



**Module Code & Module Title CS4001NP Programming**

**Assessment weightage & Type 30% Individual Coursework-1 Year and Semester** **2023 Autumn**

**Student Name: Birat Adhikari** **Submitted To: Sushil Poudel Group-C5**

**College ID: NP04CP4A230104 London Met ID: 23048781 Assessment Due Date: 2024-10-10**

**Table of Contents**

[1. Introduction 4](#_Toc166233070)

[2. Class Diagram 5](#_Toc166233071)

[3. Pseudocode 9](#_Toc166233072)

[3.1 Pseudocode of Teacher 10](#_Toc166233073)

[4. Short description of each method 15](#_Toc166233074)

[5. Testing 17](#_Toc166233075)

[5.1Test1 17](#_Toc166233076)

[5.2Test2 18](#_Toc166233077)

[5.3Test3 23](#_Toc166233078)

[6. Error correction and detection 26](#_Toc166233079)

[6.1 Syntax Error 26](#_Toc166233080)

[6.2 Logical Error 28](#_Toc166233081)

[6.3 Runtime Error 29](#_Toc166233082)

[7. Conclusion 30](#_Toc166233083)

[8.References 30](#_Toc166233084)

[9.Appendix 31](#_Toc166233085)

**Table of Fig**

[Figure 1:Class Diagram 9](#_Toc166233094)

[Figure 2::running program from command prompt 18](#_Toc166233095)

[Figure 3:adding data in lecturer 19](#_Toc166233096)

[Figure 4:Display Dialog box 19](#_Toc166233097)

[Figure 5: Removing Tutor 20](#_Toc166233098)

[Figure 6:adding lecturer 21](#_Toc166233099)

[Figure 7:displaying message after adding lecturer 21](#_Toc166233100)

[Figure 8:grading assignment 22](#_Toc166233101)

[Figure 9:setting salary 22](#_Toc166233102)

[Figure 10:displaying message after setting salary 23](#_Toc166233103)

[Figure 11:adding string value in interger 24](#_Toc166233104)

[Figure 12:displaying after entering incorrect value 24](#_Toc166233105)

[Figure 13:adding string value in integer for lecturer 25](#_Toc166233106)

[Figure 14:displaying message after wrong input 26](#_Toc166233107)

# 

# 1. Introduction

As we all know computers does not understand natural language as human does. So, in order to communicate with computers and perform some task we need some medium to communicate with machines which is defined as Programming languages. Java is one of the popular programming languages. Although, the code written in high-level language is not understood by computers the high-level code is later converted to byte code or let’s say machine code by computer it-self.

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Patrick Naughton, Mike Sheridan, and Jame Gosling, known as the Green team, initiated the development of Java in the year 1991. In 1996, the first public implementation was released as Java 1.0. The new versions have multiple different profile that have built for various platforms (Scott, 2018).

BlueJ is an integrated development environment (IDE) for the Java programming language. BlueJ is developed mainly for educational purposes but suitable for small scale software development. It is beginner friendly IDE that does not have more developer features which eventually enhanced the knowledge of learner’s in effective way. The main or first interface of BlueJ shows class structure of an application which in under development phase.

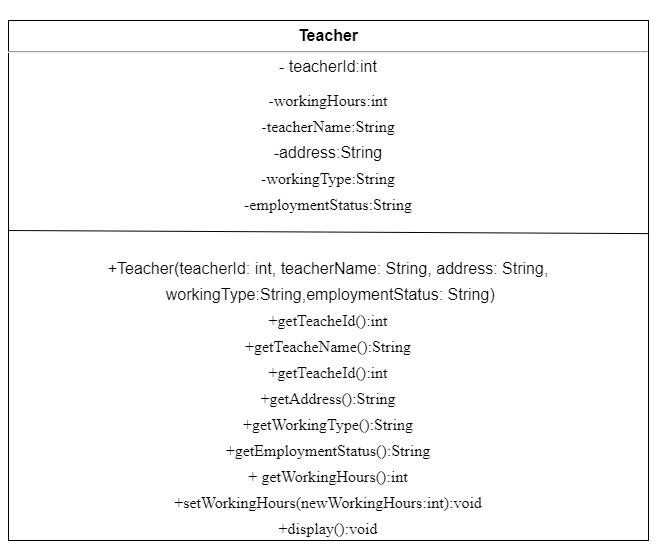
Object-oriented concepts are represented visually (Rouse, 2013).

This course carries 30% of the total percentage of our Module. This project helps me to apply theoretical knowledge to real world situations. This project improves my collaboration and teamwork skills. This project promotes our skill, continuous learning, and a deeper understanding of course content.

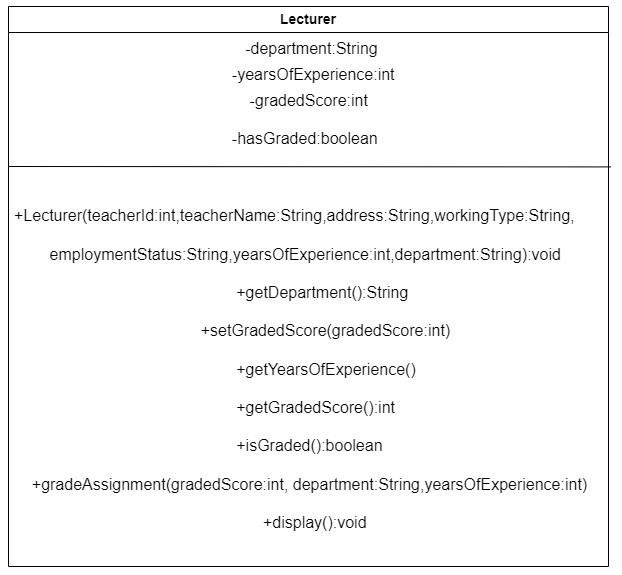
During this with balancing multiple coursework assignments.

# 2. Class Diagram

Class diagrams are static structure diagrams that visualize and model object-oriented systems or, in simple words class diagram is simply a visual representation of classes you’ve defined. Class diagrams are created using Unified Modelling Language (UML). UML is a common, standardized graphical language that lets you visualize software systems. The biggest benefit of class diagrams is that they let you visualize systems. Complex ideas in visual form is important because [our brain can process images faster](https://www.emailaudience.com/research-picture-worth-1000-words-marketing/) than other types of information (Rouse, 2013).



