Assignment #4

Objectives:

- Use database to retrieve data.
- Write database programs that insert, update, and query data in a relational database

Since all of you share the same database server and may use the same database (schema), all tables of your work need to have a suffix of your team number. For example, you need a table "Student" to store student's information and your team number is 01, so the table name should be "Student_01". If you want to get all student's data, the SQL statement is "SELECT * FROM Student_01".

Following tables are tables you need to create in your database.

Student			
Student_id	Student_name	Student_current_credits	Student_max_credits
CHAR(9)	VARCHAR(10)	int	int

Course		
Course_id	Course_name	Course_credits
CHAR(9)	VARCHAR(10)	int

Enroll		
Student _id	Course_id	Grade
CHAR(9)	CHAR(9)	int

- Student.Student_id, Course.Course_id, and (Enroll.Student_id, Enroll.Course_id) are primary keys.

Class description:

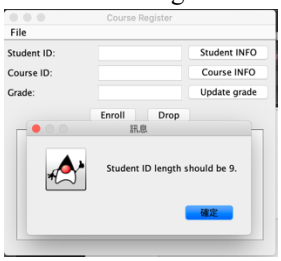
Student	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	studentID The student ID.
String	studentName The student's name
int	currentCredits current credits
int	maxCredits credits limit
Constructor	
Student(String studentID, String name, int credit, int max) **Constructs a student object with given values.**	
Instance methods	
-	Getter: studentID, studentName, currentCredits, maxCredits. Setter: currentCredits. Note that setCurrentCredits(…) must update the corresponding value in the database.

String	info()
	Return a String description of the student as following example. Hint: join Enroll and Course.
	`Student ID: 107356010`
	`Stuednt Name: Tester`
	`Credits: 2/25`
	`Enrolled courses:`
	`306049001-OOP-100`

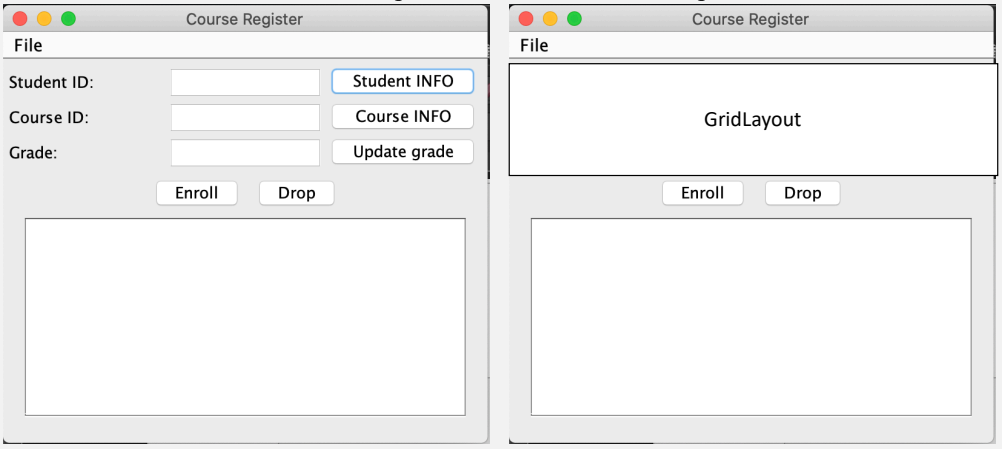
Course	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	courseID The course number of this course.
String	courseName The course name of this course.
int	credits The credits of the course.
Constructor	
Course(String id, String name, int credits) **Constructs a Course object with given od, name, and credits.**	
Instance methods	
-	Getter for all attributes. No setter required.
String	toString() Return a String description of the course. (See output in figure 1.)

