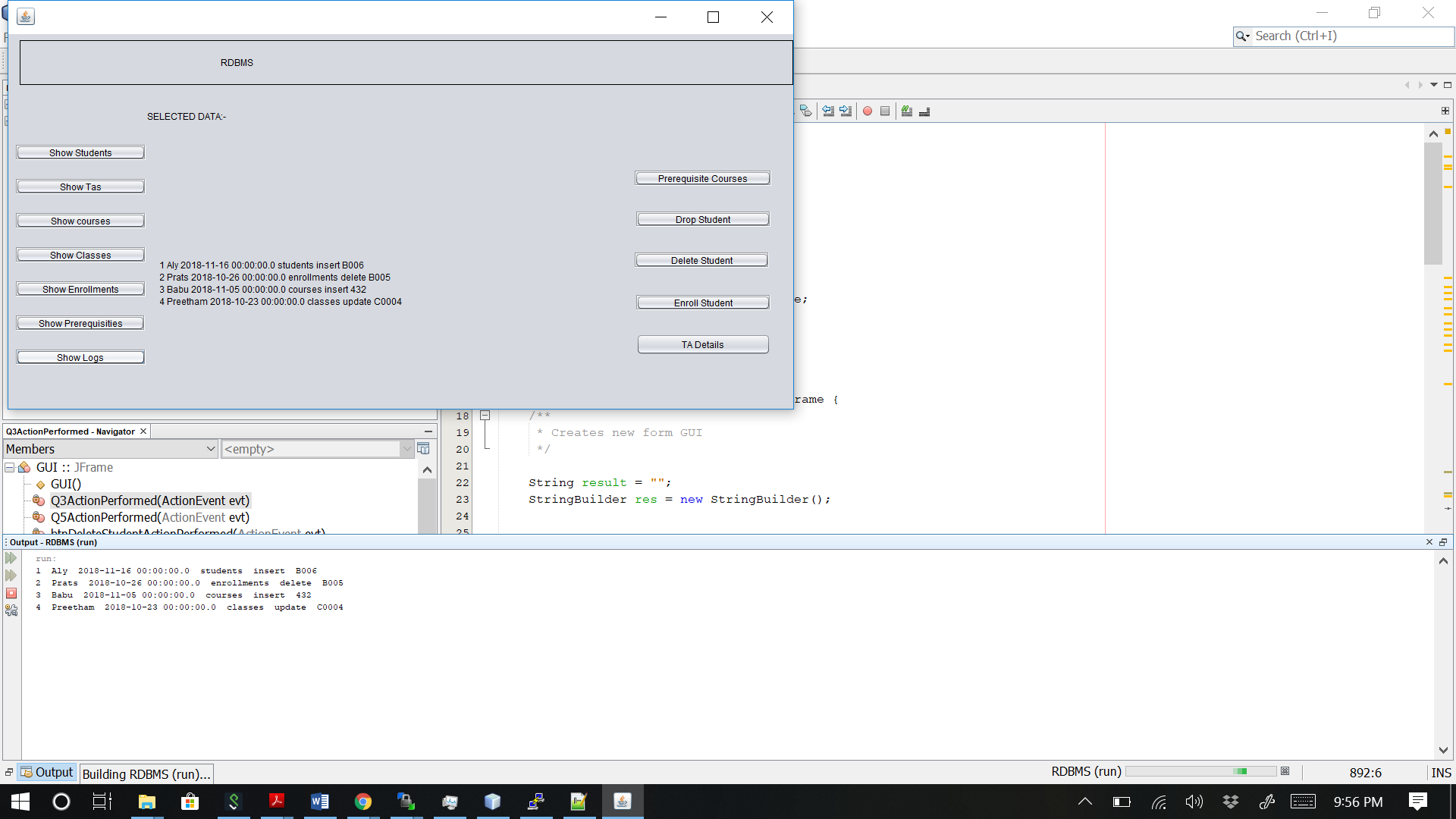
* **Click on Show Enrollments button**

****

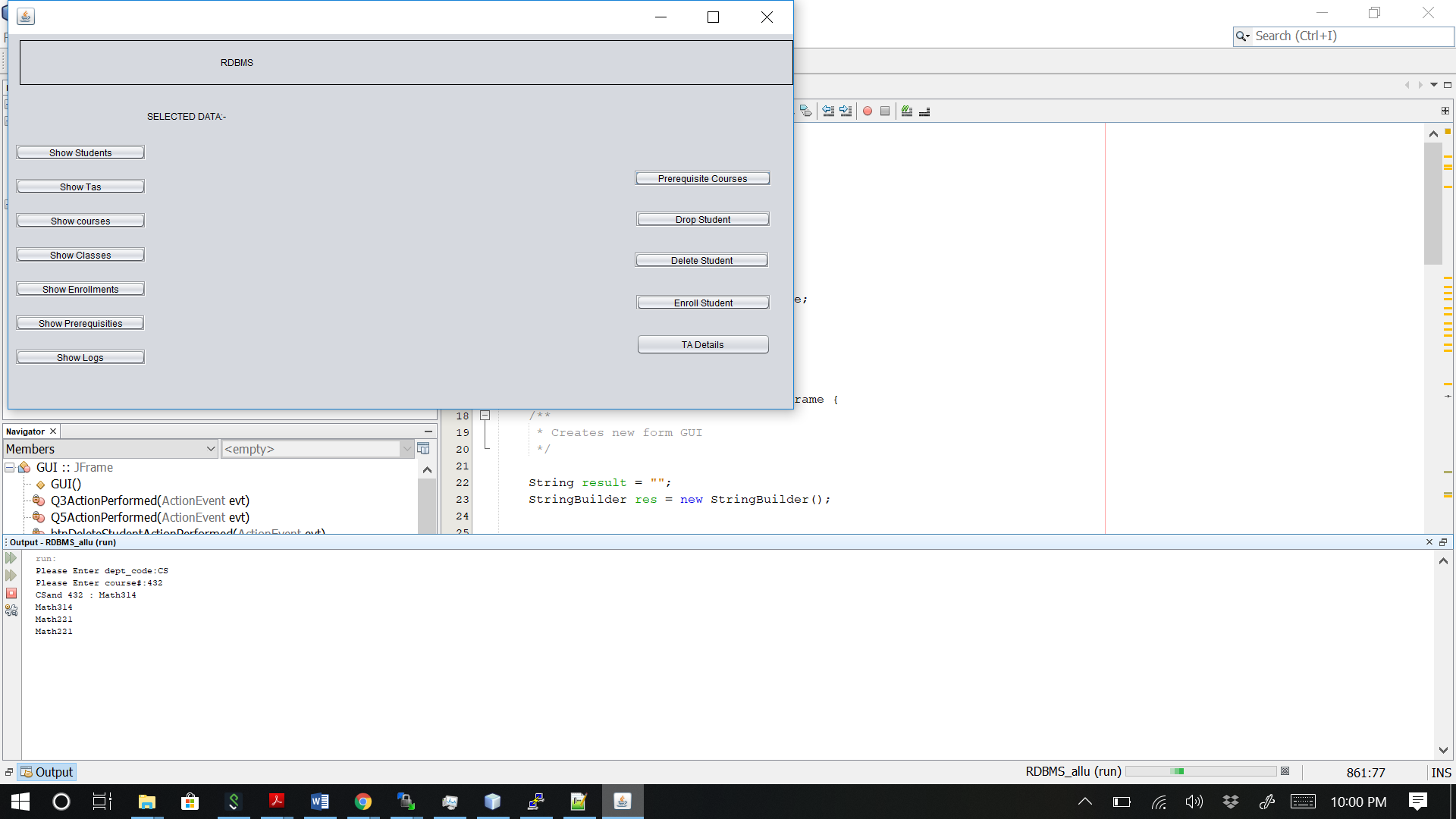
* **Click on Show Prerequisites button**

