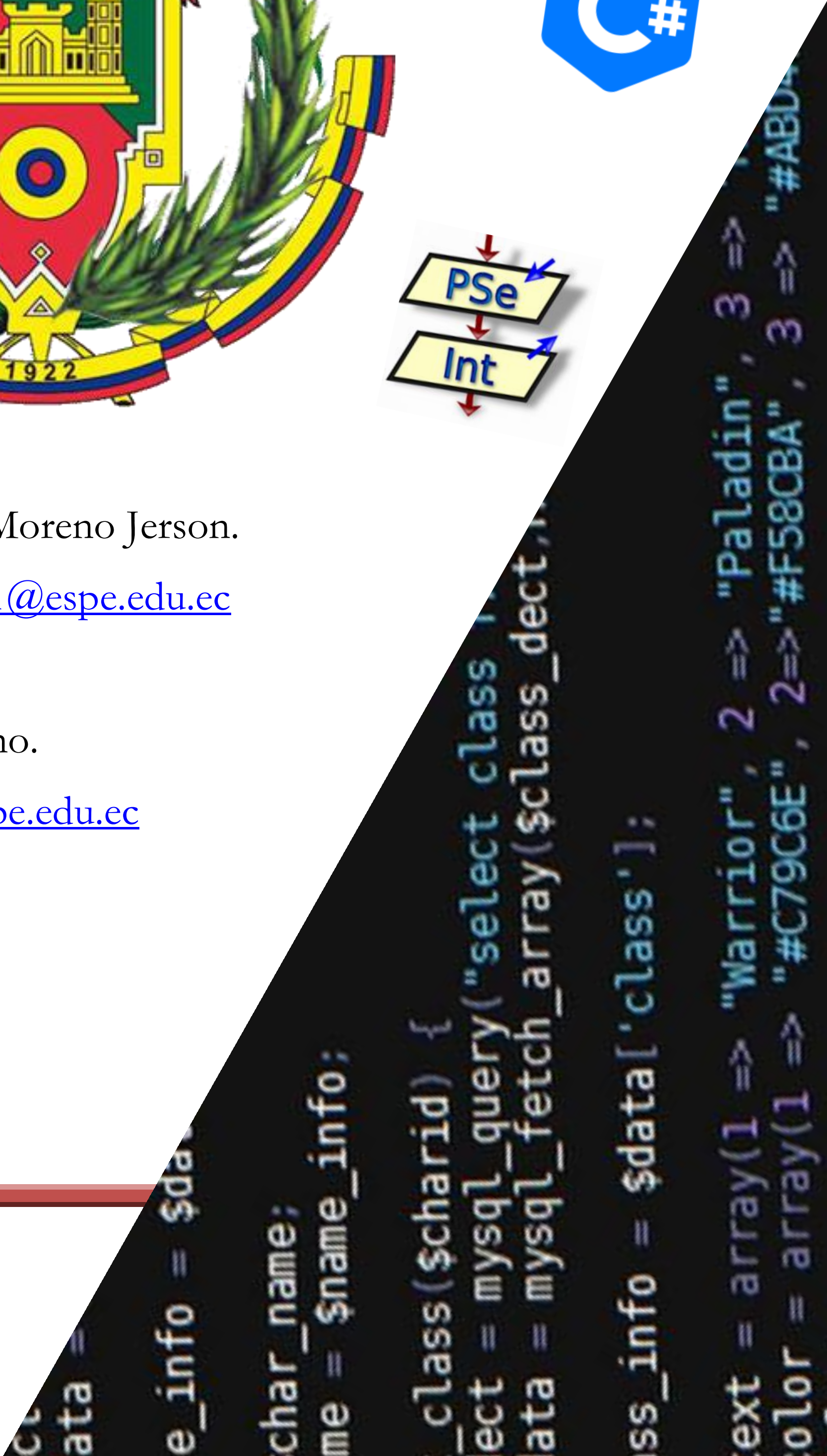
ESPE UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Ciencias de la Computación

