



slington college
(इस्लिङ्टन कलेज)

Module Code & Module Title
CS4001NI Programming
COURSEWORK-2

Assessment Weightage & Type
30% Individual Coursework
Semester and Year
Spring 2021

Student Name: Prithab Raj Adhikari
Group: C13

London Met ID: 20048966

College ID: NP01CP4S210308

Assignment Due Date: 20th August,2021

Assignment Submission Date: 20th August,2021

I confirm that I understand my coursework needs to be submitted online via Google classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submission will be treated as non-submission and a mark of zero will be awarded.

Contents

TABLE OF FIGURES:	3
TABLE OF TABLES:.....	4
Introduction:	5
• Introduction to The Assigned Work	5
CLASS DIAGRAM.....	7
ING class diagram.....	8
PSEUDO CODE	9
ING College Class Code	9
METHOD DESCRIPTION	36
• Get () Method.....	36
• ING_College () Method:	38
• actionPerformed (Action Event e) Method:	38
• Main (String args []) Method:	39
TESTS: -	39
• TEST 1:.....	39
• TEST 2:.....	41
• TEST 3:.....	45
• TEST 4:.....	50
ERROR DETECTION.....	51
• SYNTAX ERROR: -	51
• IMPORT ERROR:	54
• SEMANTIC ERROR:	56
CONCLUSION:.....	58
APPENDIX:	59
• COURSE CLASS:	59
• ACADEMIC COURSE CLASS:	61
• NON-ACADEMIC COURSE CLASS:.....	64
• ING_College Course Appendix:	68

TABLE OF FIGURES:

Figure 1: Running the codes of the classes using CMD prompt.	40
Figure 2:Evidence that the code for Academic Course runs	40
Figure 3:Evidence that the code for Non-Academic Course runs	41
Figure 4:Data for adding academic course	42
Figure 5:Dialogue Box for adding academic course	42
Figure 6: Adding data for registering academic course.....	43
Figure 7: Dialogue box for adding academic course.....	43
Figure 8:Data for adding nonacademic course	43
Figure 9:Dialogue Box for added non academic course	43
Figure 10: Data for registering non academic course class	44
Figure 11: Dialogue Box for registered non academic course	44
Figure 12: Dialogue Box for removing non academic course.....	45
Figure 13:Adding academic course and getting the dialogue	46
Figure 14: Repeating the academic course ID and getting error box	46
Figure 15:Dialogue box for registering academic course	47
Figure 16: Adding non academic course and getting appropriate dialogue box	47
Figure 17:registering the nonacademic course class successfully.....	48
Figure 18: trying to reregister the same curse and getting error box.....	48
Figure 19:Dialogue box after removing the nonacademic course class	49
Figure 20: Displaying all the entered data of Academic Course	50
Figure 21: Displaying all the entered data of Non-Academic Course.....	51
Figure 22: Syntax Error	52
Figure 23: Correction of Syntax Error	53
Figure 24: Import error	54
Figure 25:Correction of import error	55
Figure 26: SEMANTIC ERROR.....	56
Figure 27: Correction of SEMANTIC ERROR.....	57

TABLE OF TABLES:

Table 1:Representation.....	7
Table 2:Class Diagram of ING College Class.....	8
Table 3:Table for describing getters method	38
Table 4: TEST 1 For compiling and running program in CMD prompt	39
Table 5: TEST 2 for checking ADD , REGISTER and REMOVE buttons.....	41
Table 6: TEST 3 for checking repeated operations	45
Table 7: TEST 4 for displaying	50

Introduction:

- Introduction to The Assigned Work

This project was created in order to form a new class and add it to the previous coursework in order to make a graphical user interface (GUI) in BlueJ for a system that stores the details of Course class which includes the details of both the academic and non-academic course. Thus, the project includes four classes in total which are; ING_College, Course, AcademicCourse, NonAcademicCourse.

The Course class. AcademicCourse class and NonAcademicCourse is connected to ING_College with the help of array list made inside the ING_College. The ING_College consists of various aspects that enhance the GUI and the entire program.