****

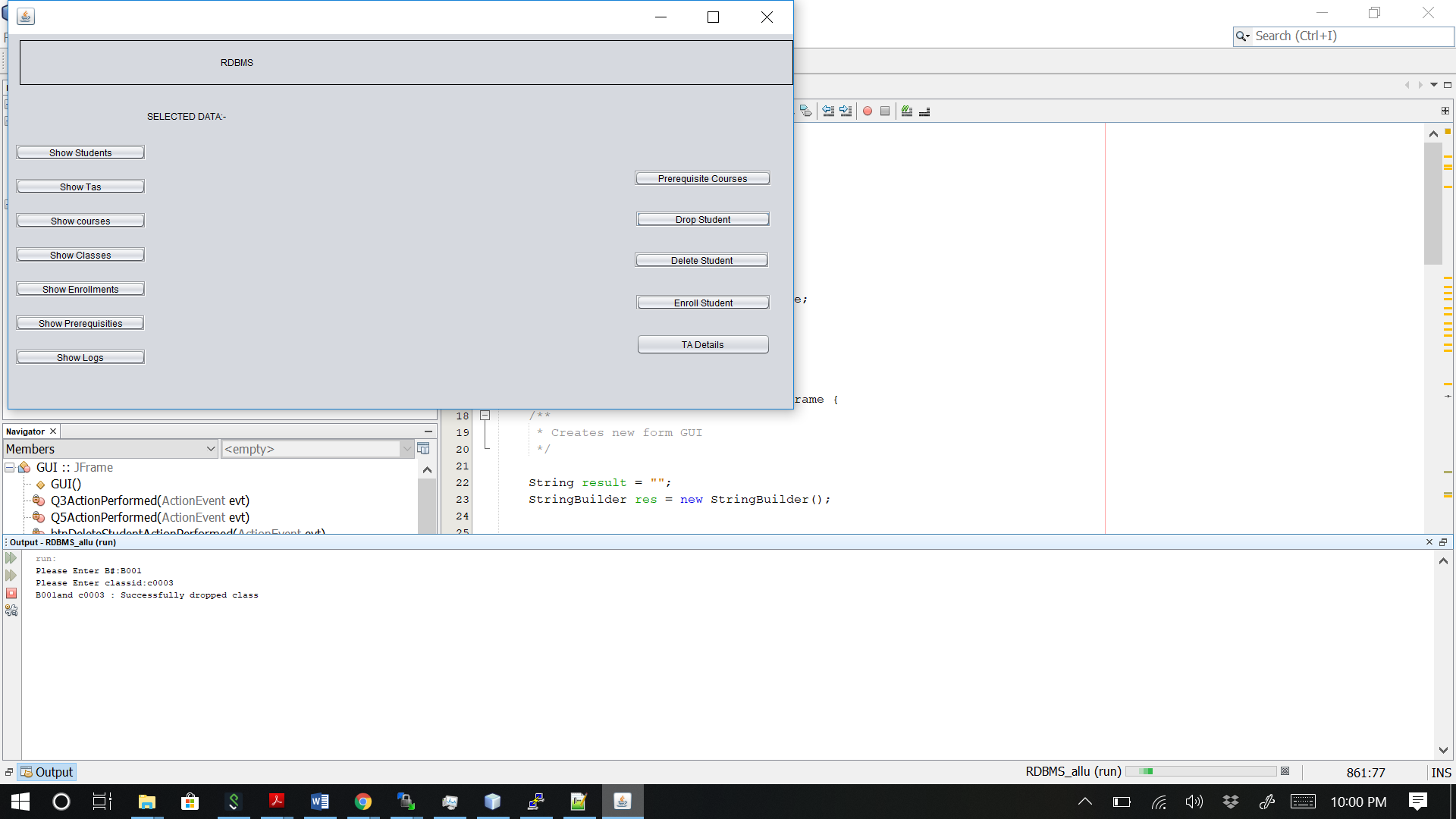
* **Click on Show Logs button**

****

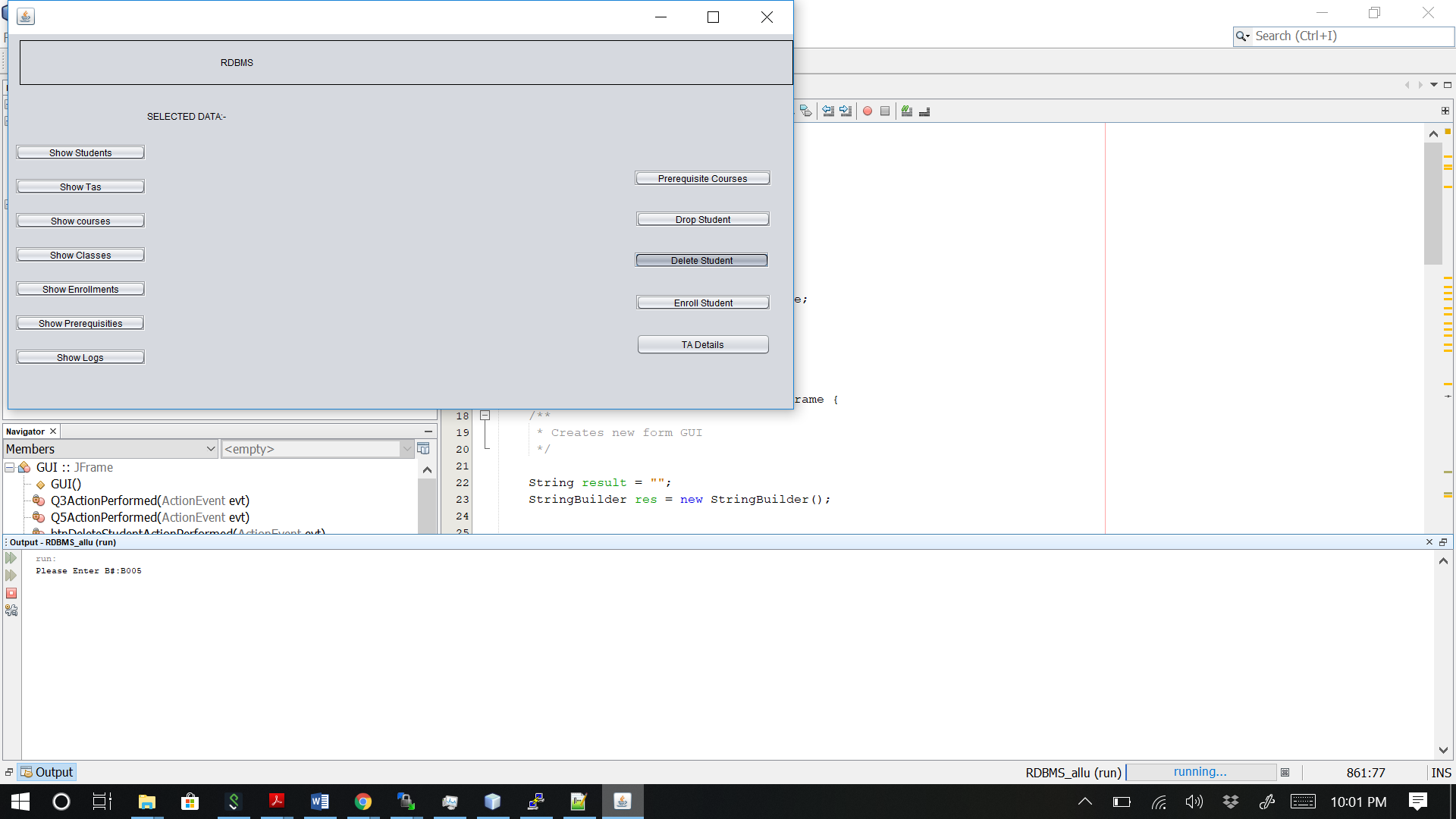
* **Click on Prerequisite Courses button**