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

# OBJECT ORIENTED PROGRAMMING



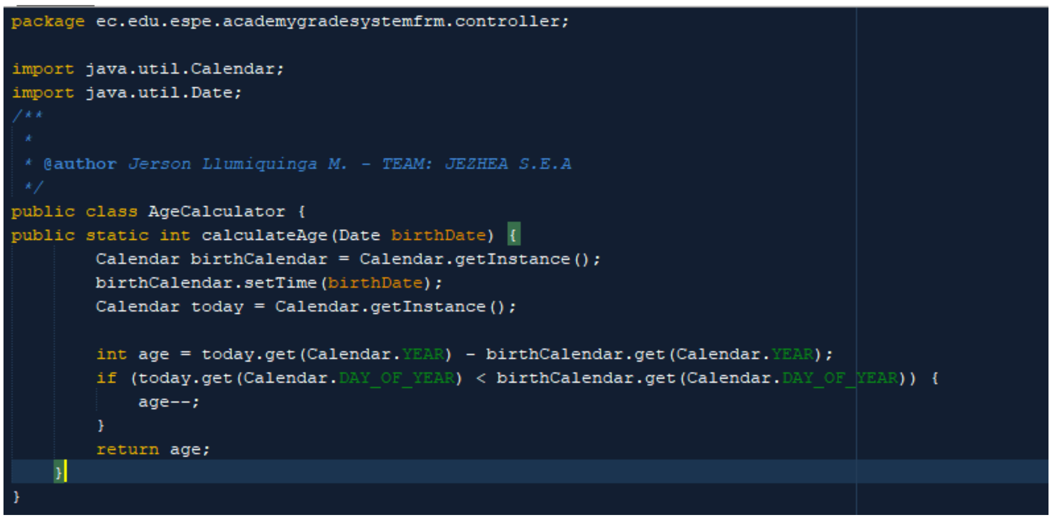
| | |
|---|---|
| **Student:** | Llumiquinga Moreno Jerson. |
| E-mail: | jsllumiquinga1@espe.edu.ec |
| #Cell: | 0980460057 |
| **Professor:** | Edison Lascano. |
| E-mail: | jelascano@espe.edu.ec |
| #Cell: | 0961195050 |
| **NRC:** | 14539 |

**MAY24 - SEP24**

## Handling null in birthDate:

**Problem:** If the calculateAge method receives a birthDate that is null, a NullPointerException will be thrown when trying to set the time to birthCalendar.

**Solution:** Add a check at the beginning of the method to handle this case.

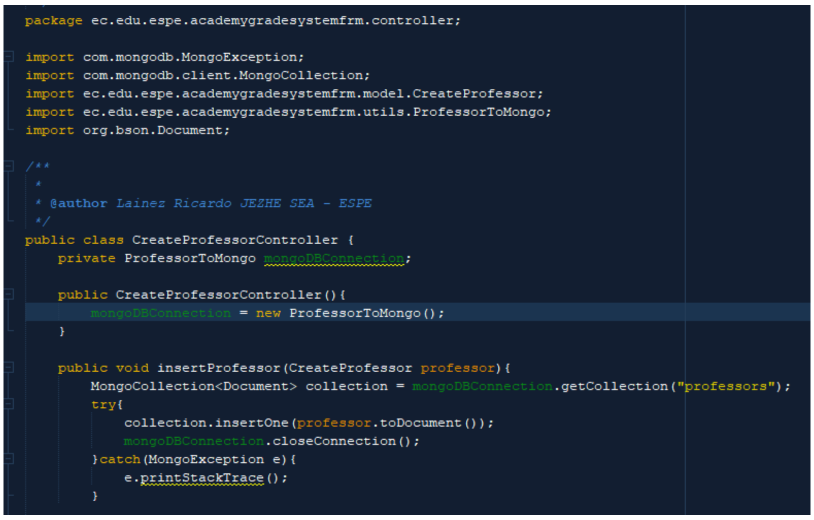


```
package ec.edu.espe.academygradesystemfrm.controller;

import java.util.Calendar;
import java.util.Date;
/**
 *
 * @author Jerson Llumiquinga M. - TEAM: JEZHEA S.E.A
 */
public class AgeCalculator {
public static int calculateAge(Date birthDate) {
        Calendar birthCalendar = Calendar.getInstance();
        birthCalendar.setTime(birthDate);
        Calendar today = Calendar.getInstance();

        int age = today.get(Calendar.YEAR) - birthCalendar.get(Calendar.YEAR);
        if (today.get(Calendar.DAY_OF_YEAR) < birthCalendar.get(Calendar.DAY_OF_YEAR)) {
            age--;
        }
        return age;
    }
}
```