Inside the ING_College there are various packages like java swing package, java util package, awt packages which are imported for the construction of GUI. There are multiple components of Java swing like Frame, JComboBox, JLabel, JTextField and many others. This class also contains the components of Java awt package like Panel, Dialogue Box, Checkbox etc. These components work together to create a Frame which consists of two panels each for storing the details of AcademicCourse class and another for storing the details of NonAcademicCourse.

There are multiple buttons which carries their own functions i.e.:

- Add Button for Academic Course:

The add button is used to add the academic course to the GUI after all the details are entered.

- Register Button for Academic Course:

This button is used to register the Academic course after entering all the values.

- Display Button for Academic Course:
When this button is pressed, the details will be displayed in the text area of the GUI.
- Clear Button for Academic Course:
When this button is pressed, all the details are cleared from the GUI.
- Add Button for NonAcademic Course:
The add button is used to add the nonacademic course to the GUI after all the details are entered.
- Register Button for NonAcademic Course:
This button is used to register the NonAcademic course after entering all the values.
- Display Button for NonAcademic Course:
When this button is pressed, the details will be displayed in the text area of the GUI.
- Clear Button for NonAcademic Course:
When this button is pressed, all the details are cleared from the GUI.
- Remove Button for NonAcademic Course:
When this button is pressed, the course is removed.

Various components of Java swing package, Java awt package, Java, different methods and functions has been put together for the building ING_College class.

BlueJ is used for coding as it is easily available with favorable Java Development Environment (JDE) for student and beginners and for the report

part we used Microsoft Word as it provides us with easy access to the components such as tables, inserting figures, etc.

CLASS DIAGRAM

- A class diagram is a representation of the attributes of the various classes with proper arrangement of the methods called. The following signs are used to denote the attributes and represent the methods:

1.	PRIVATE	-
2.	PUBLIC	+
3.	PROTECTED	#
4.	DEFAULT	~

Table 1:Representation

ING class diagram

ING College		
-frame: JFrame -panel1: JPanel -panel2: JPanel -title: JLabel -title2: JLabel -CourseID: JLabel -Course Name: JLabel -Course Leader: JLabel -Lecturer Name: JLabel -Level: JLabel -Credit: JLabel -NoOfAssessments: JLabel -S1: JLabel -C1: JLabel -Duration: JLabel -CourseID1: JLabel -Course Name1: JLabel -Course Leader1: JLabel -Instructor Name: JLabel -S1: JLabel -S2: JLabel -E1: JLabel -res: JLabel -res1: JLabel -D1: JLabel -P1: JLabel	-Course_ID: JTextField -Course_Name: JTextField -Course_Leader: JTextField -Lecture_Name: JTextField -level: JTextField -credit: JTextField -No_Of_Assessments: JTextField -duration: JTextField -Course_ID1: JTextField -Course_Name1: JTextField -Course_Leader1: JTextField -Instructor_Name: JTextField -d1: JTextField -date_1: JComboBox -month_1: JComboBox -year_1: JComboBox -date_2: JComboBox -month_2: JComboBox -year_2: JComboBox Dates, Months, Years: String [] -ArrayList<Course>list: ArrayList -Academic: AcademicCourse -nonAcademic: NonAcademic Course	-year: JComboBox -date: JComboBox -year: JComboBox -term: JCheckBox -term1: JCheckBox -Add: JButton -Clear: JButton -Register: JButton -Display: JButton -Add1: JButton -Register1: JButton -Clear1: JButton -Display1: JButton -Remove: JButton -Add_NC: JButton -Add_C: JButton -tout: JTextArea -tout1: JTextArea
+ ING_College (): void + AcademicCourse (): void + NonAcademicCourse (): void + ING_College (): void + actionPerformed (ActionEvent e) (): void +getCourseID: String +getCourseName: String +getDuration: String +getCourseLeader: String +getLecturerName: String +getLevel: String +main (String [] args): void	+getCredit: String +getNoOfAssessments: String +getS1: String +getC1: String +getCourseID1: String +getCourseName1: String +getCourseLeader1: String +getD1: String +getS2: String +getC2: String +getE1: String +getInstructorName: String +getP1: String	