****

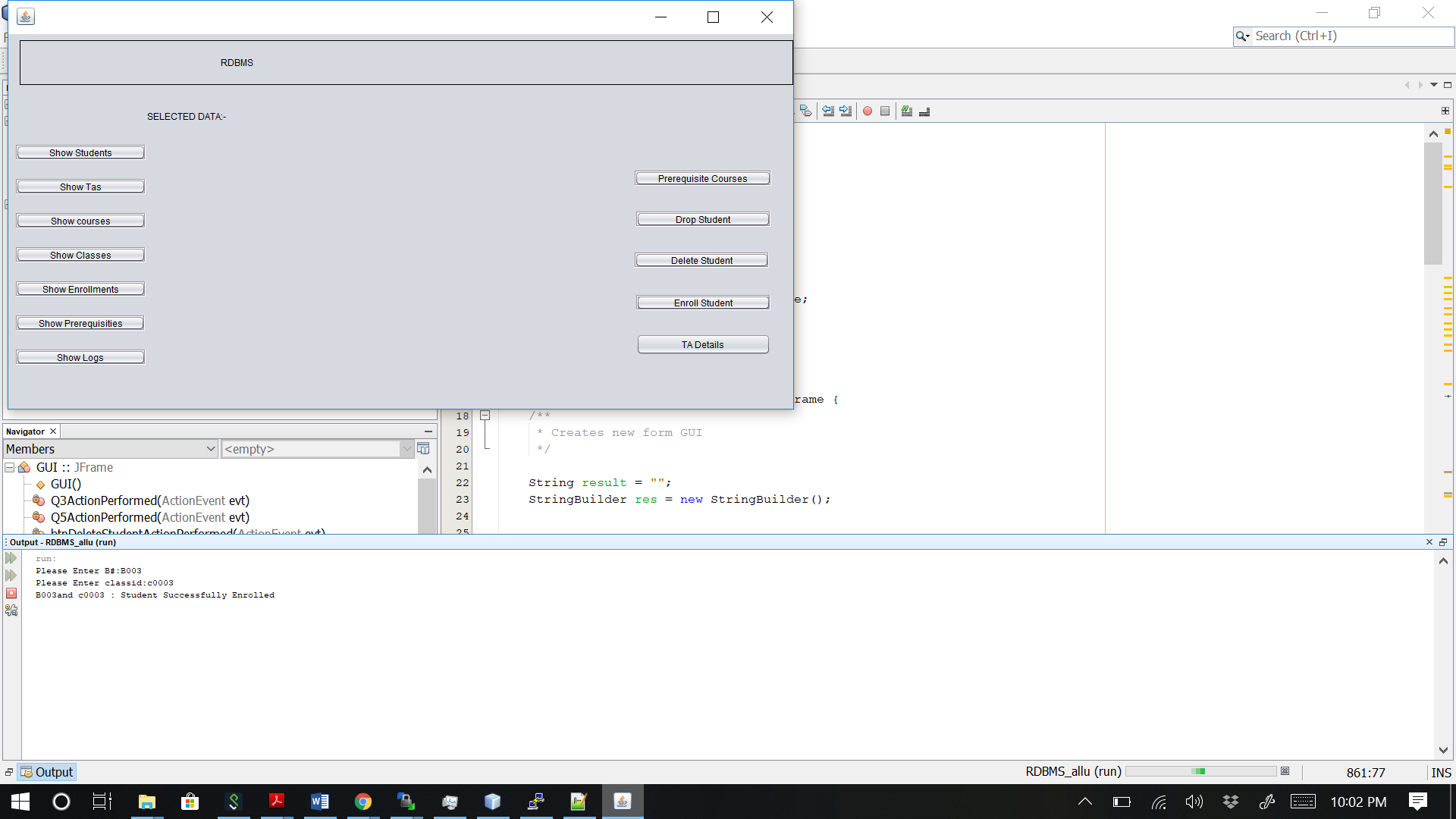
* **Click on Drop Student button**

****

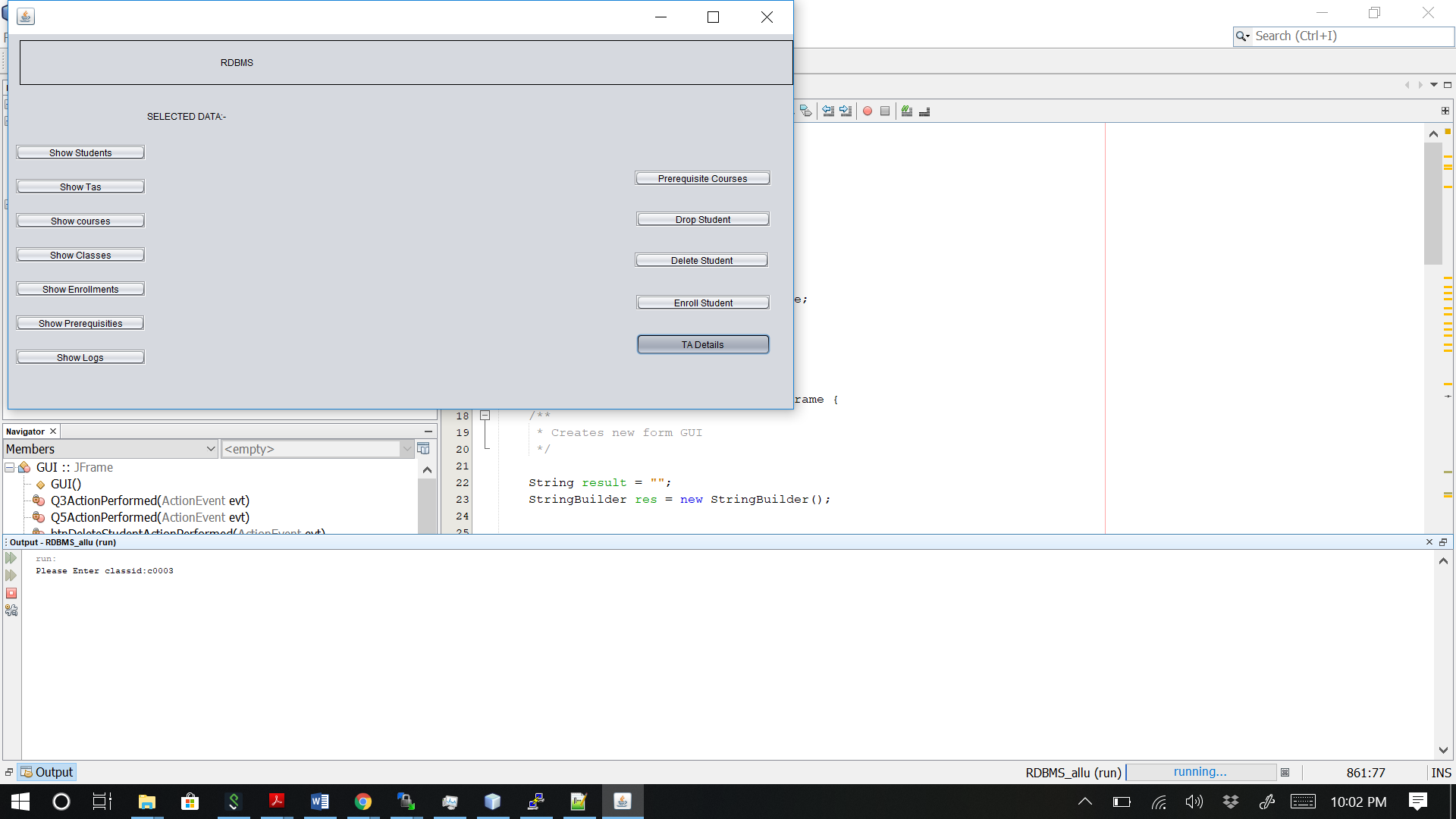
* **Click on Delete Student button**

****

* **Click on Enroll Student button**

****

* **Click on TA Details button**

****

**SOURCE CODE:**

**SCRIPT FILE**

drop table enrollments;

drop table classes;

drop table tas;

drop table prerequisites;

drop table courses;

drop table students;

drop table logs;

create table students (B# char(4) primary key check (B# like 'B%'),

first\_name varchar2(15) not null, last\_name varchar2(15) not null, status varchar2(10)

check (status in ('freshman', 'sophomore', 'junior', 'senior', 'MS', 'PhD')),

gpa number(3,2) check (gpa between 0 and 4.0), email varchar2(20) unique,

bdate date, deptname varchar2(4) not null);

create table tas (B# char(4) primary key references students,

ta\_level varchar2(3) not null check (ta\_level in ('MS', 'PhD')),

office varchar2(10));

create table courses (dept\_code varchar2(4), course# number(3)

check (course# between 100 and 799), title varchar2(20) not null,

primary key (dept\_code, course#));

create table classes (classid char(5) primary key check (classid like 'c%'),

dept\_code varchar2(4) not null, course# number(3) not null,

sect# number(2), year number(4), semester varchar2(8)

check (semester in ('Spring', 'Fall', 'Summer 1', 'Summer 2')), limit number(3),

class\_size number(3), room varchar2(10), ta\_B# char(4) references tas,

foreign key (dept\_code, course#) references courses on delete cascade,

unique(dept\_code, course#, sect#, year, semester), check (class\_size <= limit));

create table enrollments (B# char(4) references students, classid char(5) references classes,

lgrade varchar2(2) check (lgrade in ('A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-','D',

'F', 'I')), primary key (B#, classid));

create table prerequisites (dept\_code varchar2(4) not null,

course# number(3) not null, pre\_dept\_code varchar2(4) not null,

pre\_course# number(3) not null,

primary key (dept\_code, course#, pre\_dept\_code, pre\_course#),

