

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE  
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

# OBJECT ORIENTED PROGRAMMING

# JS



**Student:**  Llumiquinga Moreno Jerson.

E-mail: [isllumiquinga1@espe.edu.ec](mailto:isllumiquinga1@espe.edu.ec)

#Cell: 0980460057

**Professor:** Edison Lascano.

E-mail: [jelascano@espe.edu.ec](mailto:jelascano@espe.edu.ec)

#Cell: 0961195050

**NRC:** 14539


**MAY24 - SEP24**

Moreno Jerson.

[jesper.moreno@espe.edu.ec](mailto:jesper.moreno@espe.edu.ec)

no.

[jesper.moreno@espe.edu.ec](mailto:jesper.moreno@espe.edu.ec)



```
data = $data;
e_info = $data['e_info'];
char_name;
me = $name_info;

class($charid) {
    _select = mysql_query("select class_id, char_name,
    _data = mysql_fetch_array($class_data);

    ss_info = $data['class'];

    ext = array(1 => "Warrior", 2 => "Paladin", 3 => "Priest", 4 => "Rogue", 5 => "Mage", 6 => "Druid", 7 => "Hunter", 8 => "Shaman", 9 => "Monk", 10 => "Death Knight", 11 => "Warlock", 12 => "Demon Hunter", 13 => "Bard", 14 => "Cleric", 15 => "Druid", 16 => "Mage", 17 => "Monk", 18 => "Paladin", 19 => "Priest", 20 => "Rogue", 21 => "Shaman", 22 => "Warlock", 23 => "Warrior", 24 => "Death Knight", 25 => "Demon Hunter", 26 => "Bard", 27 => "Cleric", 28 => "Druid", 29 => "Mage", 30 => "Monk", 31 => "Paladin", 32 => "Priest", 33 => "Rogue", 34 => "Shaman", 35 => "Warlock", 36 => "Warrior", 37 => "Death Knight", 38 => "Demon Hunter", 39 => "Bard", 40 => "Cleric", 41 => "Druid", 42 => "Mage", 43 => "Monk", 44 => "Paladin", 45 => "Priest", 46 => "Rogue", 47 => "Shaman", 48 => "Warlock", 49 => "Warrior", 50 => "Death Knight", 51 => "Demon Hunter", 52 => "Bard", 53 => "Cleric", 54 => "Druid", 55 => "Mage", 56 => "Monk", 57 => "Paladin", 58 => "Priest", 59 => "Rogue", 60 => "Shaman", 61 => "Warlock", 62 => "Warrior", 63 => "Death Knight", 64 => "Demon Hunter", 65 => "Bard", 66 => "Cleric", 67 => "Druid", 68 => "Mage", 69 => "Monk", 70 => "Paladin", 71 => "Priest", 72 => "Rogue", 73 => "Shaman", 74 => "Warlock", 75 => "Warrior", 76 => "Death Knight", 77 => "Demon Hunter", 78 => "Bard", 79 => "Cleric", 80 => "Druid", 81 => "Mage", 82 => "Monk", 83 => "Paladin", 84 => "Priest", 85 => "Rogue", 86 => "Shaman", 87 => "Warlock", 88 => "Warrior", 89 => "Death Knight", 90 => "Demon Hunter", 91 => "Bard", 92 => "Cleric", 93 => "Druid", 94 => "Mage", 95 => "Monk", 96 => "Paladin", 97 => "Priest", 98 => "Rogue", 99 => "Shaman", 100 => "Warlock", 101 => "Warrior", 102 => "Death Knight", 103 => "Demon Hunter", 104 => "Bard", 105 => "Cleric", 106 => "Druid", 107 => "Mage", 108 => "Monk", 109 => "Paladin", 110 => "Priest", 111 => "Rogue", 112 => "Shaman", 113 => "Warlock", 114 => "Warrior", 115 => "Death Knight", 116 => "Demon Hunter", 117 => "Bard", 118 => "Cleric", 119 => "Druid", 120 => "Mage", 121 => "Monk", 122 => "Paladin", 123 => "Priest", 124 => "Rogue", 125 => "Shaman", 126 => "Warlock", 127 => "Warrior", 128 => "Death Knight", 129 => "Demon Hunter", 130 => "Bard", 131 => "Cleric", 132 => "Druid", 133 => "Mage", 134 => "Monk", 135 => "Paladin", 136 => "Priest", 137 => "Rogue", 138 => "Shaman", 139 => "Warlock", 140 => "Warrior", 141 => "Death Knight", 142 => "Demon Hunter", 143 => "Bard", 144 => "Cleric", 145 => "Druid", 146 => "Mage", 147 => "Monk", 148 => "Paladin", 149 => "Priest", 150 => "Rogue", 151 => "Shaman", 152 => "Warlock", 153 => "Warrior", 154 => "Death Knight", 155 => "Demon Hunter", 156 => "Bard", 157 => "Cleric", 158 => "Druid", 159 => "Mage", 160 => "Monk", 161 => "Paladin", 162 => "Priest", 163 => "Rogue", 164 => "Shaman", 165 => "Warlock", 166 => "Warrior", 167 => "Death Knight", 168 => "Demon Hunter", 169 => "Bard", 