Table 2:Class Diagram of ING College Class

PSEUDO CODE

ING College Class Code

Import javax. swing. *

Import java.awt. Font

Import java.awt. *

Import java.awt. Color

Import java. util.ArrayList

Import java.awt. event.(ActionEvent)

Import java.awt. event. ActionListener

CREATE ING_College implementing ACTIONLISTENER

DO

INITIALIZE JFrame

Frame

INITIALIZE JPanel;

panel1

INITIALIZE JLabel

Title, CourseID, CourseName, CourseLeader, LecturerName, Level, Credit,
NoOfAssessments, S1, C1, Duration, res

INITIALIZE JTextField

Course_ID, Course_Name, Course_Leader, Lecturer_Name, level, credit,
NO_Of_Assessments, duration

INITIALIZE JComboBox

date_1, month_1, year_1, date_2, month_2, year_2

INITIALIZE JCheckBox

Term1

INITIALIZE JTextArea

tout

INITIALIZE JButton

Add, Display, Clear, Register, Add_NC

SETPANEL frame

SETTITE panel1

SETBACKGROUND panel1

SETTEXT Title

Add Title

SETLOCATION Title

SETFONT Title

SETTEXT

CourseID, CourseName, CourseLeader, LecturerName, Level, Credit,
NoOfAssessments, S1, C1 Duration, res, Course_ID, Course_Name, Course_Leader,
Lecturer_Name, level, credit, NO_Of_Assessments, duration, date_1, month_1, year_1,
date_2, month_2, year_2, Term1, tout , Add, Register, Display, Clear, Add_NC

SETLOCATION

CourseID, CourseName, CourseLeader, LecturerName, Level, Credit,
NoOfAssessments, S1, C1 Duration, res, Course_ID, Course_Name, Course_Leader,
Lecturer_Name, level, credit, NO_Of_Assessments, duration, date_1, month_1, year_1,
date_2, month_2, year_2, Term1, tout , Add, Register, Display, Clear, Add_NC

SETFONT

CourseID, CourseName, CourseLeader, LecturerName, Level, Credit,
NoOfAssessments, S1, C1 Duration, res, Course_ID, Course_Name, Course_Leader,
Lecturer_Name, level, credit, NO_Of_Assessments, duration, date_1, month_1, year_1,
date_2, month_2, year_2, Term1, tout , Add, Register, Display, Clear, Add_NC

ADD ACTION LISTINER

Add, Display, Clear, Register, Add_NC

ADD

CourseID, CourseName, CourseLeader, LecturerName, Level, Credit,
NoOfAssessments, S1, C1 Duration, res, Course_ID, Course_Name, Course_Leader,
Lecturer_Name, level, credit, NO_Of_Assessments, duration, date_1, month_1, year_1,
date_2, month_2, year_2, Term1, tout , Add, Register, Display, Clear, Add_NC

INITIALIZE JLabel

Title1, CourseID1, CourseName1, CourseLeader1, InstructorName, Level, Credit,
P1, S2, C2, E1, D1, res1

INITIALIZE JTextField

Course_ID1, Course_Name1, Course_Leader1, Instructor_Name, p1, d1

INITIALIZE JComboBox

date_1, month_1, year_1, date_2, month_2, year_2, date, month, year

INITIALIZE JButton

Add1, Remove, Display1, Clear1, Register1, Add_C

INITIALIZE JCheckBox

Term

INITIALIZE JTextArea

Tout1

SETTITE panel2

SETBACKGROUND panel2

\SETTEXT Title1

SETBOUNDS Title1

Add Title1

SETFONT Title1

SETTEXT

CourseID1, CourseName1, CourseLeader1, InstructorName, P1, E1, S2, C2, D1, res,
Course_ID1, Course_Name1, Course_Leader1, Instructor_Name, p1, d1, date_1,
month_1, year_1, date_2, month_2, year_2, date, month, year, Term, tout1 , Add1,
Register1, Display1, Clear1,Remove, Add_C