foreign key (dept\_code, course#) references courses on delete cascade,

foreign key (pre\_dept\_code, pre\_course#) references courses on delete cascade);

create table logs (log# number(4) primary key, op\_name varchar2(10) not null, op\_time date not null,

table\_name varchar2(12) not null, operation varchar2(6) not null, key\_value varchar2(10));

insert into students values ('B001', 'Anne', 'Broder', 'junior', 3.17, 'broder@bu.edu', '17-JAN-90', 'CS');

insert into students values ('B002', 'Terry', 'Buttler', 'senior', 3.0, 'buttler@bu.edu', '28-MAY-89', 'Math');

insert into students values ('B003', 'Tracy', 'Wang', 'senior', 4.0, 'wang@bu.edu','06-AUG-93', 'CS');

insert into students values ('B004', 'Barbara', 'Callan', 'junior', 2.5, 'callan@bu.edu', '18-OCT-91', 'Math');

insert into students values ('B005', 'Jack', 'Smith', 'MS', 3.2, 'smith@bu.edu', '18-OCT-91', 'CS');

insert into students values ('B006', 'Terry', 'Zillman', 'PhD', 4.0, 'zillman@bu.edu', '15-JUN-88', 'Biol');

insert into students values ('B007', 'Becky', 'Lee', 'senior', 4.0, 'lee@bu.edu', '12-NOV-92', 'CS');

insert into students values ('B008', 'Tom', 'Baker', 'freshman', null, 'baker@bu.edu', '23-DEC-96', 'CS');

insert into students values ('B009', 'Ben', 'Liu', 'MS', 3.8, 'liu@bu.edu', '18-MAR-92', 'Math');

insert into students values ('B010', 'Sata', 'Patel', 'MS', 3.9, 'patel@bu.edu', '12-OCT-90', 'CS');

insert into students values ('B011', 'Art', 'Chang', 'PhD', 4.0, 'chang@bu.edu', '08-JUN-89', 'CS');

insert into students values ('B012', 'Tara', 'Ramesh', 'PhD', 3.98, 'ramesh@bu.edu', '29-JUL-91', 'Math');

insert into tas values ('B005', 'MS', 'EB G26');

insert into tas values ('B009', 'MS', 'WH 112');

insert into tas values ('B010', 'MS', 'EB G26');

insert into tas values ('B011', 'PhD','EB P85');

insert into tas values ('B012', 'PhD','WH 112');

insert into courses values ('CS', 432, 'database systems');

insert into courses values ('Math', 314, 'discrete math');

insert into courses values ('CS', 240, 'data structure');

insert into courses values ('Math', 221, 'calculus I');