170 => "Cleric", 171 => "Druid", 172 => "Mage", 173 => "Monk", 174 => "Paladin", 175 => "Priest", 176 => "Rogue", 177 => "Shaman", 178 => "Warlock", 179 => "Warrior", 180 => "Death Knight", 181 => "Demon Hunter", 182 => "Bard", 183 => "Cleric", 184 => "Druid", 185 => "Mage", 186 => "Monk", 187 => "Paladin", 188 => "Priest", 189 => "Rogue", 190 => "Shaman", 191 => "Warlock", 192 => "Warrior", 193 => "Death Knight", 194 => "Demon Hunter", 195 => "Bard", 196 => "Cleric", 197 => "Druid", 198 => "Mage", 199 => "Monk", 200 => "Paladin", 201 => "Priest", 202 => "Rogue", 203 => "Shaman", 204 => "Warlock", 205 => "Warrior", 206 => "Death Knight", 207 => "Demon Hunter", 208 => "Bard", 209 => "Cleric", 210 => "Druid", 211 => "Mage", 212 => "Monk", 213 => "Paladin", 214 => "Priest", 215 => "Rogue", 216 => "Shaman", 217 => "Warlock", 218 => "Warrior", 219 => "Death Knight", 220 => "Demon Hunter", 221 => "Bard", 222 => "Cleric", 223 => "Druid", 224 => "Mage", 225 => "Monk", 226 => "Paladin", 227 => "Priest", 228 => "Rogue", 229 => "Shaman", 230 => "Warlock", 231 => "Warrior", 232 => "Death Knight", 233 => "Demon Hunter", 234 => "Bard", 235 => "Cleric", 236 => "Druid", 237 => "Mage", 238 => "Monk", 239 => "Paladin", 240 => "Priest", 241 => "Rogue", 242 => "Shaman", 243 => "Warlock", 244 => "Warrior", 245 => "Death Knight", 246 => "Demon Hunter", 247 => "Bard", 248 => "Cleric", 249 => "Druid", 250 => "Mage", 251 => "Monk", 252 => "Paladin", 253 => "Priest", 254 => "Rogue", 255 => "Shaman", 256 => "Warlock", 257 => "Warrior", 258 => "Death Knight", 259 => "Demon Hunter", 260 => "Bard", 261 => "Cleric", 262 => "Druid", 263 => "Mage", 264 => "Monk", 265 => "Paladin", 266 => "Priest", 267 => "Rogue", 268 => "Shaman", 269 => "Warlock", 270 => "Warrior", 271 => "Death Knight", 272 => "Demon Hunter", 273 => "Bard", 274 => "Cleric", 275 => "Druid", 276 => "Mage", 277 => "Monk", 278 => "Paladin", 279 => "Priest", 280 => "Rogue", 281 => "Shaman", 282 => "Warlock", 283 => "Warrior", 284 => "Death Knight", 285 => "Demon Hunter", 286 => "Bard", 287 => "Cleric", 288 => "Druid", 289 => "Mage", 290 => "Monk", 291 => "Paladin", 292 => "Priest", 293 => "Rogue", 294 => "Shaman", 295 => "Warlock", 296 => "Warrior", 297 => "Death Knight", 298 => "Demon Hunter", 299 => "Bard", 300 => "Cleric", 301 => "Druid", 302 => "Mage", 303 => "Monk", 304 => "Paladin", 305 => "Priest", 306 => "Rogue", 307 => "Shaman", 308 => "Warlock", 309 => "Warrior", 310 => "Death Knight", 311 => "Demon Hunter", 312 => "Bard", 313 => "Cleric", 314 => "Druid", 315 => "Mage", 316 => "Monk", 317 => "Paladin", 318 => "Priest", 319 => "Rogue", 320 => "Shaman", 321 => "Warlock", 322 => "Warrior", 323 => "Death Knight", 324 => "Demon Hunter", 325 => "Bard", 326 => "Cleric", 327 => "Druid", 328 => "Mage", 329 => "Monk", 330 => "Paladin", 331 => "Priest", 332 => "Rogue", 333 => "Shaman", 334 => "Warlock", 335 => "Warrior", 336 => "Death Knight", 337 => "Demon Hunter", 338 => "Bard", 339 => "Cleric", 340 => "Druid", 341 => "Mage", 342 => "Monk", 343 => "Paladin", 344 => "Priest", 345 => "Rogue", 346 => "Shaman", 347 => "Warlock", 348 => "Warrior", 349 => "Death Knight", 350 => "Demon Hunter", 351 => "Bard", 352 => "Cleric", 353 => "Druid", 354 => "Mage", 355 => "Monk", 356 => "Paladin", 357 => "Priest", 358 => "Rogue", 359 => "Shaman", 360 => "War
```

## 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: JEEHEA 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 JEEHE 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.

```
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) {
```

```
        } 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;
        }
    }
}
```