//Create a database called university. Within this database, define a table named student with the following columns:
// id, name, surname, birthday, gpa.
//1. Write a program to retrieve and display all records from the student table (JTable)
//2. Write a program to retrieve and display all students whose names are palindromes.
//3. Write a program to retrieve and display all students who are younger than 22 years old.
//4. Write a program to sort the students by GPA
//5. Write a program that calculates and displays the average GPA of all students stored in the table.

//Main.java

```
public class Main {
    public static void main(String[] args) {
        MyWindow myWindow = new MyWindow();
        myWindow.setDefaultCloseOperation(3);
    }
}
```

//Student.java

```
import java.sql.Date;

public class Student {
    //field names will be given
    private int id;
    private String name;
    private String surname;
    private Date birthday;
    private int gpa;

    //Constructors, Setters and Getters can be generated from Code->Generate
    public Student(int id, String name, String surname, Date birthday, int gpa) {
        this.id = id;
        this.name = name;
        this.surname = surname;
        this.birthday = birthday;
        this.gpa = gpa;
    }
    public Student(){

    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public Date getBirthday() {
        return birthday;
    }

    public void setBirthday(Date birthday) {
        this.birthday = birthday;
    }

    public int getGpa() {
        return gpa;
    }

    public void setGpa(int gpa) {
        this.gpa = gpa;
    }

    //Depending on the number of fields and their types, modify this function that transforms a
    //Student object into an array of strings
    public String[] toRow(){
        String[] st = new String[5];
        st[0] = String.valueOf(getId());
        st[1] = getName();
        st[2] = getSurname();
        st[3] = String.valueOf(getBirthday());
        st[4] = String.valueOf(getGpa());

        return st;
    }
}
```

//StudentDAO.java

```
import java.sql.*;
import java.util.ArrayList;
import java.time.LocalDate;
import java.time.Period;

public class StudentDAO {
```

```
//No changes in the first 2 lines
private ArrayList<Student> list = new ArrayList<>();
public ArrayList<Student> getList(){return list;}
//Change the names of the columns to the ones given
public String[] columns = {"id", "name" , "surname", "birthday", "gpa"};
//No changes
public String[][] studentsData(ArrayList<Student> newList){
   String[][] data = new String[newList.size()][columns.length];
   for (int i =0; i<newList.size();i++)
      data[i] = newList.get(i).toRow();
   return data;
}

//Minimal changes
public void selectStudents(){
   list = new ArrayList<Student>();
   try{
      Statement statement = ConnectionDB.connectDB().createStatement();
      ResultSet resultSet = statement.executeQuery("Select * from students");
      while(resultSet.next()){
         //only modify these to match the given fields
         int id = resultSet.getInt(1);
         String name = resultSet.getString(2);
         String surname = resultSet.getString(3);
         Date birthday = resultSet.getDate(4);
         int gpa = resultSet.getInt(5);

         Student student = new Student(id, name, surname, birthday, gpa);
         //till here
         list.add(student);
      }
   } catch (SQLException e) {
      throw new RuntimeException(e);
   }
}
//The tasks are just examples, some parts of this code can be useful to have at the exam
public String[][] task2(){
   ArrayList<Student> newList = new ArrayList<>();
   if (getList().isEmpty())
      selectStudents();
   for (Student s: getList()){
      if (isPalindrome(s.getName())){
         System.out.println(s.getName() + "\n");
         newList.add(s);
      }
   }
   return studentsData(newList);
}
private boolean isPalindrome(String str){
   for (int i = 0; i < str.length()/2; i++) {
      if (str.toLowerCase().charAt(i) != str.charAt(str.length() - i-1)){
         return false;
      }
   }
   return true;
}
```

```
    public String[][] task3(){
        ArrayList<Student> newList = new ArrayList<>();
        if (getList().isEmpty())
            selectStudents();
        for (Student s: getList()){
            LocalDate birthdate = s.getBirthday().toLocalDate();
            LocalDate today = LocalDate.now();
            int age = Period.between(birthdate, today).getYears();
            if (age < 22) newList.add(s);
        }
        return studentsData(newList);
    }
    public String[][] task4() {
        ArrayList<Student> sortedList = new ArrayList<>();
        if (getList().isEmpty()) selectStudents();
        while (!list.isEmpty()) {
            int maxIndex = 0;
            for (int i = 1; i < list.size(); i++) {
                if (list.get(i).getGpa() > list.get(maxIndex).getGpa()) {
                    maxIndex = i;
                }
            }
            sortedList.add(list.get(maxIndex));
            list.remove(maxIndex);
        }
        return studentsData(sortedList);
    }
    public String task5(){
        ArrayList<Student> sortedList = new ArrayList<>();
        if (getList().isEmpty()) selectStudents();
        float avg =0;
        for (Student s: list){
            avg += s.getGpa();
        }
        avg/=list.size();

        return String.valueOf(avg);

    }
}
```