insert into courses values ('CS', 532, 'database systems');

insert into courses values ('CS', 552, 'operating systems');

insert into courses values ('Biol', 425, 'molecular biology');

insert into classes values ('c0001', 'CS', 432, 1, 2018, 'Fall', 2, 1, 'LH 005', 'B005');

insert into classes values ('c0002', 'Math', 314, 1, 2018, 'Spring', 4, 4, 'LH 009', 'B012');

insert into classes values ('c0003', 'Math', 314, 2, 2018, 'Spring', 4, 2, 'LH 009', 'B009');

insert into classes values ('c0004', 'CS', 432, 2, 2018, 'Fall', 2, 2, 'SW 222', 'B005');

insert into classes values ('c0005', 'CS', 240, 1, 2017, 'Spring', 3, 2, 'LH 003', 'B010');

insert into classes values ('c0006', 'CS', 532, 1, 2018, 'Fall', 3, 2, 'LH 005', 'B011');

insert into classes values ('c0007', 'Math', 221, 1, 2017, 'Fall', 7, 6, 'WH 155', null);

insert into classes values ('c0008', 'CS', 552, 1, 2018, 'Fall', 1, 0, 'EB R15', null);

insert into classes values ('c0009', 'CS', 240, 1, 2018, 'Fall', 2, 1, 'EB R15', null);

insert into prerequisites values ('Math', '314', 'Math', '221');

insert into prerequisites values ('CS', '432', 'Math', '314');

insert into prerequisites values ('CS', '532', 'Math', '314');

insert into prerequisites values ('CS', '552', 'CS', '240');

insert into enrollments values ('B001', 'c0001', null);

insert into enrollments values ('B001', 'c0003', 'B');

insert into enrollments values ('B001', 'c0007', 'B+');

insert into enrollments values ('B002', 'c0002', 'B');

insert into enrollments values ('B002', 'c0007', 'B');

insert into enrollments values ('B003', 'c0004', null);

insert into enrollments values ('B003', 'c0002', 'A-');

insert into enrollments values ('B003', 'c0007', 'B');

insert into enrollments values ('B004', 'c0004', null);

insert into enrollments values ('B004', 'c0005', 'B+');

insert into enrollments values ('B004', 'c0003', 'A');

insert into enrollments values ('B004', 'c0007', 'A');

insert into enrollments values ('B005', 'c0006', null);

insert into enrollments values ('B005', 'c0002', 'B');

insert into enrollments values ('B005', 'c0007', 'B');

insert into enrollments values ('B006', 'c0006', null);

insert into enrollments values ('B006', 'c0002', 'A');

insert into enrollments values ('B006', 'c0007', 'A-');

insert into enrollments values ('B007', 'c0005', 'C');

insert into enrollments values ('B008', 'c0009', null);

**PL/SQL CODE**

1. CREATE SEQUENCE logs\_num

START WITH 100

INCREMENT BY 1

ORDER

NOCACHE

NOCYCLE;

2. CREATE PACKAGE dbproject2

AS

procedure show\_students(students\_recordset OUT SYS\_REFCURSOR);

procedure show\_tas(tas\_recordset OUT SYS\_REFCURSOR);

procedure show\_courses(courses\_recordset OUT SYS\_REFCURSOR);

procedure show\_classes(classes\_recordset OUT SYS\_REFCURSOR);

procedure show\_enrollments(enrollments\_recordset OUT SYS\_REFCURSOR);

procedure show\_prerequisites(prerequisites\_recordset OUT SYS\_REFCURSOR);

procedure show\_logs(logs\_recordset OUT SYS\_REFCURSOR);

END dbproject2;

/

CREATE OR REPLACE PACKAGE BODY dbproject2

AS

procedure show\_students(students\_recordset OUT SYS\_REFCURSOR)

is

begin

open students\_recordset for

select \* from students;

end show\_students;

procedure show\_tas(tas\_recordset OUT SYS\_REFCURSOR)

is

begin

open tas\_recordset for

select \* from tas;

end show\_tas;

procedure show\_courses(courses\_recordset OUT SYS\_REFCURSOR)

is

begin

open courses\_recordset for

select \* from courses;

end show\_courses;

procedure show\_classes(classes\_recordset OUT SYS\_REFCURSOR)

is

begin

open classes\_recordset for

select \* from classes;

end show\_classes;

procedure show\_enrollments(enrollments\_recordset OUT SYS\_REFCURSOR)

is

begin

open enrollments\_recordset for

select \* from enrollments;

end show\_enrollments;

procedure show\_prerequisites(prerequisites\_recordset OUT SYS\_REFCURSOR)

is

begin

open prerequisites\_recordset for

select \* from prerequisites;

end show\_prerequisites;

procedure show\_logs(logs\_recordset OUT SYS\_REFCURSOR)

is

begin

open logs\_recordset for

select \* from logs;

end show\_logs;

END;

/

3. create or replace package proj2\_package AS

procedure show\_ta(class\_id in classes.classid%type, show\_message OUT varchar);

END proj2\_package;

/

create or replace package body proj2\_package AS

procedure show\_ta(class\_id in classes.classid%type, show\_message OUT varchar)

is

class\_count number :=0;

tab classes.ta\_b#%type;

bnum students.B#%type;

fname students.first\_name%type;

lname students.last\_name%type;

begin

select count(\*) into class\_count from classes where classid=class\_id;

select ta\_b# into tab from classes c where class\_id = c.classid;

if(class\_count = 0) then

show\_message := 'The classid is invalid.';

elsif(tab = NULL) then

show\_message := 'The class has no TA';

else

select s.B#, s.first\_name, s.last\_name into bnum,fname,lname from students s, classes c where class\_id=classid and c.ta\_b#=s.b#;

system.output.println(bnum||' '||fname||' '||lname);

end if;

end;

END;

/

4. create or replace package proj2q4

AS

PROCEDURE show\_prereq(deptcode\_in in prerequisites.dept\_code%type,course#\_in in prerequisites.course#%type,results out varchar);

END proj2q4;

/

create or replace package body proj2q4

AS

PROCEDURE show\_prereq(deptcode\_in in prerequisites.dept\_code%type,course#\_in in prerequisites.course#%type,results out varchar)

as

deptcode varchar2(4);

course\_num number(3);

temp varchar(8000);

cursor prereq\_cursor is

(select pre\_dept\_code, pre\_course# from prerequisites where dept\_code = deptcode\_in and course# = course#\_in);

prereq\_rec prereq\_cursor%rowtype;

cursor course\_cursor is

(select \* from courses where dept\_code = deptcode\_in and course# = course#\_in);

course\_rec course\_cursor%rowtype;