SETLOCATION

CourseID1, CourseName1, CourseLeader1, InstructorName, P1, E1, S2, C2 D1, res,
Course_ID1, Course_Name1, Course_Leader1, Instructor_Name, p1, d1, date_1,
month_1, year_1, date_2, month_2, year_2, date, month, year, Term, tout1 , Add1,
Register1, Display1, Clear1,Remove, Add_C

SETFONT

CourseID1, CourseName1, CourseLeader1, InstructorName, P1, E1, S2, C2 D1, res,
Course_ID1, Course_Name1, Course_Leader1, Instructor_Name, p1, d1, date_1,
month_1, year_1, date_2, month_2, year_2, date, month, year, Term, tout1 , Add1,
Register1, Display1, Clear1,Remove, Add_C

ADD ACTION LISTINER

Add1, Remove, Display1, Clear1, Register1, Add_C

ADD

CourseID1, CourseName1, CourseLeader1, InstructorName, P1, E1, S2, C2 D1, res,
Course_ID1, Course_Name1, Course_Leader1, Instructor_Name, p1, d1, date_1,
month_1, year_1, date_2, month_2, year_2, date, month, year, Term, tout1 , Add1,
Register1, Display1, Clear1, Remove, Add_C

SETLAYOUT OF frame **TO** null

SET VISIBILITY OF frame **TO** true

SETRESIZABLE OF frame **TO** true

SETDEFAULTCLOSEOPERATION OF frame **TO** JFrame.EXIT_ON_CLOSE

FUNCTION actionPerformed (ActionEvent e)

DO

IF (e.getSource () ==Add_NC)

SETVISIBILITY OF panel1 **TO** false

SETVISIBILITY OF panel2 **TO** true

END IF

IF (e.getSource () ==Add_C)

SETVISIBILITY OF panel1 **TO** true

SETVISIBILITY OF panel2 **TO** false

END IF

DO

IF (e. getSource () ==Add)

DO

TRY

String CourseID = Course_ID. getText ()

String CourseName = Course_Name. getText ()

String Credit = credit. getText()

String Level = level. getText ()

String NoOfAssessments = No_Of_Assessments. getText ()

String Duration = duration. getText ()

Boolean isDuplicateCID = false

IF (CourseID. isEmpty () || CourseName. isEmpty () || Credit. isEmpty () ||
Level.isEmpty() || NoOfAssessments. isEmpty () || Duration. isEmpty ())

Display ERROR message dialog

Else

int Duration1 = Integer. parseInt (duration. getText ())

int NumberOfAssessments1 = Integer.parseInt (No_Of_Assessments.getText())

End else

End IF

FOR (course var: Array_List)

IF (var. getcourseID ()). equals (CourseID))

isDuplicateCID = true

break;

END IF

END FOR

IF (isDuplicateCID == false)

Academic = new AcademicCourse (CourseID, CourseName, Duration1, Credit,
Level, NoOfAssessments1)

Array_List.add (Academic)

Display suitable message dialog

Else

Display ERROR message dialog

End if

End try

Catch (Exception A)

Display ERROR message dialog

End catch

DO

Else IF (e. getSource () ==Register)

DO**TRY**

String courseID = Course_ID. getText ()

String courseLeader = Course_Leader. getText ()

String lecturerName = Lecturer_Name. getText ()

String S1 = date_1. getSelectedItem (). toString () + (month_1. getSelectedItem (). toString ()) + (year_1. getSelectedItem (). toString ())

String C1 = date_2. getSelectedItem (). toString () +(month_2. getSelectedItem (). toString ()) +(year_2. getSelectedItem (). toString ())

Boolean isalreadyreg=true

For (course var: Array_List)

IF (var. getcourseID (). equals (courseID))

IF (var instanceof AcademicCourse)

Isalreadyreg=true

AcademicCourse REGaca = (AcademicCourse) var

IF (REGaca.getisRegistered() == false)

Display Error message

Else

REGaca.register(courseLeader, lecturerName, S1, C1)