//MyWindow.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;

public class MyWindow extends JFrame implements ActionListener {
    //No changes
    private JTable j;
    private StudentDAO stdao = new StudentDAO();
    //Create as many buttons as necessary
    private JButton task2 = new JButton("Task 2");
    private JButton task3 = new JButton("Task 3");
private JButton task4 = new JButton("Task 4");
private JButton task5 = new JButton("Task 5");
private JButton refresh = new JButton("Task1");
private JLabel averageGPA = new JLabel("");
//Change the names of the columns to the ones given
private String[] columnNames = {"id", "name", "surname", "birthday", "gpa"};

public MyWindow(){
   //No changes
   setLayout(new FlowLayout());
   String [][] data = stdao.studentsData(stdao.getList());
   j = new JTable(data, columnNames);
   j.setBounds(30,40,200,300);
   JScrollPane sp = new JScrollPane(j);

   //do this for all of your buttons
   task2.addActionListener(this);
   refresh.addActionListener(this);
   task3.addActionListener(this);
   task4.addActionListener(this);
   task5.addActionListener(this);

   //Add all components you have
   add(sp);
   add(averageGPA);
   add(refresh);
   add(task2);
   add(task3);
   add(task4);
   add(task5);

   //No changes
   setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   setSize(900, 600);
   setVisible(true);
}
//No changes
private void updateTable(String[][] data){
   Container parent = j.getParent();
   parent.remove(j);
   j = new JTable(data, columnNames);
   parent.add(new JScrollPane(j));
   parent.revalidate();
   parent.repaint();
}
//Keep the structure, only modify DAO function names if necessary
@Override
public void actionPerformed(ActionEvent e) {
   if (e.getSource() == refresh){
      stdao.selectStudents();
      updateTable(stdao.studentsData(stdao.getList()));
   }
   else if (e.getSource() == task2){
      String[][] data = stdao.task2();
      updateTable(data);
   }
```

```
        else if(e.getSource() == task3){
            String[][] data = stdao.task3();
            updateTable(data);
        }
        else if (e.getSource() == task4){
            String [][] data = stdao.task4();
            updateTable(data);
        }
        else if (e.getSource() == task5){
            averageGPA.setText(stdao.task5());
        }
    }
}
```

//ConnectionDB.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class ConnectionDB {
    public static Connection connectDB(){
        Connection con = null;
        try{
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/university", "root",
"4852");
            System.out.println("Connected");
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
        return con;
    }
}
```

How to Connect DB

1) Open MySQL Command Line Client
2) input password
3) write the commands (use shift+enter to go to a new line)

```
CREATE DATABASE university;
USE university;
CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    surname VARCHAR(50),
    birthday DATE,
    gpa INT
    );
INSERT INTO students (name, surname, birthday, gpa) VALUES
  ('Liam', 'Taylor', '1998-11-30', 50),
  ('Sophia', 'Davis', '2002-03-14', 29),
  ('Ethan', 'Miller', '2000-06-22', 98),
  ('Olivia', 'Anderson', '2001-09-10', 45),
  ('Isaac', 'Martin', '1999-04-05', 76);
```

4) check the database to be sure

```
   SELECT * FROM students;
```

How to add MySQL Connector

In IntelliJ, go to File -> Project Structure -> Libraries, select the “+” -> Java,
Find and select the **mysql-connector-j-8.3.0.jar** file -> OK -> Apply -> OK