BEGIN

open course\_cursor;

fetch course\_cursor into course\_rec;

if (course\_cursor%notfound) then

results := 'Course# does not exist';

else

open prereq\_cursor;

fetch prereq\_cursor into prereq\_rec;

while prereq\_cursor%found loop

deptcode := prereq\_rec.pre\_dept\_code;

course\_num := prereq\_rec.pre\_course#;

if (prereq\_rec.pre\_dept\_code IS NOT NULL) then

if (results IS NULL) then

results := deptcode || course\_num;

end if;

show\_prereq(deptcode, course\_num, temp);

if (results IS not NULL) then

results := results || chr(10) || deptcode || course\_num;

end if;

if (temp IS not NULL) then

results := results || chr(10) || temp;

end if;

else

return;

end if;

fetch prereq\_cursor into prereq\_rec;

end loop;

end if;

end;

END proj2q4;

/

5. create or replace package proj2q5 AS

procedure enroll\_student( B\_num in students.B#%type, class\_id in classes.classid%type,

show\_message OUT varchar);

END proj2q5;

/

create or replace package body proj2q5 AS procedure enroll\_student( B\_num in students.B#%type, class\_id in classes.classid%type, show\_message OUT varchar)

IS

classid\_count number := 0;

B#\_count number := 0;

already\_in number := 0;

num\_classes number := 0;

deptcode varchar2(4);

course\_num number;

num\_prereq number := 0;

class\_size number := 0;

class\_semester varchar2(6);

class\_limit number := 0;

class\_year number;

BEGIN

select count(\*) into B#\_count from students where B# = B\_num;

select count(\*) into classid\_count from classes where classid = class\_id;

IF B#\_count = 0 then

show\_message := 'The B# is invalid';

ELSIF classid\_count = 0 then

show\_message := 'The classid is invalid';

ELSE

select classes.dept\_code into deptcode from classes where classes.classid = class\_id;

select classes.class\_size into class\_size from classes where classes.classid = class\_id;

select classes.semester into class\_semester from classes where classes.classid = class\_id;

select classes.year into class\_year from classes where classes.classid = class\_id;

select classes.limit into class\_limit from classes where classes.classid = class\_id;

select classes.course# into course\_num from classes where classes.classid = class\_id;

select count(B\_num) into already\_in from enrollments

join classes on enrollments.classid = classes.classid

where classes.classid = class\_id and enrollments.B# = B\_num;

select count(\*) into num\_classes from enrollments

join classes on enrollments.classid = classes.classid

where classes.year = class\_year and classes.semester = class\_semester

and enrollments.B# = B\_num;

select count(\*) into num\_prereq from

(select prerequisites.pre\_dept\_code, prerequisites.pre\_course# from prerequisites

connect by prior pre\_dept\_code = dept\_code AND PRIOR pre\_course# = course#

START WITH dept\_code = deptcode AND course# = course\_num

minus

select classes.dept\_code, classes.course# from classes

join enrollments on classes.classid = enrollments.classid

where enrollments.B# = B\_num and enrollments.lgrade < 'D'

and enrollments.lgrade is not null);

IF class\_size >= class\_limit then

show\_message := 'The class is already full';

ELSIF already\_in > 0 then

show\_message := 'The student is already in the class';

ELSIF num\_prereq > 0 then

show\_message := 'Prerequisite not satisfied';

ELSIF num\_classes >= 5 then

show\_message := 'Students cannot be enrolled in more than five classes in the same semester';

ELSE

IF num\_classes = 4 then

show\_message := 'The student will be overloaded with the new enrollment';

END IF;

insert into enrollments values (B\_num, class\_id, null);

END IF;

END IF;

END enroll\_student;

END proj2q5;

/

1. create or replace package proj2q6

AS

procedure drop\_class(b\_num in students.b#%type, class\_id in classes.classid%type, show\_message OUT varchar);

END proj2q6;

/

create or replace package body proj2q6

AS

procedure drop\_class( b\_num in students.b#%type, class\_id in classes.classid%type, show\_message OUT varchar)

IS

cid\_count number := 0;

b#\_count number := 0;

enrolled\_count number := 0;

total\_classes number := 0;

students\_left number := 0;

BEGIN

select count(\*) into b#\_count from students where b# = b\_num;

select count(\*) into cid\_count from classes where classid = class\_id;

IF cid\_count = 0 then

show\_message := 'The classid is invalid';

ELSIF b#\_count = 0 then

show\_message := 'The B# is invalid';

ELSE

select count(\*) into enrolled\_count from enrollments

where enrollments.b# = b\_num and enrollments.classid = class\_id;

select count(\*) into total\_classes from enrollments

where enrollments.b# = b\_num;

select classes.class\_size into students\_left from classes

where classes.classid = class\_id;

IF enrolled\_count = 0 then

show\_message := 'The student is not enrolled in the class';

ELSE

IF total\_classes = 1 then

show\_message := 'This student is not enrolled in any classes';

END IF;

IF students\_left = 1 then

show\_message := 'The class now has no students';

END IF;

delete from enrollments where b# = b\_num and classid = class\_id;

IF show\_message is null then

show\_message := 'Successfully dropped class';

END IF;

END IF;

END IF;

END drop\_class;

END proj2q6;

/

1. create or replace package proj2q7 AS

PROCEDURE delete\_student(b\_num in students.b#%type, results OUT varchar);

END proj2q7;

/

create or replace package body proj2q7

AS

PROCEDURE delete\_student(b\_num in students.b#%type, results OUT varchar)

IS

cursor student\_cur is

select \* from students where b#=b\_num;

student\_rc student\_cur%rowtype;

begin

open student\_cur;

fetch student\_cur into student\_rc;

if (student\_cur%notfound) then

results:= 'The B# is invalid';

else

delete from students where b#=b\_num;

results:= 'Deleted';

end if;

end;

END proj2q7;

/

1. 🡪 Add tuple to Logs table when a student is deleted from Students table

create or replace trigger delete\_student\_after\_logs after delete on students for each row

begin

insert into logs values (logs\_num.NEXTVAL, USER, sysdate, 'students', 'delete', :old.b#);

end;

/

🡪 Add tuple to Logs table when a tuple is inserted into the Enrollments table

create or replace trigger insert\_enrollments\_after\_logs after insert on enrollments for each row

begin

insert into logs values (logs\_num.NEXTVAL, USER, sysdate, 'enrollments', 'insert', :new.B# || ',' || :new.classid);

end;

/

🡪 Add tuple to Logs table when a tuple is deleted from the Enrollments table

create or replace trigger DELETE\_ENROLLMENTS\_ADD\_LOGS

after

delete on enrollments

for each row

begin

insert into logs values (logs\_num.NEXTVAL, USER, sysdate, 'enrollments', 'delete', :old.b# || ',' || :old.classid);

end;

/

🡪 Delete all tuples in the Enrollments table involving the student when a student is deleted

create or replace trigger delete\_enrollments\_students

before

delete on students

for each row

begin

delete from enrollments where enrollments.B# = :old.B#;

end;

/

🡪 Update the values caused by successfully enrolling a student into a class

create or replace trigger update\_logs\_insert\_students

after

insert on students

for each row

begin

insert into logs values (logs\_num.NEXTVAL, USER, sysdate, 'students', 'insert', :new.B#);

end;