## Close the connection after each insert:

**Problem**: In the insertProfessor method, the connection to MongoDB is closed right after inserting a teacher. If insertProfessor is called multiple times, it will open and close the connection on each call, which could be inefficient.

**Solution:** Keep the connection open while the application is running and close it only when it is no longer needed, such as when ending the application.

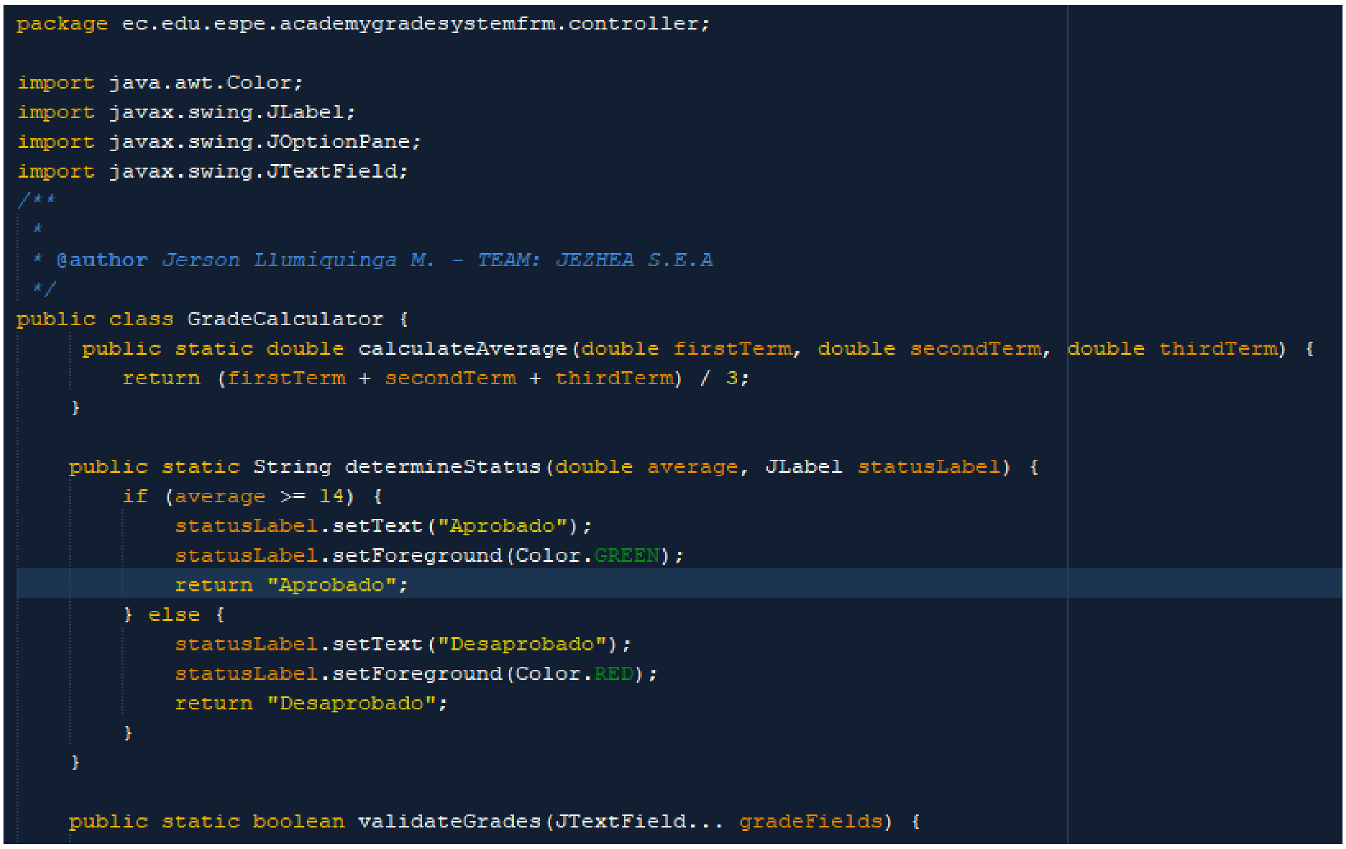


```
package ec.edu.espe.academygradesystemfrm.controller;

import com.mongodb.MongoException;
import com.mongodb.client.MongoCollection;
import ec.edu.espe.academygradesystemfrm.model.CreateProfessor;
import ec.edu.espe.academygradesystemfrm.utils.ProfessorToMongo;
import org.bson.Document;

/**
 *
 * @author Lainez Ricardo JEZHE SEA - ESPE
 */
public class CreateProfessorController {
    private ProfessorToMongo mongoDBConnection;

    public CreateProfessorController(){
        mongoDBConnection = new ProfessorToMongo();
    }

    public void insertProfessor(CreateProfessor professor){
        MongoCollection<Document> collection = mongoDBConnection.getCollection("professors");
        try{
            collection.insertOne(professor.toDocument());
            mongoDBConnection.closeConnection();
        }catch(MongoException e){
            e.printStackTrace();
        }
```