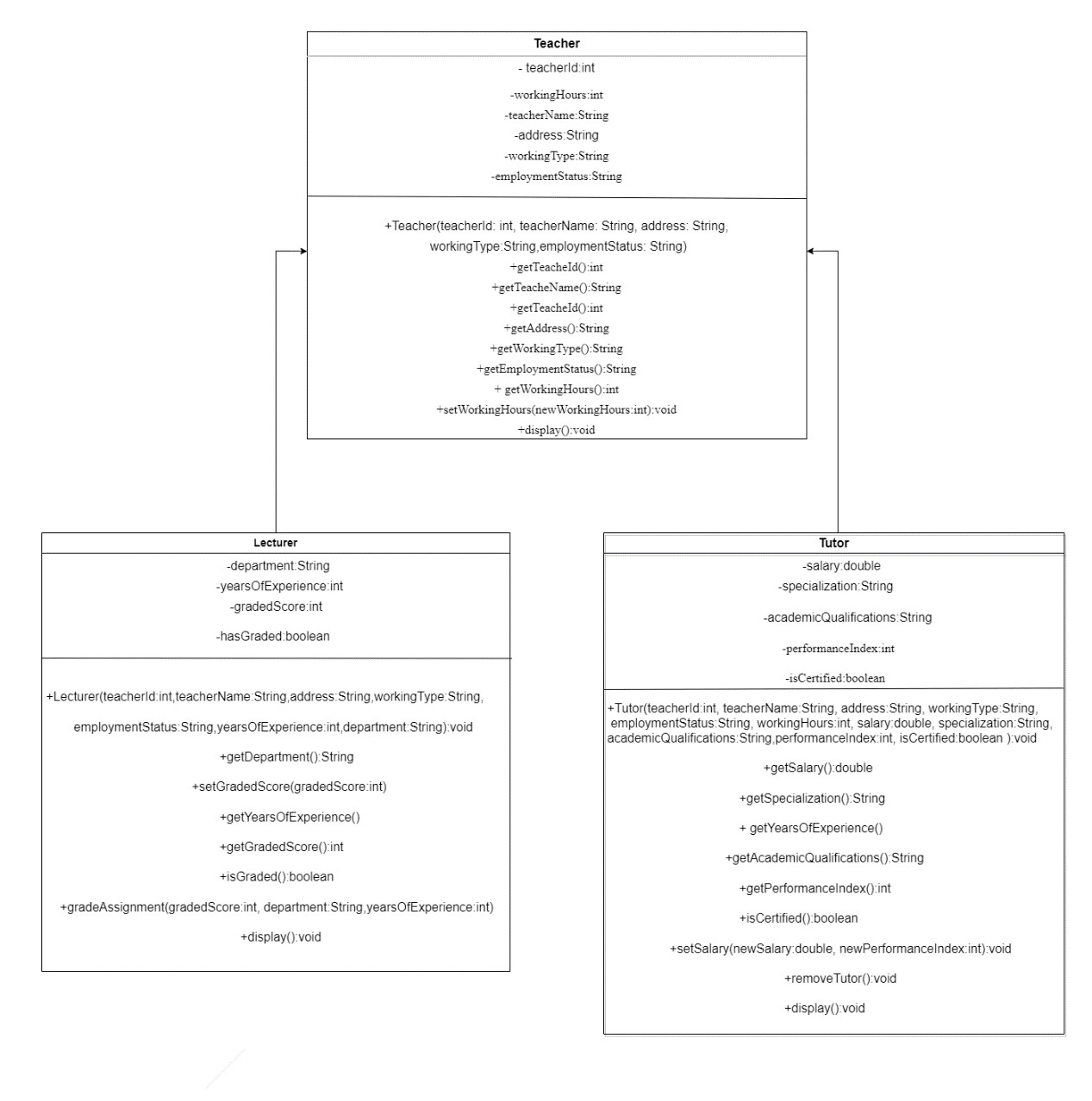
*Figure 1:Class Diagram of Teacher*



*Figure 2:Class Diagram of Lecturer*



*Figure 3:Class Diagram of Tutor*



*Figure 4:Combined Class Diagram of 3 classes*

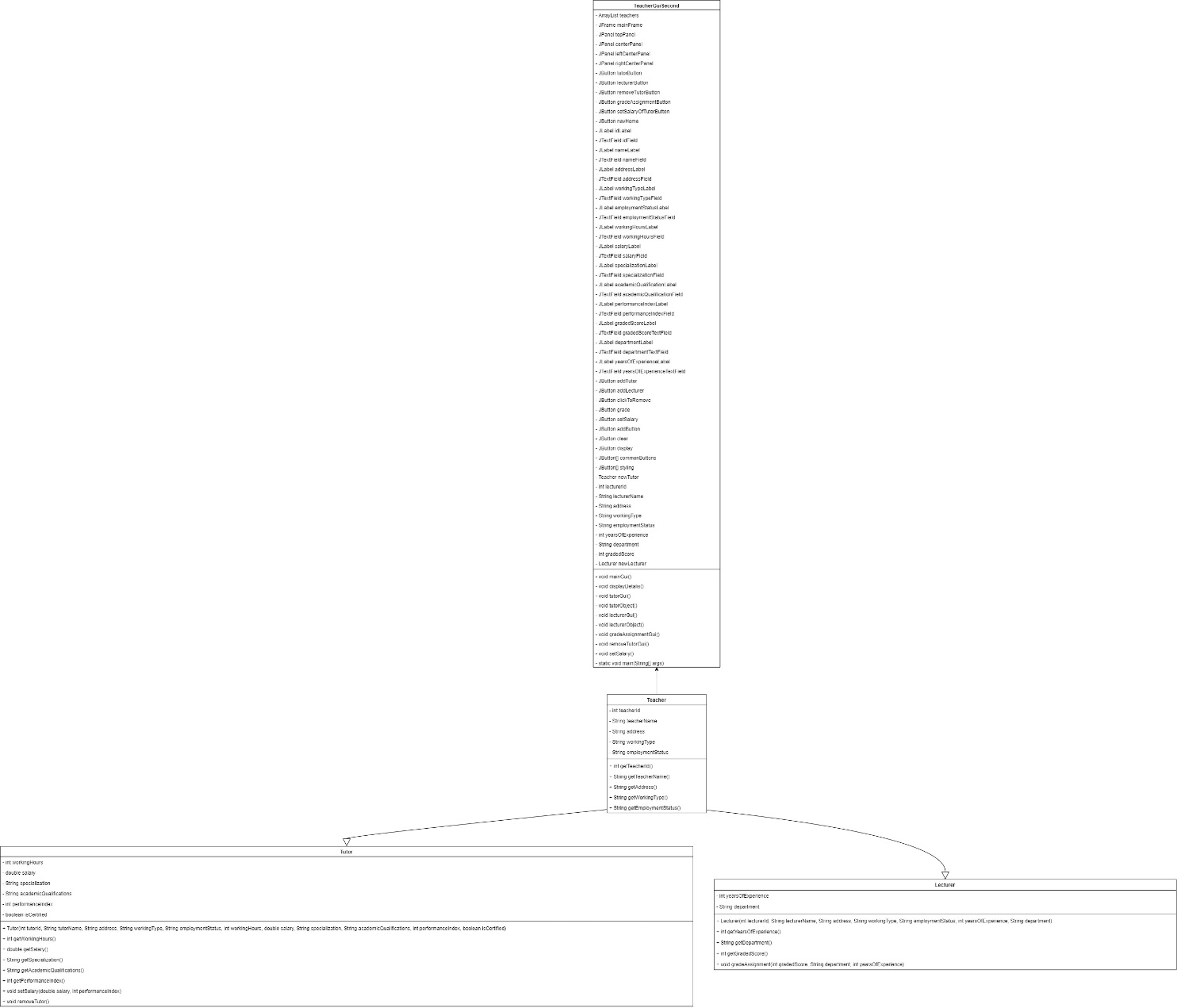


Figure 1:Class Diagram

# 3. Pseudocode

Pseudocode is an informal way of programming description that does not require any programming language syntax or other considerations. It is used for creating an outline or a rough draft of a program. It is not a programming language so it cannot be compiled into an executable program. It uses simple programming language to write code for programs before development. Pseudocode is understood by all programmers easily. Pseudocode is written by programmers before the development process of software or program (Ubah, 2021).

## 3.1 Pseudocode of Teacher

BEGIN

DECLARE ArrayList teachers

CREATE JFrame mainFrame

CREATE JPanel topPanel

CREATE JPanel centerPanel

CREATE JPanel leftCenterPanel

CREATE JPanel rightCenterPanel

CREATE JButtons tutorButton, lecturerButton, removeTutorButton, gradeAssignmentButton, setSalaryOfTutorButton, navHome

CREATE JLabels idLabel, nameLabel, addressLabel, workingTypeLabel, employmentStatusLabel, workingHoursLabel, salaryLabel, specializationLabel, academicQualificationLabel, performanceIndexLabel, gradedScoreLabel, departmentLabel, yearsOfExperienceLabel

CREATE JTextFields idField, nameField, addressField, workingTypeField, employmentStatusField, workingHoursField, salaryField, specializationField, academicQualificationField, performanceIndexField, gradedScoreTextField, departmentTextField, yearsOfExperienceTextField

CREATE JButtons addButton, clear, display

FUNCTION TeacherGuiSecond()

CALL mainGui()

FUNCTION mainGui()

FOR EACH JButton button IN {tutorButton, gradeAssignmentButton, setSalaryOfTutorButton, removeTutorButton, lecturerButton, display}

SET button font to Arial, plain, size 15

SET button foreground color to white

SET button background color to RGB(40, 40, 43)

SET button border to raised bevel border

SET topPanel background color to RGB(40, 40, 43)

CREATE JLabel topLabel with text "Informatics College Pokhara", font TimesRoman, bold, size 20, foreground color white

SET topLabel icon to logo

ADD topLabel to topPanel

SET leftCenterPanel background color to white

CREATE JLabel leftLabel with text "Welcome Home !!", font TimesRoman, bold, size 30

ADD leftLabel to leftCenterPanel

SET rightCenterPanel layout to 3x2 grid

FOR EACH JButton button IN {tutorButton, gradeAssignmentButton, setSalaryOfTutorButton, removeTutorButton, lecturerButton, display}

ADD button to rightCenterPanel

ADD ActionListener to tutorButton, lecturerButton, removeTutorButton, gradeAssignmentButton, setSalaryOfTutorButton, display

ADD topPanel and centerPanel to mainFrame

SET mainFrame attributes: title to "CourseWork", size to (600, 500), resizable to false, visible to true, locationRelativeTo null, defaultCloseOperation to EXIT\_ON\_CLOSE

FUNCTION displayDetalis()

CREATE JFrame displayFrame with title "Teacher Details"

CREATE DefaultTableModel model with column names {"Teacher Id", "Teacher Name", "Address", "Working Type", "Employment Status", "Working Hours", "Salary", "Specialization", "Academic", "Performance Index"}

FOR EACH Teacher teacher IN teachers

IF teacher IS Tutor

CREATE Object[] rowData with values from tutor

ELSE IF teacher IS Lecturer

CREATE Object[] rowData with values from lecturer

ADD rowData to model

CREATE JTable table with model

CREATE JScrollPane scrollPane with table

ADD scrollPane to displayFrame

SET displayFrame attributes: defaultCloseOperation to DISPOSE\_ON\_CLOSE, pack, locationRelativeTo null, visible to true

FUNCTION tutorGui()

CREATE JFrame tutorFrame with title "Add Tutor"

CREATE JPanel tutorPanel

ADD labels and text fields to tutorPanel

ADD common buttons and ActionListener to home and addButton

SET tutorFrame attributes: iconImage to logo, size to (400, 500), resizable to false, visible to true, locationRelativeTo null

FUNCTION tutorObject()

ADD ActionListener to addButton

TRY

PARSE input values

IF tutorId exists in teachers

DISPLAY "This teacherId is already added, try a new teacherId"

ELSE

CREATE new Tutor object with input values

ADD new Tutor to teachers

DISPLAY "Teacher added successfully"

CATCH NumberFormatException

DISPLAY "Enter Correct Values"

FUNCTION lecturerGui()

CREATE JFrame lecturerFrame with title "Add Lecturer"

CREATE JPanel lecturerPanel

ADD labels and text fields to lecturerPanel

ADD clear and addLecturer buttons with ActionListener to addLecturer

SET lecturerFrame attributes: iconImage to logo, size to (400, 500), resizable to false, visible to true, locationRelativeTo null

FUNCTION lecturerObject()

TRY

PARSE input values

IF lecturerId exists in teachers

DISPLAY "This teacherId is already added, try a new teacherId"

ELSE

CREATE new Lecturer object with input values

ADD new Lecturer to teachers

DISPLAY "Teacher added successfully"

CATCH NumberFormatException

DISPLAY "Enter Correct Values"

FUNCTION gradeAssignmentGui()

CREATE JFrame gradeAssignmentFrame with title "Grade Assignment"

CREATE JPanel gradeAssignmentPanel

ADD labels and text fields to gradeAssignmentPanel

ADD grade and clear buttons with ActionListener to grade

SET gradeAssignmentFrame attributes: iconImage to logo, size to (400, 500), resizable to false, visible to true, locationRelativeTo null

FUNCTION removeTutorGui()

CREATE JFrame removeFrame with title "Remove Tutor"

CREATE JPanel removePanel

ADD labels, text fields, and buttons to removePanel

ADD ActionListener to clickToRemove

SET removeFrame attributes: iconImage to logo, size to (400, 500), resizable to false, visible to true, locationRelativeTo null

FUNCTION setSalary()

CREATE JFrame setSalaryFrame with title "Set Salary"

CREATE JPanel setSalaryPanel

ADD labels, text fields, and buttons to setSalaryPanel

ADD ActionListener to setSalary

SET setSalaryFrame attributes: iconImage to logo, size to (400, 500), resizable to false, visible to true, locationRelativeTo null

CALL TeacherGuiSecond()

END

# 4. Short description of each method

Add Button:

The "Add" button facilitates the creation of a new Tutor object within the system. When clicked, it triggers the tutorObject() method, which attempts to gather input data from the text fields representing various attributes of a Tutor, such as ID, name, address, etc. If the entered ID is unique, indicating that the Tutor is not already present in the system, a new Tutor object is created and added to the list of teachers. If the ID already exists, indicating a duplicate entry, an error message is displayed, prompting the user to enter a unique ID.