/

**INTERFACE CODE**

**JAVA CODE**

**myconnection.java**

import java.sql.\*;

import oracle.jdbc.\*;

import java.math.\*;

import java.io.\*;

import java.awt.\*;

import oracle.jdbc.pool.OracleDataSource;

public class Myconnection {

public static void main (String args []) throws SQLException {

try

{

//Connecti to Oracle server. Need to replace username and

//password by your username and your password. For security

//consideration, it's better to read them in from keyboard.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

// Query

Statement stmt = conn.createStatement ();

// Save result

ResultSet rset;

rset = stmt.executeQuery ("select \* from students");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

System.out.print (rset.getString (4)+" ");

System.out.print (rset.getString (5)+" ");

System.out.println (rset.getString (6)+" ");

}

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

}

**GUI.java**

import java.sql.CallableStatement;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.sql.Types;

import java.util.ArrayList;

import java.util.List;

import oracle.jdbc.OracleTypes;

import oracle.jdbc.pool.OracleDataSource;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.\*;

public class GUI extends javax.swing.JFrame {

/\*\*

\* Creates new form GUI

\*/

String result = "";

StringBuilder res = new StringBuilder();

public GUI() {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

btnShowStudents = new javax.swing.JButton();

btnShowTas = new javax.swing.JButton();

btnShowCourses = new javax.swing.JButton();

btnShowClasses = new javax.swing.JButton();

btnShowEnrolllments = new javax.swing.JButton();

btnShowPrerequisities = new javax.swing.JButton();

btnShowLogs = new javax.swing.JButton();

LableRDBMS = new javax.swing.JLabel();

labelSelectedData = new javax.swing.JLabel();

Q5 = new javax.swing.JButton();

btnques4 = new javax.swing.JButton();

btnDropStudent = new javax.swing.JButton();

btnDeleteStudent = new javax.swing.JButton();

msgbox = new javax.swing.JLabel();

Q3 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

btnShowStudents.setText("Show Students");

btnShowStudents.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowStudents.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowStudentsActionPerformed(evt);

}

});

btnShowTas.setText("Show Tas");

btnShowTas.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowTas.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowTasActionPerformed(evt);

}

});

btnShowCourses.setText("Show courses");

btnShowCourses.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowCourses.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowCoursesActionPerformed(evt);

}

});

btnShowClasses.setText("Show Classes");

btnShowClasses.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowClasses.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowClassesActionPerformed(evt);

}

});

btnShowEnrolllments.setText("Show Enrollments");

btnShowEnrolllments.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowEnrolllments.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowEnrolllmentsActionPerformed(evt);

}

});

btnShowPrerequisities.setText("Show Prerequisities");

btnShowPrerequisities.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowPrerequisities.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowPrerequisitiesActionPerformed(evt);

}

});

btnShowLogs.setText("Show Logs");

btnShowLogs.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnShowLogs.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnShowLogsActionPerformed(evt);

}

});

LableRDBMS.setText(" RDBMS");

LableRDBMS.setBorder(javax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0, 0)));

labelSelectedData.setText(" SELECTED DATA:-");

Q5.setText("Enroll Student");

Q5.setBorder(javax.swing.BorderFactory.createEtchedBorder());

Q5.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

Q5ActionPerformed(evt);

}

});

btnques4.setText("Prerequisite Courses");

btnques4.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnques4.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnques4ActionPerformed(evt);

}

});

btnDropStudent.setText("Drop Student");

btnDropStudent.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnDropStudent.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDropStudentActionPerformed(evt);

}

});

btnDeleteStudent.setText("Delete Student");

btnDeleteStudent.setBorder(javax.swing.BorderFactory.createEtchedBorder());

btnDeleteStudent.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnDeleteStudentActionPerformed(evt);

}

});

msgbox.setBorder(new javax.swing.border.MatteBorder(null));

Q3.setText("TA Details");

Q3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

Q3ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(15, 15, 15)

.addComponent(LableRDBMS, javax.swing.GroupLayout.PREFERRED\_SIZE, 1020, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(150, 150, 150)

.addComponent(labelSelectedData, javax.swing.GroupLayout.PREFERRED\_SIZE, 650, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(btnShowStudents, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowTas, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowCourses, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowClasses, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowEnrolllments, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowPrerequisities, javax.swing.GroupLayout.PREFERRED\_SIZE, 170, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnShowLogs, javax.swing.GroupLayout.PREFERRED\_SIZE, 171, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(19, 19, 19)

.addComponent(msgbox, javax.swing.GroupLayout.PREFERRED\_SIZE, 608, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(Q5, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 177, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(Q3, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 177, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(btnDropStudent, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 177, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(layout.createSequentialGroup()

.addGap(18, 18, 18)

.addComponent(btnques4, javax.swing.GroupLayout.PREFERRED\_SIZE, 180, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(18, 18, 18)

.addComponent(btnDeleteStudent, javax.swing.GroupLayout.PREFERRED\_SIZE, 177, javax.swing.GroupLayout.PREFERRED\_SIZE))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(8, 8, 8)

.addComponent(LableRDBMS, javax.swing.GroupLayout.PREFERRED\_SIZE, 59, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(33, 33, 33)

.addComponent(labelSelectedData)

.addGap(30, 30, 30)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(btnShowStudents)

.addGap(25, 25, 25)

.addComponent(btnShowTas)

.addGap(25, 25, 25)

.addComponent(btnShowCourses)

.addGap(25, 25, 25)

.addComponent(btnShowClasses)

.addGap(25, 25, 25)

.addComponent(btnShowEnrolllments)

.addGap(25, 25, 25)

.addComponent(btnShowPrerequisities)

.addGap(25, 25, 25)

.addComponent(btnShowLogs))

.addComponent(msgbox, javax.swing.GroupLayout.PREFERRED\_SIZE, 348, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addGap(34, 34, 34)

.addComponent(btnques4)

.addGap(34, 34, 34)

.addComponent(btnDropStudent)

.addGap(34, 34, 34)

.addComponent(btnDeleteStudent)

.addGap(36, 36, 36)

.addComponent(Q5)

.addGap(31, 31, 31)

.addComponent(Q3))))

);

pack();

}// </editor-fold>

private void btnShowStudentsActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_students(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

System.out.print (rset.getString (4)+" ");

System.out.print (rset.getString (5)+" ");

System.out.println (rset.getString (6)+" ");

System.out.println (rset.getString (7)+" ");

System.out.println (rset.getString (8)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append(rset.getString (4)+" ");

res.append(rset.getString (5)+" ");

res.append(rset.getString (6)+" ");

res.append(rset.getString (7)+" ");

res.append(rset.getString (8)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

private void btnShowTasActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_tas(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n");}

}

private void btnShowCoursesActionPerformed(java.awt.event.ActionEvent evt) {

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_courses(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n");}

}

private void btnShowClassesActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_classes(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

System.out.print (rset.getString (4)+" ");

System.out.print (rset.getString (5)+" ");

System.out.println (rset.getString (6)+" ");

System.out.println (rset.getString (7)+" ");

System.out.println (rset.getString (8)+" ");

System.out.println (rset.getString (9)+" ");

System.out.println (rset.getString (10)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append(rset.getString (4)+" ");

res.append(rset.getString (5)+" ");

res.append(rset.getString (6)+" ");

res.append(rset.getString (7)+" ");

res.append(rset.getString (8)+" ");

res.append(rset.getString (9)+" ");

res.append(rset.getString (10)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

private void btnShowEnrolllmentsActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_enrollments(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

private void btnShowPrerequisitiesActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_prerequisites(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

System.out.print (rset.getString (4)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append(rset.getString (4)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

private void btnShowLogsActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

//Connecting to Oracle server.