Register	
Modifier and type	**Method (or Variable) and description**
Constructor	
Register() **Constructs a Register object**	
void	addStudent(String id, String name) Insert a student record to the database.
void	addCourse(String id, String name, int credits) Insert a course record to the database.
Student	findStudent(String studentID) Find the student in the database by **studentID**. If found, returns the student as a Student object. Otherwise, returns null.
Course	findCourse(String courseID) Find the course in the database by **courseId**. If found, returns the course as a Course object. Otherwise, returns null.
boolean	enrollCourse(String studentID, String courseID) 1. Find the student by given id and find the course by given id in the database. 2. If both can be found: A. Check if (1) the currentCredits of the student after adding the course is no greater maxCredits and (2) the student hasn't enrolled in the course. B. If all conditions are met. Do C and D. C. Adjust currentCredits of the student

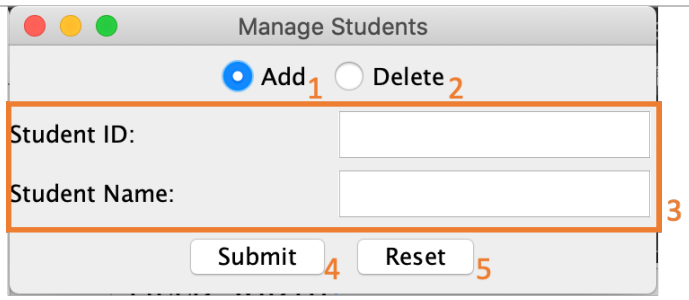
	D. Add an enroll record to Enroll and set Grade as 0 then return true. 3. Return false if there is any wrong operation.
boolean	dropCourse(String studentID, String courseID) 1. Find the student by given id and find the object by given id in the database. 2. If both can be found: A. Check if the student is enrolled in the course by the courseID. B. If student is in the course, do C and D C. Adjust currentCredits of the student D. Delete the enroll record in Enroll. 3. Return false if there is any wrong operation.
void	removeStudent(String studentID) Remove the student object in the list by given student id.
void	removeCourse(String courseID) Remove the course object in the list by given course id.
void	updateGrade(String studentID, String courseID, int grade) Update the grade by given value in Enroll table.

RegisterFrame Extends from JFrame	
Modifier and type	**Method (or Variable) and description**
Constant variable	
int	FRAME_WIDTH The width of the frame. 400
int	FRAME_HEIGHT The height of the frame. 360
int	FIELD_WIDTH The width of the TextField. 10
Int	AREA_WIDTH The width of the TextArea. 30
int	AREA_HEIGHT The height of the TextArea. 10
Instance variable	
Register	register
JPanel	panel
JLabel	studentIDLabel, courseIDLabel, gradeLabel
JTextField	studentIDField, courseIDField, gradeField
JButton	studentInfoButton, courseInfoButton, enrollButton, dropButton, updateButton
JScrollPane	scrollPane
JTextArea	outputTextArea
JMenuBar	menuBar
Constructor and Description	
RegisterFrame() Constructs a RegisterFrame. In the constructor you have to set the GUI title as "Course Register", and set the frame size by constant variables, FRAME_WIDTH and FREAME_HEIGHT. And then call all help methods to create a GUI. Don't forget to add the menu bar to the frame.	
Instance methods	
void	createStudentIDComp() Instantiates a JLabel studentIDLabel, a JTextField studentIDField with FIELD_WIDTH, and a JButton studentInfoButton and define an inner class which implements ActionListener then assign it to studentInfoButton. When the button is clicked, it will perform the corresponding jobs: 1. Get the input value of studentIDField 2. If the length of input String is not 9, use JOptionPane.showMessageDialog(…) to show a message "Student ID length should be 9. " (see the following figure.)  3. Find the student in the database by given value. 4. If the object isn't null, append info() to outputTextArea. Otherwise, append "False". *Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)
void	createCourseIDComp() Instantiates a JLabel courseIDLabel, a JTextField courseIDField with FIELD_WIDTH, and a JButton