Clear Button:

The "Clear" button provides users with a convenient way to reset all text fields in the graphical user interface (GUI). When clicked, it triggers an ActionListener that clears all text fields, allowing users to input fresh data without the need to manually delete existing content. This feature enhances user experience by simplifying data entry processes.

Navigation to Home Button:

The "Home" button serves as a navigation tool, allowing users to return to the main home screen of the application. Depending on the context, this button is associated with various ActionListener instances that either close the current frame or navigate back to the primary interface, providing users with a consistent and intuitive navigation experience throughout the application.

Add Lecturer Button:

The "Add Lecturer" button facilitates the addition of a new Lecturer object to the system. Upon clicking, it triggers the lecturerObject() method, which collects input data from the text fields representing attributes such as ID, name, address, etc., required for creating a Lecturer. Similar to the "Add" button for Tutors, this button ensures data integrity by checking for duplicate IDs before adding the new Lecturer to the list of teachers.

Click to Remove Button:

The "Click to Remove" button provides functionality to remove a Tutor from the system. When clicked, it initiates the removeTutorGui() method, prompting users to enter the ID of the Tutor they wish to remove. If the entered ID corresponds to an existing Tutor, it is removed from the list of teachers. However, if the ID is invalid or the Tutor does not exist, an appropriate error message is displayed to notify the user.

Grade Assignment Button:

The "Grade Assignment" button enables the grading of assignments for Lecturers within the system. Clicking this button initiates the gradeAssignmentGui() method, presenting users with a form to input relevant details such as the Lecturer's ID, graded score, department, and years of experience. Based on this input, the system grades the assignment, providing feedback to the user and updating relevant data accordingly.

Set Salary Button:

The "Set Salary" button allows administrators to adjust the salary of Tutors within the system. Upon clicking, it triggers the setSalary() method, which prompts users to input the Tutor's ID, new salary, and performance index. After validating the input, the system updates the Tutor's salary accordingly, ensuring that performance considerations are taken into account during the salary adjustment process.

Display Button:

The "Display" button facilitates the visualization of detailed information about all teachers (both Tutors and Lecturers) within the system. When clicked, it invokes the displayDetails() method, which generates a separate frame containing a table with comprehensive details of each teacher, including their ID, name, address, working type, employment status, and additional attributes specific to their role (e.g., working hours, specialization, academic qualifications, etc.). This feature provides administrators with a convenient overview of the workforce, aiding in management and decision-making processes.

# 5. Testing

# 5.1Test1

|  |  |
| --- | --- |
| Objective | Compile the program and run the program using command prompt |
| Action | 1. Open the program in command prompt 2. Run the program |
| Expected outcome | Program should run from command prompt |
| Actual Output | Program is runs from command prompt |
| Conclusion | Test was successfully completed |

*Table 1:Description of Test 1*

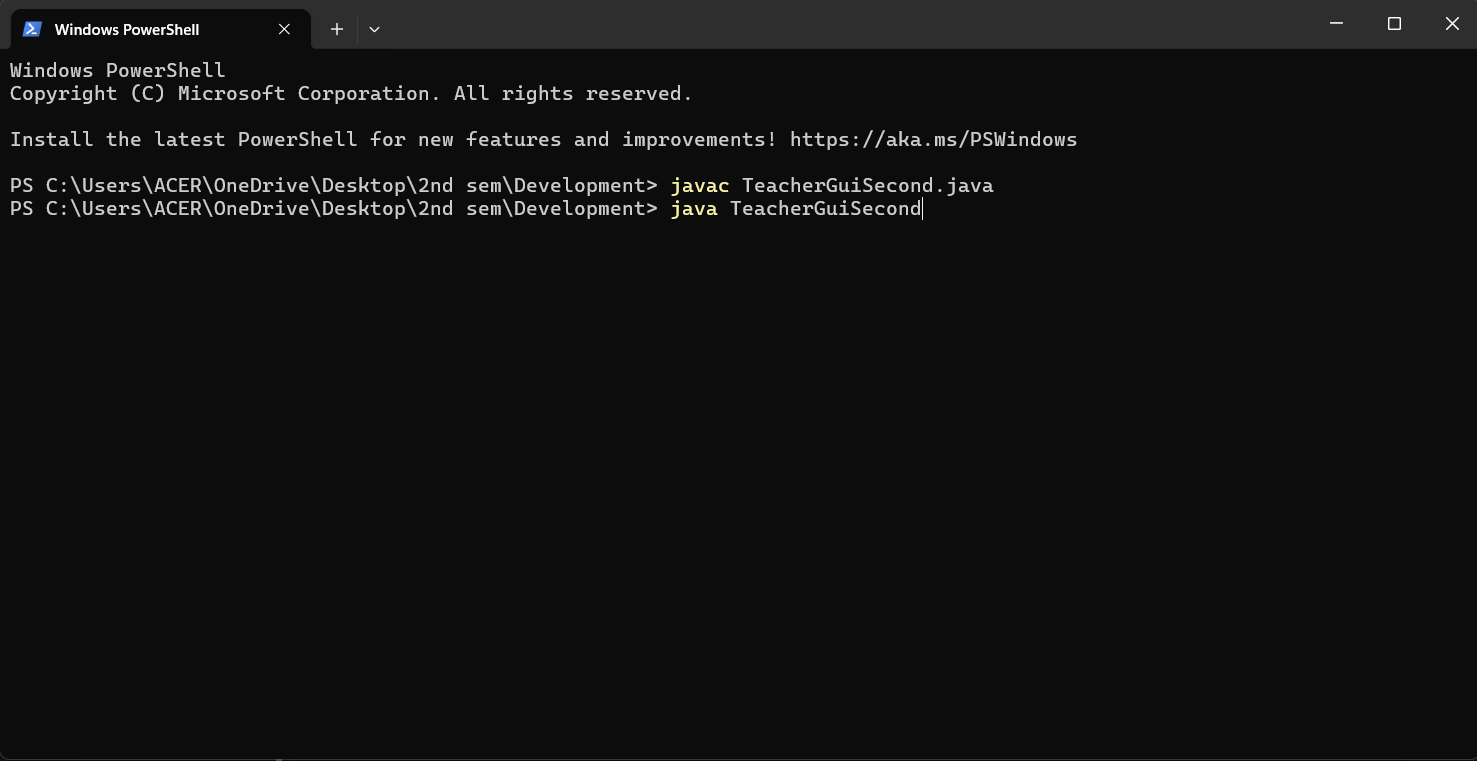


Figure 2::running program from command prompt

# 5.2Test2

|  |  |
| --- | --- |
| Objective | Checking the functionality of buttons |
| Action | 1. Add lecturer 2. Add Tutor 3. Grade Assignments of Lecturer 4. Set salary of Tutor 5. Remove Tutor |
| Expected outcome | When the respective buttons are clicked and they should function. |
| Actual Output | When respective buttons are clicked then they functions. |
| Conclusion | Test was successfully completed |