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:acad111");

Connection conn = ds.getConnection("asaraka1, "Aafemh14");

//Prepare to call stored procedure:

CallableStatement cs = conn.prepareCall("begin dbproject2.show\_logs(?); end;");

//register the out parameter (the first parameter)

cs.registerOutParameter(1, OracleTypes.CURSOR);

// Input gpg threshold from keyboard

/\*Double tgpa = Double.parseDouble("2.5");

cs.setDouble(2, tgpa);\*/

// execute and retrieve the result set

cs.execute();

ResultSet rset = (ResultSet)cs.getObject(1);

// print the results

res.append("<html><br>");

// Print

while (rset.next ()) {

System.out.print (rset.getString (1)+" ");

System.out.print (rset.getString (2)+" ");

System.out.print (rset.getString (3)+" ");

System.out.print (rset.getString (4)+" ");

System.out.print (rset.getString (5)+" ");

System.out.println (rset.getString (6)+" ");

res.append(rset.getString (1)+" ");

res.append(rset.getString (2)+" ");

res.append(rset.getString (3)+" ");

res.append(rset.getString (4)+" ");

res.append(rset.getString (5)+" ");

res.append(rset.getString (6)+" ");

res.append("<br>");

}

res.append("</html>");

result= res.toString();

msgbox.setText(result);

}

catch (SQLException e) { e.printStackTrace(System.out);}

catch (Exception e) {System.out.println (e);}

}

private void btnques4ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

// Input sid from keyboard

//BufferedReader readKeyBoard;

// String sid;

// String dept\_co;

// int courseNo;

//textInputMsg.setText("Please Enter dept\_code and Course# below:");

//dept\_code = Getinput1.getText();

//courseNo = getinput2.getText();

//(Number)rateField.getValue()

//sid = readKeyBoard.readLine();

// Input sid from keyboard

BufferedReader readKeyBoard;

String dept\_co;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter dept\_code:");

dept\_co = readKeyBoard.readLine();

int courseNo;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter course#:");

courseNo = Integer.parseInt(readKeyBoard.readLine());

CallableStatement cs = conn.prepareCall("begin proj2q4.show\_prereq(:1,:2,:3); end;");

//set the in parameter (the first parameter)

cs.setString(1, dept\_co);

cs.setInt(2, courseNo);

//register the out parameter (the second parameter)

cs.registerOutParameter(3, Types.VARCHAR);

//execute the stored procedure

cs.executeQuery();

//get the out parameter result.

String status = cs.getString(3);

System.out.println( dept\_co + "and " + courseNo + " : "+status);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n" + e.getMessage());}

}

private void Q5ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

BufferedReader readKeyBoard;

String bnum;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter B#:");

bnum = readKeyBoard.readLine();

String classid;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter classid:");

classid = readKeyBoard.readLine();

CallableStatement cs = conn.prepareCall("begin proj2q5.enroll\_student(:1,:2,:3); end;");

//

//set the in parameter (the first parameter)

cs.setString(1, bnum);

cs.setString(2, classid);

//register the out parameter (the second parameter)

cs.registerOutParameter(3, Types.VARCHAR);

//execute the stored procedure

cs.executeQuery();

//get the out parameter result.

String status = cs.getString(3);

System.out.println( bnum + "and " + classid + " : "+status);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n" + e.getMessage());}

}

private void btnDropStudentActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here5

try

{

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

BufferedReader readKeyBoard;

String bnum;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter B#:");

bnum = readKeyBoard.readLine();

String classid;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter classid:");

classid = readKeyBoard.readLine();

CallableStatement cs = conn.prepareCall("begin proj2q6.drop\_class(:1,:2,:3); end;");

//

//set the in parameter (the first parameter)

cs.setString(1, bnum);

cs.setString(2, classid);

//register the out parameter (the second parameter)

cs.registerOutParameter(3, Types.VARCHAR);

//execute the stored procedure

cs.executeQuery();

//get the out parameter result.

String status = cs.getString(3);

System.out.println( bnum + "and " + classid + " : "+status);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n" + e.getMessage());}

}

private void btnDeleteStudentActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

BufferedReader readKeyBoard;

String bnum;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter B#:");

bnum = readKeyBoard.readLine();

CallableStatement cs = conn.prepareCall("begin proj2q7.delete\_student(:1,:2); end;");

//

//set the in parameter (the first parameter)

cs.setString(1, bnum);

//register the out parameter (the second parameter)

cs.registerOutParameter(2, Types.VARCHAR);

//execute the stored procedure

cs.executeQuery();

//get the out parameter result.

String status = cs.getString(2);

System.out.println( bnum + "and " + " : "+status);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n" + e.getMessage());}

}

private void Q3ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try

{

OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();

ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");

Connection conn = ds.getConnection("asaraka1", "Aafemh14");

BufferedReader readKeyBoard;

String classid;

readKeyBoard = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please Enter classid:");

classid = readKeyBoard.readLine();

CallableStatement cs = conn.prepareCall("begin proj2\_package.show\_ta(:1,:2); end;");

//

//set the in parameter (the first parameter)

cs.setString(1, classid);

//register the out parameter (the second parameter)

cs.registerOutParameter(2, Types.VARCHAR);

//execute the stored procedure

cs.executeQuery();

//get the out parameter result.

String status = cs.getString(2);

System.out.println( classid + " : "+status);

//close the result set, statement, and the connection

cs.close();

conn.close();

}

catch (SQLException ex) { System.out.println ("\n\*\*\* SQLException caught \*\*\*\n" + ex.getMessage());}

catch (Exception e) {System.out.println ("\n\*\*\* other Exception caught \*\*\*\n" + e.getMessage());}

}

public static void main(String args[]) {

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new GUI().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JLabel LableRDBMS;

private javax.swing.JButton Q3;

private javax.swing.JButton Q5;

private javax.swing.JButton btnDeleteStudent;

private javax.swing.JButton btnDropStudent;

private javax.swing.JButton btnShowClasses;

private javax.swing.JButton btnShowCourses;

private javax.swing.JButton btnShowEnrolllments;

private javax.swing.JButton btnShowLogs;

private javax.swing.JButton btnShowPrerequisities;

private javax.swing.JButton btnShowStudents;

private javax.swing.JButton btnShowTas;

private javax.swing.JButton btnques4;

private javax.swing.JLabel labelSelectedData;

private javax.swing.JLabel msgbox;

// End of variables declaration

}