<table>
<tr><td></td><td>courseInfoButton and define an inner class which implements ActionListener then assign it to courseInfoButton. When the button is clicked, it will perform the corresponding jobs:
1. Get the input value of courseIDField
2. If the length of input String is not 9, use JOptionPane.showMessageDialog(…) to show a message “Course ID length should be 9. ”
3. Find the course in database by given value.
4. If the object isn’t null, append toString() to outputTextArea. Otherwise, append “False”.
*Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)</td></tr>
<tr><td>void</td><td>createGradeComp()
Instantiates a JLabel gradeLabel, a JTextField gradeField with FIELD_WIDTH, and a JButton updateButton and define an inner class which implements ActionListener then assign it to updateButton. When the button is clicked, it will perform the corresponding jobs:
1. Get the input value of gradeField
2. If the input value can’t be parsed to integer, use JOptionPane.showMessageDialog(…) to show a message “Grade must be integer. ”
3. Update the grade in the database by given value.
*Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)</td></tr>
<tr><td>void</td><td>createEnrollBtn()
Instantiates a JButton enrollButton and define an inner class which implements ActionListener then assign it to the button. When the button is clicked, it will perform the corresponding jobs:
1. Get the input values of studentIDField and courseIDField
2. If any ID length is not 9, use JOptionPane.showMessageDialog(…) to show a message “Course /Student ID length should be 9. ”
3. Execute register.enrollCourse(…)
4. Use the return value from 2 and append the result to outputTextArea based on the following: True: “studentID enrolled in courseID” / False: “False”
* Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)</td></tr>
<tr><td>void</td><td>createDropBtn()
Instantiates a JButton dropButton and define an inner class which implements ActionListener then assign it to the button. When the button is clicked, it will perform the corresponding jobs:
1. Get the input values of studentIDField and courseIDField
2. If any ID length is not 9, use JOptionPane.showMessageDialog(…) to show a message “Course /Student ID length should be 9. ”
3. Execute register.dropCourse(…)
4. Use the return value from 2 and append the result to outputTextArea based on the following: True: “studentID dropped out of courseID” / False: “False”
* Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)</td></tr>
<tr><td>void</td><td>createOutputArea()
1. Instantiates the JTextArea and JScrollPane, the add the text area to the scroll pane.
2. Sets the line-wrapping policy of the text area. If the lines are too long to fit within the allocated with, they will be wrapped.</td></tr>
<tr><td>void</td><td>createMenuBar()
<table><tr><td>JMenu</td><td>File</td></tr><tr><td rowspan="3">JMenuItem</td><td>Student</td></tr><tr><td>Course</td></tr><tr><td>Exit</td></tr></table>Instantiates all components in the menu bar in this method or by using help methods.
1. When “Student” is clicked, it will perform the corresponding jobs:</td></tr>
</table>

	A. Instantiate a StudentFrame object which title is “Manage Students” and display it next to “Course Register”. Hint: JFrame.setLocation(x, y) 2. When “Course” is clicked, it will perform the corresponding jobs: A. Instantiate a CourseFrame object which title is “Manage Courses” and display it next to “Course Register”. Hint: JFrame.setLocation(x, y) 3. When “Exit” is clicked, end the program.
void	createPanel() Instantiates a JPanel and add all components into it, then add the panel to the frame. 
-	Getter: register

StudentFrame **Extends from JFrame**	
Modifier and type	**Method (or Variable) and description**
Constant variable	
int	FRAME_WIDTH The width of the frame. 360
int	FRAME_HEIGHT The height of the frame. 160
int	FIELD_WIDTH The width of the TextField. 10
Instance variable	
JRadioButton	addButton, deleteButton
JLabel	studentIDLabel, studentNameField
JTextField	studentIDField, studentNameField
JButton	submitButton, resetButton
Constructor and Description	
StudentFrame() Constructs a StudentFrame. Set the frame size by constant variables, FRAME_WIDTH and FREAME_HEIGHT. And then call all help methods to create a GUI.	
GUI instructions	



Please use help methods to implement this frame.