*Table 2:Description of Test 2*

5.2.1 Add Tutor

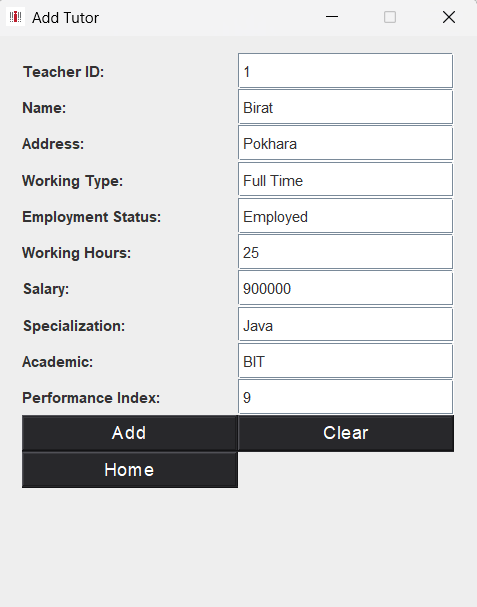


Figure 3:adding data in lecturer

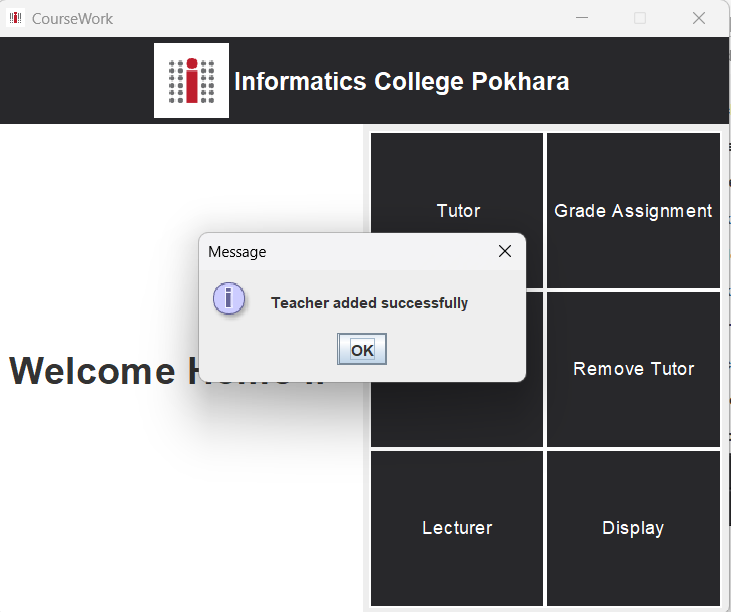


Figure 4:Display Dialog box

5.2.2 Remove Tutor

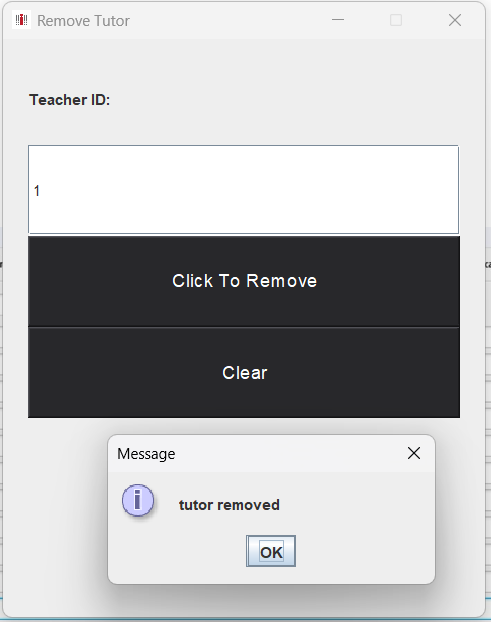


Figure 5: Removing Tutor

5.2.2 Add Lecturer

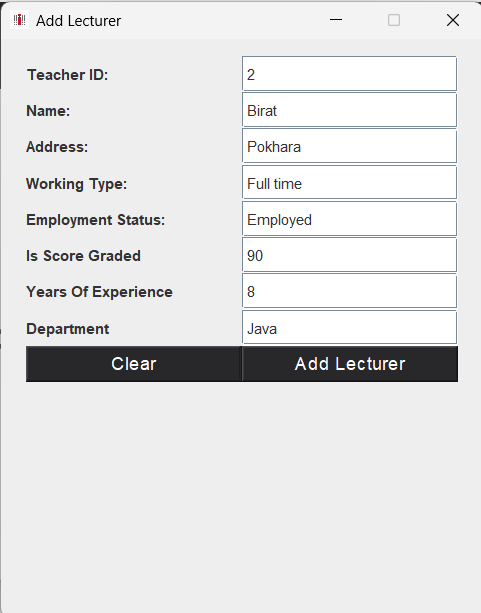


Figure 6:adding lecturer

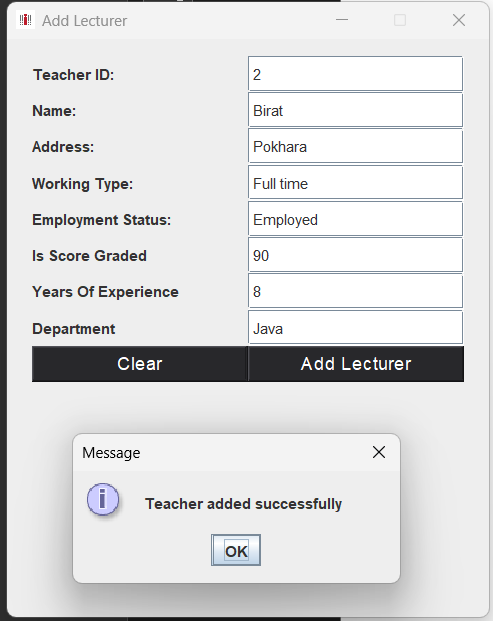


Figure 7:displaying message after adding lecturer

4.2.3 Grade Assignment

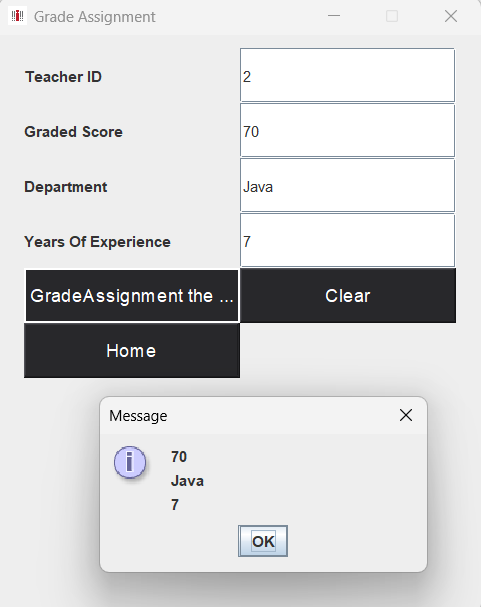


Figure 8:grading assignment

4.2.4 Set Salary

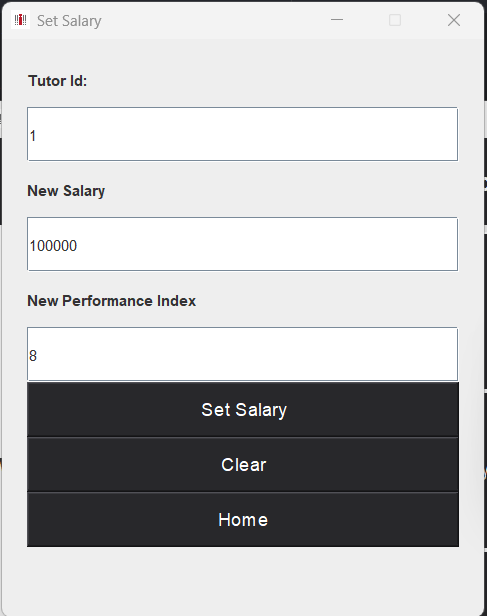


Figure 9:setting salary

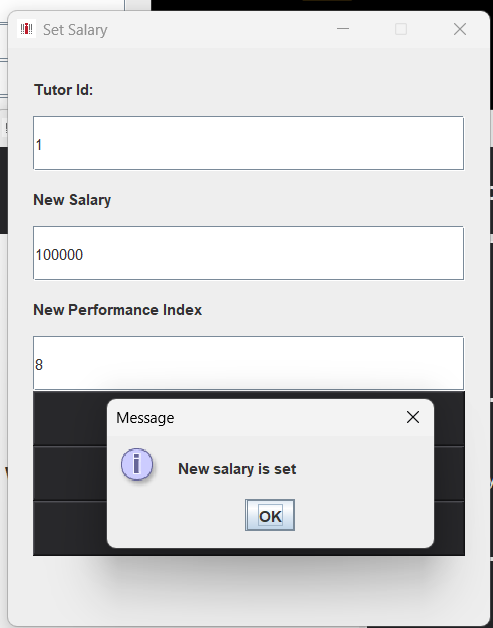


Figure 10:displaying message after setting salary

# 5.3Test3

|  |  |
| --- | --- |
| Objective | Testing Appropriate Dialog boxes when unsuitable values entered |
| Action | Input the incorrect value |
| Expected outcome | Error should come and pop up message should be shown  . |
| Actual Output | Error has come and pop up message is showed |
| Conclusion | Test was successfully completed |