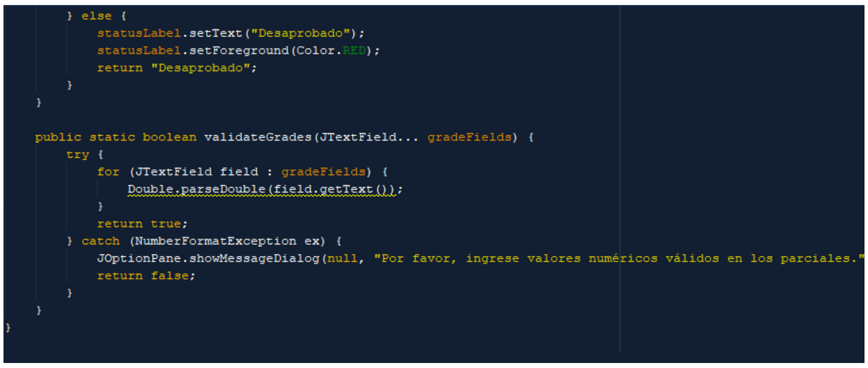
**Close the connection after each insert:**

**Problem:** In the insertProfessor method, the connection to MongoDB is closed right after inserting a teacher. If insertProfessor is called multiple times, it will open and close the connection on each call, which could be inefficient.

**Solution:** Keep the connection open while the application is running and close it only when it is no longer needed, such as when ending the application.



```java
package ec.edu.espe.academygradesystemfrm.controller;

import java.awt.Color;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
/**
 *
 * @author Jerson Llumiquinga M. - TEAM: JEZHEA S.E.A
 */
public class GradeCalculator {
     public static double calculateAverage(double firstTerm, double secondTerm, double thirdTerm) {
        return (firstTerm + secondTerm + thirdTerm) / 3;
    }

    public static String determineStatus(double average, JLabel statusLabel) {
        if (average >= 14) {
            statusLabel.setText("Aprobado");
            statusLabel.setForeground(Color.GREEN);
            return "Aprobado";
        } else {
            statusLabel.setText("Desaprobado");
            statusLabel.setForeground(Color.RED);
            return "Desaprobado";
        }
    }

    public static boolean validateGrades(JTextField... gradeFields) {
```



```java
        } else {
            statusLabel.setText("Desaprobado");
            statusLabel.setForeground(Color.RED);
            return "Desaprobado";
        }
    }

    public static boolean validateGrades(JTextField... gradeFields) {
        try {
            for (JTextField field : gradeFields) {
                Double.parseDouble(field.getText());
            }
            return true;
        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(null, "Por favor, ingrese valores numéricos válidos en los parciales."
            return false;
        }
    }
}
```