Programming instructions:

A. User can only either select "Add" or "Delete" at one time.
B. The default selected radio button is "Add".
C. When "Add" is selected, all text field are editable.
D. When "Delete" is selected, the text field for student name is not editable.
E. When "Submit" is clicked, identify which radio button (add or delete) is selected then perform the following jobs:
   1. If "Add" is selected, get input values from text fields then add a student record to the database.
   2. If "Delete" is selected, get the input value of student id then remove the student from the database.
F. When "Reset" is clicked, it will empty all textfields.
G. ***Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)**

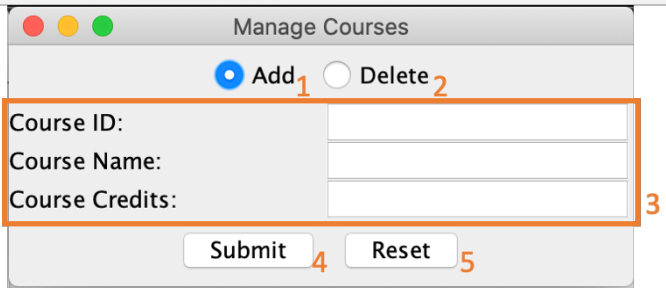
Hint:

A. You can define two methods as enableFields() and disableFields() to control the text field is editable or not.
B. Call setSelected(boolean) for programming instruction B.
C. Use isSelected() to identify which radio button is selected for programming instruction E.

Layout instructions:

3 panel is required:

A. A panel which contain component 1 and 2, and is at the top of the container.
B. A panel (component 3) which applied gird layout (2x2), and is at the center of the container.
C. A panel which contain component 4 and 5, and is at the bottom of the container.

CourseFrame Extends from JFrame	
Modifier and type	**Method (or Variable) and description**
Constant variable	
int	FRAME_WIDTH The width of the frame. 360
int	FRAME_HEIGHT The height of the frame. 160
int	FIELD_WIDTH The width of the TextField. 10
Instance variable	
JRadioButton	addButton, deleteButton
JLabel	courseIDLabel, courseNameLabel, courseCreditsLabel
JTextField	courseIDField, courseNameField, courseCreditsField
JButton	submitButton, resetButton
Constructor and Description	
CourseFrame() Constructs a CourseFrame. Set the frame size by constant variables, FRAME_WIDTH and FREAME_HEIGHT. And then call all help methods to create a GUI.	
GUI instructions	
 Please use help methods to implement this frame.	
Programming instructions: H. User can only either select "Add" or "Delete" at one time. I. The default selected radio button is "Add". J. When "Add" is selected, all text fields are editable. K. When "Delete" is selected, the text field for course name and credits are not editable. L. When "Submit" is clicked, identify which radio button (add or delete) is selected then perform the following jobs: 1. If "Add" is selected, get input values from text fields then add a course to the database. 2. If "Delete" is selected, get the input value of course id then remove the course from the database. M. When "Reset" is clicked, it will empty all text fields. ***Catch the SQLException and show the message by using JOptionPane.showMessageDialog(…)** **Hint:** A. You can define two methods as enableFields() and disableFields() to control the text field is editable or not. B. Call setSelected(boolean) for programming instruction B. C. Use isSelected() to identify which radio button is selected for programming instruction E.	
Layout instructions: 3 panel is required: A. A panel which contain component 1 and 2, and is at the top of the container. B. A panel (component 3) which applied gird layout (3x2), and is at the center of the container.	

C. A panel which contain component 4 and 5, and is at the bottom of the container.

```
//Please add the following method in StudentFrame and CourseFrame.
private Register getRegisterFromRegisterFrame() {
	for(Frame frame: JFrame.getFrames()) {
		if(frame.getTitle().equals("Course Register")) {
			RegisterFrame registerFrame = (RegisterFrame) frame;
			return registerFrame.getRegister();
		}
	}
	return null;
}
```

RegisterViewer
main(String args[]) Use the RegisterFrame to create a GUI, then set close operation by JFrame.EXIT_ON_CLOSE and make the GUI visible.

Submission instruction:

1. Export your assignment as an executable JAR file.
2. Upload you the JAR file and the source code as ZIP file to WM5. (Two files in total.)

Teammate evaluation: If you work in pair, please fulfill this form: https://forms.gle/rv372cVuR6Pt7xGu7

Reminder: Please zip the whole project. Each team submits your work by one. Your project and file name are supposed to be like “66_HW4”, 66 is the team number.

Deadline: 5/31 23:59 (for both Mon56 and Tue23)