*Table 3:Description of Test3*

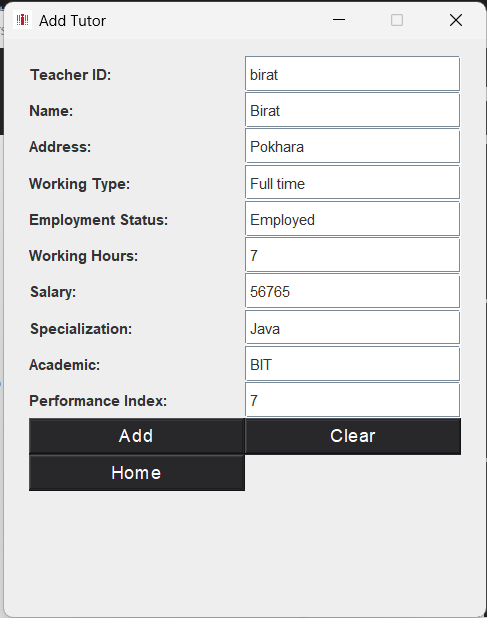


Figure 11:adding string value in interger

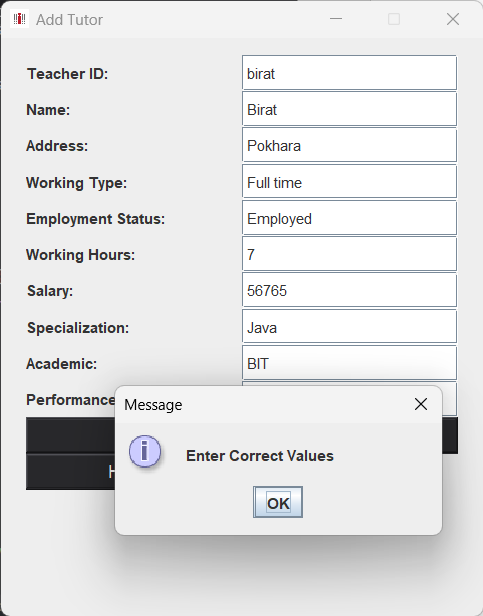


Figure 12:displaying after entering incorrect value

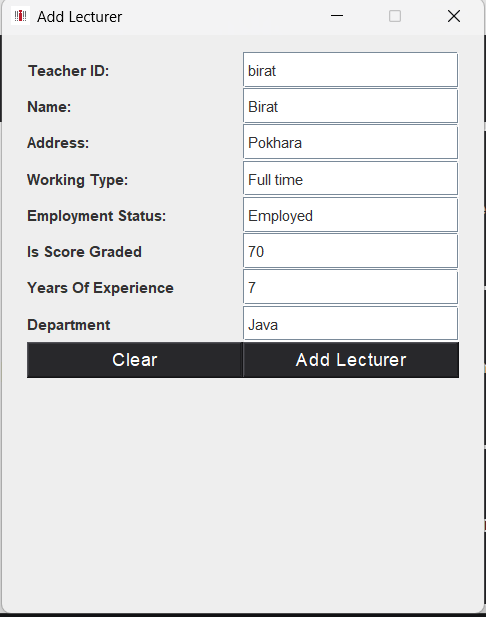


Figure 13:adding string value in integer for lecturer

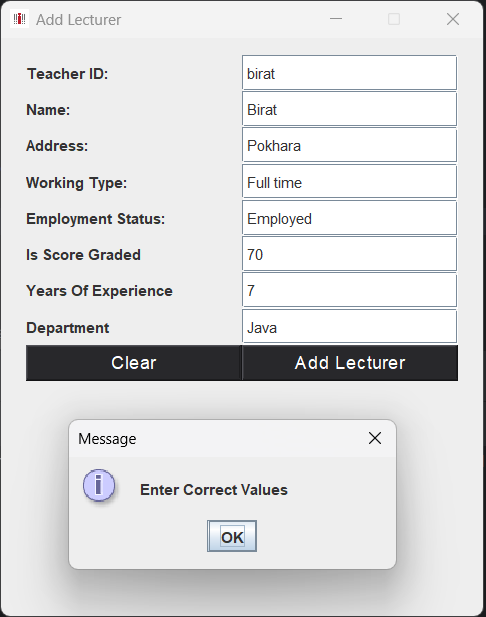


Figure 14:displaying message after wrong input

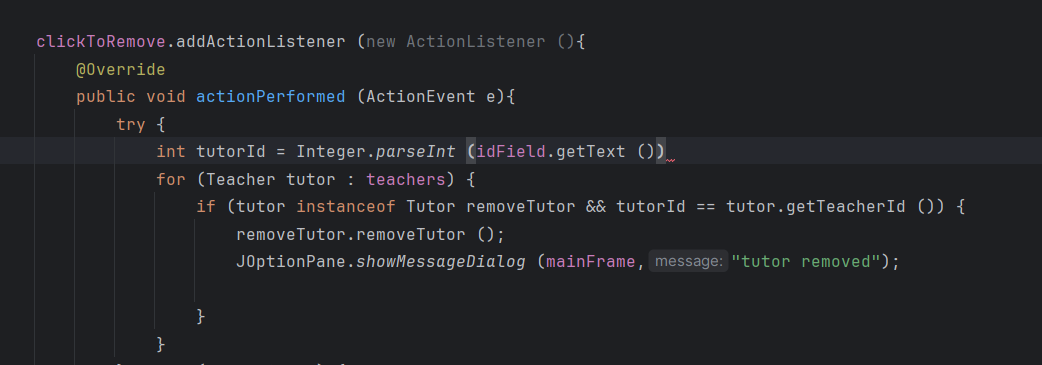
# 6. Error correction and detection

## 6.1 Syntax Error

Syntax errors occur when the source code of a computer program cont ains errors or does not follow the rules and syntax of programming lan guages. These errors may cause the program to not be written and run correctly. Syntax errors are recognized by the compiler or interpreter while converting source code into machine code.

Types of syntax error

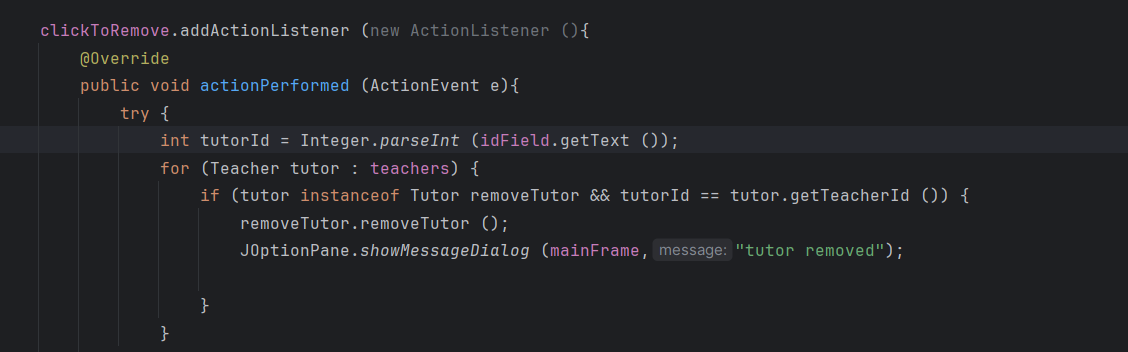
* Missing or extra parentheses
* Missing or extra braces
* Mismatched quotes
* Missing or misplaced semicolons
* Incorrectly nested loops or blocks
* Invalid variable or function names
* Incorrect use of operators **Error:**



*Figure 20:Syntax Error*

The Error occurred due to miss observation over the code. As the semi colon at the end is missing due to which syntax error occur.

**Correction:**

****

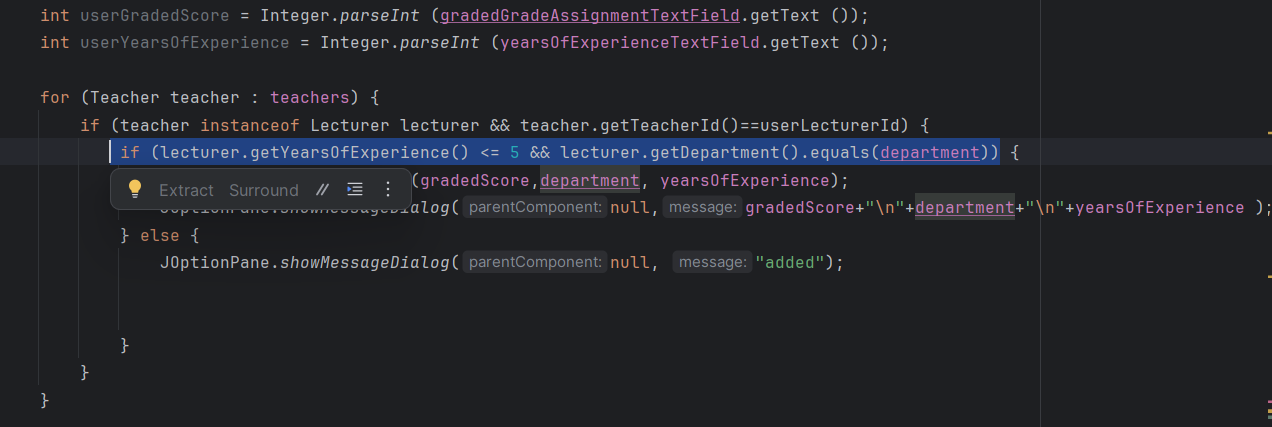
*Figure 21:Correction of Syntax Error*

The error was corrected after keeping the semi colon at the end of teacherId.

## 6.2 Logical Error

Logic errors are errors in the core of the program that cause unexpected and incorrect behaviour. It can cause crash of program while running because of logical error. Logical errors are not easily detected. This is because, such errors are valid when calculated in the language but not produce desire behaviour.

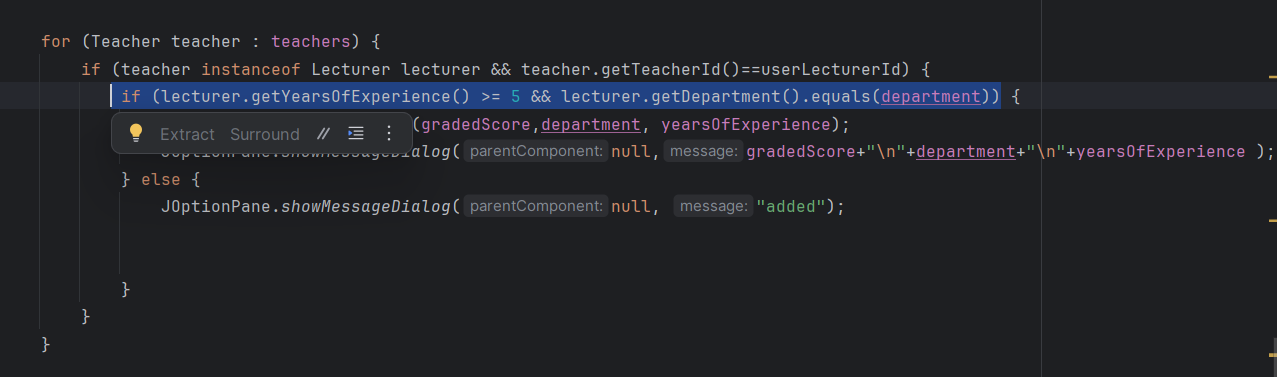
**Error:**



*Figure 22:Logical Error*

The error occurs as a sign mistake in the formula because (>) was replaced by (<).

**Correction:**

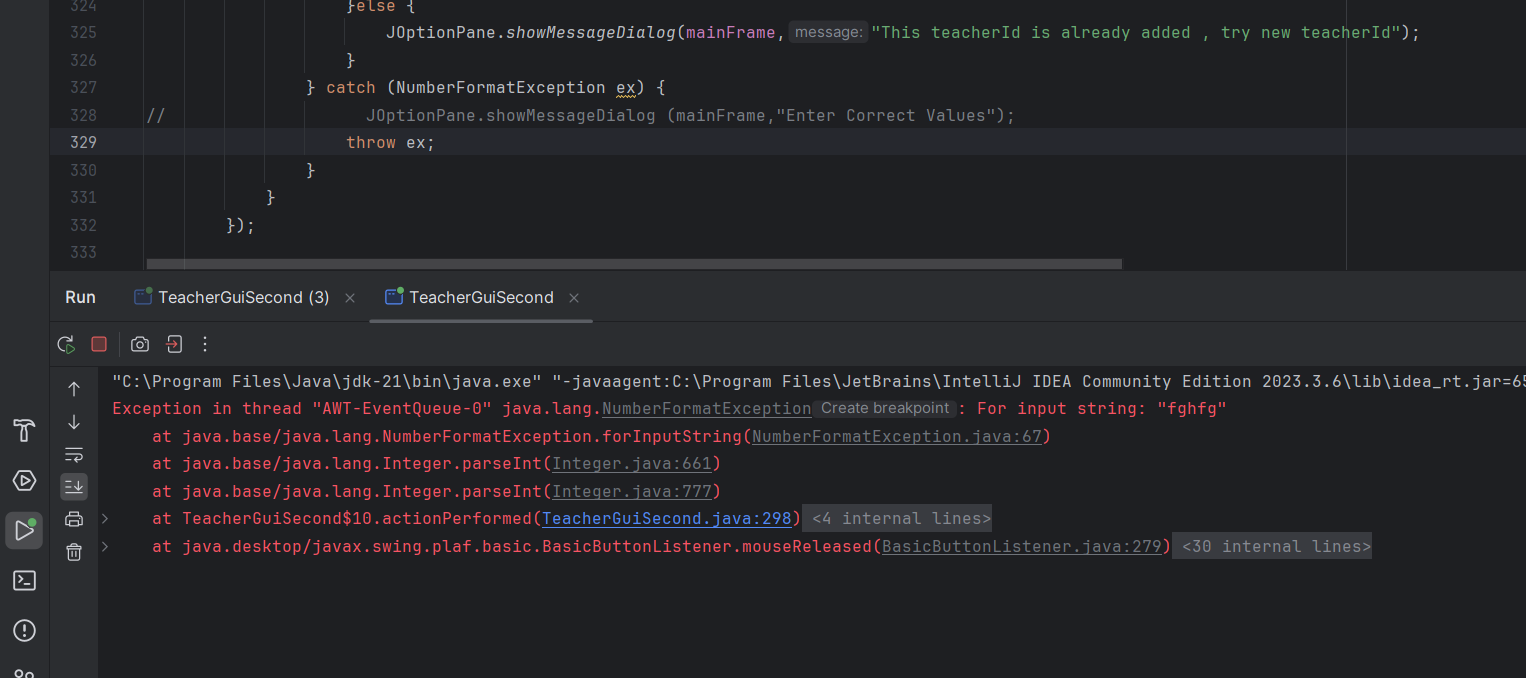


*Figure 23:Correction of logical Error*

The error was corrected after changing the (<) with (>) sign.

## 6.3 Runtime Error

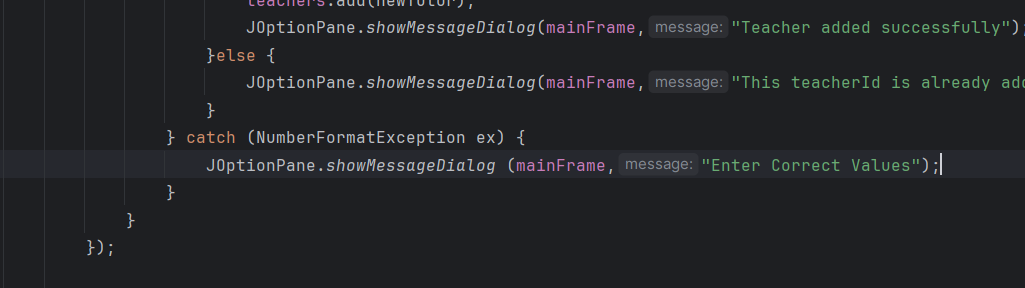
Problems occur when running the program. Time errors can occur in a software environment for many different reasons. These type of errors often occur in programming languages such as Java and Java Script, which convert code from line to machine language to execute it every time.



*Figure 24:Run time error*

This error occur as it throws the type error

**Correction:**



*Figure 26:Correction of run time error*

# 7. Conclusion

At first glance, the coursework posed a daunting challenge with its lengthy and intricate questions, calling for a unique and original design. However, upon deeper examination, I recognized familiar elements reminiscent of our previous semester's assignments. As I delved into the project, I soon realized that my understanding fell short of the task's requirements. Swiftly, I turned to revising lecture materials and sought guidance from both my teacher and peers.The main objective of the coursework was to integrate a Graphical User Interface (GUI) into our first-semester project, leveraging various layouts, Swing components. Through this endeavor, I gained valuable insights into creating user-friendly GUIs, exploring Swing properties and components. Additionally, I grasped the concept of managing teacher data using ArrayLists, as the coursework underscored the importance of UI/UX design.

Reflecting on the process, I encountered challenges stemming from minor errors that compounded into significant hurdles, leading to frustration during coding. Tasks such as adding data to ArrayLists and verifying the existence of teachers posed considerable difficulty due to my limited knowledge. Nevertheless, through perseverance, multiple revisions, and guidance from our tutor, I managed to surmount these obstacles and complete the coursework.While I successfully navigated the program, I acknowledge its limitations, particularly in terms of error handling and the potential for enhancing GUI aesthetics and data storage systems. Moving forward, I aim to refine my programming skills and address these shortcomings in future projects.

# 8.References

Rouse, M., 2013. *Techopedia.* [Online]

Available at: https://www.techopedia.com/definition/29530/bluej [Accessed 24 01 2024].

Rouse, M., 2013. *Techopedia/class-diagram.* [Online]

Available at: https://www.techopedia.com/definition/16466/class-diagram [Accessed 24 01 2024].

Scott, E., 2018. *Essay Corp.* [Online]

Available at: https://www.essaycorp.com/blog/java-programming-all-aboutjava

Ubah, K., 2021. *freecodecamo.org.* [Online]

Available at: https://www.freecodecamp.org/news/what-is-pseudocode-inprogramming/ [Accessed 24 01 2024].

# 9.Appendix

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

public class TeacherGuiSecond{

ArrayList<Teacher> teachers=new ArrayList<> ();

JFrame mainFrame = new JFrame ("CourseWork");

JPanel topPanel = new JPanel ();

JPanel centerPanel = new JPanel ();

JPanel leftCenterPanel = new JPanel ();

JPanel rightCenterPanel = new JPanel ();

JButton tutorButton = new JButton ("Tutor");

JButton lecturerButton = new JButton ("Lecturer");

JButton removeTutorButton = new JButton ("Remove Tutor");

JButton gradeAssignmentButton = new JButton ("Grade Assignment");

JButton setSalaryOfTutorButton = new JButton ("Set Salary");

JButton navHome = new JButton ("Home");

JLabel idLabel = new JLabel ("Teacher ID:");

JTextField idField = new JTextField (10);

JLabel nameLabel = new JLabel ("Name:");

JTextField nameField = new JTextField (10);

JLabel addressLabel = new JLabel ("Address:");

JTextField addressField = new JTextField (10);

JLabel workingTypeLabel = new JLabel ("Working Type:");

JTextField workingTypeField = new JTextField (10);

JLabel employmentStatusLabel = new JLabel ("Employment Status:");

JTextField employmentStatusField = new JTextField (10);

JLabel workingHoursLabel = new JLabel ("Working Hours:");

JTextField workingHoursField = new JTextField (10);

JLabel salaryLabel = new JLabel ("Salary:");

JTextField salaryField = new JTextField (10);

JLabel specializationLabel = new JLabel ("Specialization:");

JTextField specializationField = new JTextField (10);

JLabel academicQualificationLabel = new JLabel ("Academic:");

JTextField academicQualificationField = new JTextField (10);

JLabel performanceIndexLabel = new JLabel ("Performance Index:");

JTextField performanceIndexField = new JTextField (10);

JLabel gradedScoreLabel = new JLabel ("Is Score Graded");

JTextField gradedScoreTextField = new JTextField ();

JLabel departmentLabel = new JLabel ("Department");

JTextField departmentTextField = new JTextField ();

JLabel yearsOfExperienceLabel = new JLabel ("Years Of Experience");

JTextField yearsOfExperienceTextField = new JTextField ();

JButton addTutor = new JButton ("Add Tutor");

JButton addLecturer = new JButton ("Add Lecturer");

JButton clickToRemove = new JButton ("Click To Remove");

JButton grade = new JButton ("GradeAssignment the assignment");

JButton setSalary = new JButton ("Set Salary");

// JComboBox<Object> comboBox = new JComboBox<> ();

JButton addButton = new JButton ("Add");

JButton clear = new JButton ("Clear");

JButton display = new JButton ("Display");

JButton[] commonButtons = {addButton,clear,navHome};

public TeacherGuiSecond (){

mainGui ();

}

private void mainGui (){

JButton[] styling = {addButton,clear,navHome,addLecturer,clickToRemove,grade,setSalary,addTutor};

for (JButton button : styling) {

button.setFont (new Font ("Arial",Font.PLAIN,15));

button.setForeground (Color.WHITE);

button.setFocusPainted (false);

button.setBackground (new Color (40,40,43));

button.setBorder (BorderFactory.createRaisedBevelBorder ());

// button.setMargin (new Insets (2,2,2,2));

}

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

// Top Panel

topPanel.setBackground (new Color (40,40,43));

ImageIcon logo = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

JLabel topLabel = new JLabel ("Informatics College Pokhara");

topLabel.setFont (new Font ("TimesRoman",Font.BOLD,20));

topLabel.setForeground (Color.WHITE);

Image image = (logo).getImage ().getScaledInstance (60,60,Image.SCALE\_SMOOTH);

logo = new ImageIcon (image);

topLabel.setIcon (logo);

// center panel

leftCenterPanel.setBackground (Color.white);

// ImageIcon home = new ImageIcon ("D:\\JAVA\\project\\src\\home.png");

JLabel leftLabel = new JLabel ("Welcome Home !!");

leftLabel.setBorder (BorderFactory.createEmptyBorder (10,10,10,20));

leftLabel.setFont (new Font ("TimesRoman",Font.BOLD,30));

topLabel.setForeground (Color.WHITE);

leftCenterPanel.setLayout (new BorderLayout ());

leftCenterPanel.add (leftLabel);

rightCenterPanel.setLayout (new GridLayout (3,2));

JButton[] buttons = {tutorButton,gradeAssignmentButton,setSalaryOfTutorButton,removeTutorButton,lecturerButton,display};

for (JButton button : buttons) {

button.setFont (new Font ("Arial",Font.PLAIN,15));

button.setForeground (Color.WHITE);

button.setBackground (new Color (40,40,43));

button.setBorder (BorderFactory.createLineBorder (Color.WHITE,2));

button.setMargin (new Insets (2,2,2,2));

rightCenterPanel.add (button);

}

tutorButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

tutorGui ();

}

});

lecturerButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

lecturerGui ();

}

});

removeTutorButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

removeTutorGui ();

}

});

gradeAssignmentButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

gradeAssignmentGui ();

}

});

setSalaryOfTutorButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

setSalary ();

}

});

display.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

displayDetalis ();

}

});

topPanel.add (topLabel);

rightCenterPanel.setBorder (BorderFactory.createEmptyBorder (5,5,5,5));

centerPanel.setLayout (new GridLayout ());

centerPanel.add (leftCenterPanel,BorderLayout.WEST);

centerPanel.add (rightCenterPanel,BorderLayout.EAST);

mainFrame.setIconImage (icon.getImage ());

mainFrame.setDefaultCloseOperation (JFrame.EXIT\_ON\_CLOSE);

mainFrame.setSize (600,500);

mainFrame.setResizable (false);

mainFrame.setVisible (true);

mainFrame.setLocationRelativeTo (null);

mainFrame.add (topPanel,BorderLayout.NORTH);

mainFrame.add (centerPanel,BorderLayout.CENTER);

}

private void displayDetalis (){

JFrame displayFrame = new JFrame ("Teacher Details");

DefaultTableModel model = null;

JScrollPane scrollPane = null;

String[] colNamesTutor = {"Teacher Id","Teacher Name","Address","Working Type","Employment Status","Working Hours","Salary","Specialization","Academic","Performance Index"};

model = new DefaultTableModel (colNamesTutor,0);

for (Teacher teacher : teachers) {

if (teacher instanceof Tutor) {

Object[] rowData = {

teacher.getTeacherId (),

teacher.getTeacherName (),

teacher.getAddress (),

teacher.getWorkingType (),

teacher.getEmploymentStatus (),

((Tutor) teacher).getWorkingHours (),

((Tutor) teacher).getSalary (),

((Tutor) teacher).getSpecialization (),

((Tutor) teacher).getAcademicQualifications (),

((Tutor) teacher).getPerformanceIndex ()

};

model.addRow (rowData);

} else if (teacher instanceof Lecturer) {

Object[] rowData = {

teacher.getTeacherId (),

teacher.getTeacherName (),

teacher.getAddress (),

teacher.getWorkingType (),

teacher.getEmploymentStatus (),

((Lecturer) teacher).getGradedScore (),

((Lecturer) teacher).getYearsOfExperience (),

((Lecturer) teacher).getDepartment ()

};

model.addRow (rowData);

}

}

JTable table = new JTable (model);

scrollPane = new JScrollPane (table);

displayFrame.add (scrollPane);

displayFrame.setDefaultCloseOperation (JFrame.DISPOSE\_ON\_CLOSE);

displayFrame.pack ();

displayFrame.setLocationRelativeTo (null);

displayFrame.setVisible (true);

}

private void tutorGui (){

JFrame tutorFrame = new JFrame ("Add Tutor");

JPanel tutorPanel = new JPanel ();

tutorPanel.setLayout (new GridLayout (15,2));

tutorPanel.setBorder (BorderFactory.createEmptyBorder (10,20,10,20));

JButton home = new JButton ("Home");

home.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

tutorFrame.setVisible (false);

}

});

JTextField[] textFields = {idField,nameField,addressField,workingTypeField,employmentStatusField,workingHoursField,salaryField,specializationField,academicQualificationField,performanceIndexField};

JLabel[] labels = {idLabel,nameLabel,addressLabel,workingTypeLabel,employmentStatusLabel,workingHoursLabel,salaryLabel,specializationLabel,academicQualificationLabel,performanceIndexLabel};

for (int i = 0; i < textFields.length; i++) {

tutorPanel.add (labels[i]);

tutorPanel.add (textFields[i]);

textFields[i].setMargin (new Insets (2,2,2,2));

}

for (JButton btn : commonButtons) {

tutorPanel.add (btn);

}

navHome.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

tutorFrame.setVisible (false);

}

});

addButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

tutorObject ();

}

});

// setting icon for the frame

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

tutorFrame.setIconImage (icon.getImage ());

// title image

ImageIcon logo = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

JLabel topLabel = new JLabel ();

Image image = (logo).getImage ().getScaledInstance (60,60,Image.SCALE\_SMOOTH);

logo = new ImageIcon (image);

topLabel.setIcon (logo);

tutorFrame.add (tutorPanel);

tutorFrame.setDefaultCloseOperation (JFrame.HIDE\_ON\_CLOSE);

tutorFrame.setSize (400,500);

tutorFrame.setLocationRelativeTo (null);

tutorFrame.setResizable (false);

tutorFrame.setVisible (true);

}

Tutor newTutor;

private void tutorObject (){

addButton.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

try {

int tutorId = Integer.parseInt (idField.getText ());

int workingHours = Integer.parseInt (workingHoursField.getText ());

double salary = Double.parseDouble (salaryField.getText ());

int performanceIndex = Integer.parseInt (performanceIndexField.getText ());

String specialization = specializationField.getText ();

String academicsQualification = academicQualificationField.getText ();

String tutorName = nameField.getText ();

String address = addressField.getText ();

String workingType = workingTypeField.getText ();

String employmentStatus = employmentStatusField.getText ();

boolean isCertified = false;

int Idindex =-1;

for(Teacher teacher: teachers){

System.out.println(teacher.getTeacherId());

if(teacher.getTeacherId() == tutorId){

Idindex = teachers.indexOf(teacher);

break;

}

}

if(Idindex==-1){

Tutor newTutor =new Tutor(tutorId,tutorName,address,workingType,employmentStatus ,workingHours,salary,specialization,academicsQualification,performanceIndex,isCertified);

teachers.add(newTutor);

JOptionPane.showMessageDialog(mainFrame,"Teacher added successfully");

}else {

JOptionPane.showMessageDialog(mainFrame,"This teacherId is already added , try new teacherId");

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog (mainFrame,"Enter Correct Values");

}

}

});

clear.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

idField.setText ("");

nameField.setText ("");

addressField.setText ("");

workingTypeField.setText ("");

employmentStatusField.setText ("");

workingHoursField.setText ("");

salaryField.setText ("");

academicQualificationField.setText ("");

performanceIndexField.setText ("");

specializationField.setText ("");

}

});

}

// Lecturer newLecturer;

private void lecturerGui (){

JFrame lecturerFrame = new JFrame ("Add Lecturer");

JPanel lecturerPanel = new JPanel ();

lecturerPanel.setLayout (new GridLayout (15,2));

lecturerPanel.setBorder (BorderFactory.createEmptyBorder (10,20,10,20));

navHome.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

lecturerFrame.setVisible (false);

}

});

// System.out.println (labels.length);

// System.out.println (textFields.length);

JTextField[] textFields = {idField,nameField,addressField,workingTypeField,employmentStatusField,gradedScoreTextField,yearsOfExperienceTextField,departmentTextField};

JLabel[] labels = {idLabel,nameLabel,addressLabel,workingTypeLabel,employmentStatusLabel,gradedScoreLabel,yearsOfExperienceLabel,departmentLabel};

for (int i = 0; i < textFields.length; i++) {

lecturerPanel.add (labels[i]);

lecturerPanel.add (textFields[i]);

textFields[i].setMargin (new Insets (2,2,2,2));

}

lecturerPanel.add (clear);

lecturerPanel.add (addLecturer);

addLecturer.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

lecturerObject ();

}

});

// setting icon for the frame

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

lecturerFrame.setIconImage (icon.getImage ());

// title image

ImageIcon logo = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

JLabel topLabel = new JLabel ();

Image image = (logo).getImage ().getScaledInstance (60,60,Image.SCALE\_SMOOTH);

logo = new ImageIcon (image);

topLabel.setIcon (logo);

lecturerFrame.add (lecturerPanel);

lecturerFrame.setDefaultCloseOperation (JFrame.HIDE\_ON\_CLOSE);

lecturerFrame.setSize (400,500);

lecturerFrame.setLocationRelativeTo (null);

lecturerFrame.setResizable (false);

lecturerFrame.setVisible (true);

}

int lecturerId;

String lecturerName;

String address;

String workingType;

String employmentStatus;

int yearsOfExperience;

String department;

int gradedScore;

Lecturer newLecturer;

private void lecturerObject (){

try {

lecturerId = Integer.parseInt (idField.getText ());

lecturerName = nameField.getText ();

address = addressField.getText ();

workingType = workingTypeField.getText ();

employmentStatus = employmentStatusField.getText ();

gradedScore = Integer.parseInt (gradedScoreTextField.getText ());

yearsOfExperience = Integer.parseInt (yearsOfExperienceTextField.getText ());

department = departmentTextField.getText ();

int Idindex =-1;

for(Teacher teacher: teachers){

System.out.println(teacher.getTeacherId());

if(teacher.getTeacherId() == lecturerId){

Idindex = teachers.indexOf(teacher);

break;

}

}

if(Idindex==-1){

Lecturer newLecturer =new Lecturer(lecturerId,lecturerName,address,workingType,employmentStatus,yearsOfExperience,department);

teachers.add(newLecturer);

JOptionPane.showMessageDialog(mainFrame,"Teacher added successfully");

}else {

JOptionPane.showMessageDialog(mainFrame,"This teacherId is already added , try new teacherId");

}

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog (mainFrame,"Enter Correct Values");

}

clear.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

idField.setText ("");

nameField.setText ("");

addressField.setText ("");

workingTypeField.setText ("");

employmentStatusField.setText ("");

gradedScoreTextField.setText ("");

yearsOfExperienceTextField.setText ("");

departmentTextField.setText ("");

}

});

}

private void gradeAssignmentGui (){

JFrame gradeAssignmentFrame = new JFrame ("Grade Assignment");

JPanel gradeAssignmentPanel = new JPanel ();

gradeAssignmentPanel.setLayout (new GridLayout (10,1));

gradeAssignmentPanel.setSize (300,300);

gradeAssignmentPanel.setBorder (BorderFactory.createEmptyBorder (10,20,10,20));

JLabel idGradeAssignment = new JLabel ("Teacher ID");

JLabel gradedGradeAssignment = new JLabel ("Graded Score");

JLabel departmentGradeAssignment = new JLabel ("Department");

JLabel yearsOfExperienceGradeAssignment = new JLabel ("Years Of Experience");

JTextField idGradeAssignmentTextField = new JTextField ();

JTextField gradedGradeAssignmentTextField = new JTextField ();

JTextField departmentGradeAssignmentTextField = new JTextField ();

JTextField yearsOfExperienceGradeAssignmentTextField = new JTextField ();

gradeAssignmentPanel.add (idGradeAssignment);

gradeAssignmentPanel.add (idGradeAssignmentTextField);

gradeAssignmentPanel.add (gradedGradeAssignment);

gradeAssignmentPanel.add (gradedGradeAssignmentTextField);

gradeAssignmentPanel.add (departmentGradeAssignment);

gradeAssignmentPanel.add (departmentGradeAssignmentTextField);

gradeAssignmentPanel.add (yearsOfExperienceGradeAssignment);

gradeAssignmentPanel.add (yearsOfExperienceGradeAssignmentTextField);

grade.setFont (new Font ("Arial",Font.PLAIN,15));

grade.setForeground (Color.WHITE);

grade.setBackground (new Color (40,40,43));

grade.setBorder (BorderFactory.createLineBorder (Color.WHITE,2));

grade.setMargin (new Insets (2,2,2,2));

gradeAssignmentPanel.add (grade);

gradeAssignmentPanel.add (clear);

gradeAssignmentPanel.add (navHome);

navHome.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

gradeAssignmentFrame.setVisible (false);

}

});

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

gradeAssignmentFrame.setIconImage (icon.getImage ());

gradeAssignmentFrame.add (gradeAssignmentPanel);

gradeAssignmentFrame.setDefaultCloseOperation (JFrame.HIDE\_ON\_CLOSE);

gradeAssignmentFrame.setSize (400,500);

gradeAssignmentFrame.setLocationRelativeTo (null);

gradeAssignmentFrame.setResizable (false);

gradeAssignmentFrame.setVisible (true);

String department = departmentGradeAssignmentTextField.getText ();

grade.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

try {

int userLecturerId = Integer.parseInt (idGradeAssignmentTextField.getText ());

int userGradedScore = Integer.parseInt (gradedGradeAssignmentTextField.getText ());

int userYearsOfExperience = Integer.parseInt (yearsOfExperienceTextField.getText ());

for (Teacher teacher : teachers) {

if (teacher instanceof Lecturer lecturer && teacher.getTeacherId()==userLecturerId) {

if (lecturer.getYearsOfExperience() >= 5 && lecturer.getDepartment().equals(department)) {

lecturer.gradeAssignment (gradedScore,department, yearsOfExperience);

JOptionPane.showMessageDialog(null,gradedScore+"\n"+department+"\n"+yearsOfExperience );

} else {

JOptionPane.showMessageDialog(null, "added");

//

}

}

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog (mainFrame,"Please enter valid numeric values.");

}

}

});

}

private void removeTutorGui (){

JFrame removeFrame = new JFrame ("Remove Tutor");

JPanel removePanel = new JPanel ();

removePanel.setLayout (new GridLayout (6,1));

removePanel.setSize (300,300);

removePanel.setBorder (BorderFactory.createEmptyBorder (10,20,10,20));

navHome.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

removeFrame.setVisible (false);

}

});

clickToRemove.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

try {

int tutorId = Integer.parseInt (idField.getText ());

for (Teacher tutor : teachers) {

if (tutor instanceof Tutor removeTutor && tutorId == tutor.getTeacherId ()) {

removeTutor.removeTutor ();

JOptionPane.showMessageDialog (mainFrame,"tutor removed");

}

}

} catch (Exception f) {

JOptionPane.showMessageDialog (mainFrame,"Please make Tutor Object First or enter the correct value");

}

}

});

removePanel.add (idLabel);

removePanel.add (idField);

removePanel.add (clickToRemove);

removePanel.add (clear);

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

removeFrame.setIconImage (icon.getImage ());

// title image

ImageIcon logo = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

JLabel topLabel = new JLabel ();

Image image = (logo).getImage ().getScaledInstance (60,60,Image.SCALE\_SMOOTH);

logo = new ImageIcon (image);

topLabel.setIcon (logo);

removeFrame.add (removePanel);

removeFrame.setDefaultCloseOperation (JFrame.HIDE\_ON\_CLOSE);

removeFrame.setSize (400,500);

removeFrame.setLocationRelativeTo (null);

removeFrame.setResizable (false);

removeFrame.setVisible (true);

clear.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

idField.setText ("");

// JOptionPane.showMessageDialog (mainFrame,tutorId + "\n" + tutorName + "\n" + address + "\n" + workingType + "\n" + employmentStatus +"\n"+gradedScore +"\n" + yearsOfExperience);

}

});

}

private void setSalary (){

JFrame setSalaryFrame = new JFrame ("Set Salary");

JPanel setSalaryPanel = new JPanel ();

setSalaryPanel.setLayout (new GridLayout (10,2));

setSalaryPanel.setBorder (BorderFactory.createEmptyBorder (10,20,10,20));

JLabel newSalaryLabel = new JLabel ("New Salary");

JTextField newSalaryTextField = new JTextField ();

JLabel tutorLabel = new JLabel ("Tutor Id:");

JTextField tutorTextField = new JTextField ();

setSalaryPanel.add (tutorLabel);

setSalaryPanel.add (tutorTextField);

setSalaryPanel.add (newSalaryLabel);

setSalaryPanel.add (newSalaryTextField);

JLabel newPerformanceIndexLabel = new JLabel ("New Performance Index");

JTextField newPerformanceTextField = new JTextField ();

setSalaryPanel.add (newPerformanceIndexLabel);

setSalaryPanel.add (newPerformanceTextField);

// setSalaryPanel.add (addButton);

setSalaryPanel.add (setSalary);

setSalaryPanel.add (clear);

setSalaryPanel.add (navHome);

navHome.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

setSalaryFrame.setVisible (false);

}

});

setSalary.addActionListener (new ActionListener (){

@Override

public void actionPerformed (ActionEvent e){

String tutorId = tutorTextField.getText ().trim ();

String tutorSalary = newSalaryTextField.getText ().trim ();

String tutorPerformanceIndex = performanceIndexField.getText ().trim ();

try {

int userInputId = Integer.parseInt (tutorId);

double userInputSalary = Double.parseDouble (tutorSalary);

int userTutorPerformanceIndex = Integer.parseInt (tutorPerformanceIndex);

for (Teacher tutor : teachers) {

// Tutor removeTutor = (Tutor) tutor;

if (tutor instanceof Tutor && tutor.getTeacherId() == Integer.parseInt(tutorTextField.getText())) {

//

if (userTutorPerformanceIndex >= 0 && userTutorPerformanceIndex <= 10) {

((Tutor) tutor).setSalary (userInputSalary , userTutorPerformanceIndex);

JOptionPane.showMessageDialog(null, "New salary is set");

return;

}

else {

JOptionPane.showMessageDialog(null,"Performance Index should be from 0 to 10");

return;

}

}

}

} catch (Exception f) {

JOptionPane.showMessageDialog (mainFrame,"Enter correct value");

}

}

});

ImageIcon icon = new ImageIcon ("D:\\JAVA\\project\\src\\icp-final-logo.png");

setSalaryFrame.setIconImage (icon.getImage ());

setSalaryFrame.add (setSalaryPanel);

setSalaryFrame.setDefaultCloseOperation (JFrame.HIDE\_ON\_CLOSE);

setSalaryFrame.setSize (400,500);

setSalaryFrame.setLocationRelativeTo (null);

setSalaryFrame.setResizable (false);

setSalaryFrame.setVisible (true);

}

public static void main (String[] args){

new TeacherGuiSecond ();

}

}