Display Suitable message

End else

End IF

Else

Display Error message

END ELSE

END IF

IF (isalreadyreg)

Display an ERROR message

END IF

END TRY

CATCH (Exception A)

Display suitable message

END CATCH

END IF

DO

IF (e. getSource () == Display)

IF (term1. isSelected ())

String data1;

String data

= "Course ID: "

+ Course_ID. getText () + "\n"

+ "Course name: "

+ Course_Name. GetText () + "\n"

String data2

= "Start Date: "

+ (String)date_1. getSelectedItem ()

+ "/" + (String)month_1. getSelectedItem ()

+ "/" + (String)year_1. getSelectedItem ()

+ "\n"

String data3

= "Completion Date: "

+ (String)date_2. getSelectedItem ()

+ "/" + (String)month_2. getSelectedItem ()

```
+ "/" + (String)year_2. getSelectedItem ()
```

```
+ "\n"
```

```
String data4
```

```
= "Course leader : "
```

```
+ Course_Leader.getText() + "\n"
```

```
+ "Lecturer name: "
```

```
+ Lecturer_Name.getText() + "\n"
```

```
String data5= "Credit: "
```

```
+ credit. getText () + "\n"
```

```
+ "Level: "
```

```
+ level. getText () + "\n"
```

```
String data6= "Duration"
```

```
+ (String)duration. getText () + "\n"
```

```
String data7= "Assessments"
```

```
+ (String)No_Of_Assessments.getText() + "\n"
```

```
tout1.setText(data +data5 + data6 + data7+ data2 + data3 + data4)
```

```
tout1.setEditable(false)
```

```
res1.setText
```

END IF

END IF

DO

IF (e. getSource () ==Clear)

String def = ""

```
Course_ID. setText(def)
```

```
Course_Name.setText(def)
```

```
Course_Leader.setText(def)
```

```
Lecturer_Name.setText(def)
```

```
level. setText(def)
```

```
credit. setText(def)
```

```
term1.setSelected(false)
```

date_1. setSelectedIndex (0)

month_1. setSelectedIndex (0)

year_1. setSelectedIndex (0)

date_2. setSelectedIndex (0)

month_2. setSelectedIndex (0)

year_2. setSelectedIndex (0)

duration. setText(def)

No_Of_Assessments.setText(def)

tout1.setText(def)

res1.setText(def)

END IF

DO

IF (e. getSource () == Add1)

DO

TRY

String CourseID1 = Course_ID. getText ()

String CourseName1 = Course_Name.getText()

String P1 = p1. getText ()

String D1 = d1. getText ()

Boolean isDuplicateCID = false

IF (CourseID1.isEmpty() || CourseName1.isEmpty() || P1. isEmpty () || D1. isEmpty ())

Display ERROR message dialog

Else

int Duration1 = Integer.parseInt(d1. getText ())

End else

End IF

FOR (course var: Array_List)

IF (var. getcourseID (). equals (CourseID))

isDuplicateCID = true

break

END IF

END FOR

IF (isDuplicateCID == false)

nonAcademic = new NonAcademicCourse (CourseID1, CourseName1,
Duration1, P1)

Array_List. add(nonAcademic)

Display suitable message dialog

Else

Display ERROR message dialog

End IF**End try****Catch** (Exception A)

Display ERROR message dialog

End catch**DO****Else IF** (e. getSource () ==Register1)**DO****TRY**

String CourseID1 = Course_ID1.getText()

String CourseLeader1 = Course_Leader1.getText()

String InstructorName = Instructor_Name.getText()

String S2 = (date_1. getSelectedItem ()). toString () + (month_1. getSelectedItem
(). toString ()) + (year_1. getSelectedItem (). toString ())String C2 = (date_2. getSelectedItem ()). toString () +(month_2. getSelectedItem
(). toString ()) +(year_2. getSelectedItem (). toString ())String E1 = (date. getSelectedItem ()). toString () + (month. getSelectedItem ().
toString ()) + (year. getSelectedItem (). toString ())

boolean foundCourseId=false

For (course var: Array_List)

IF (var. getcourseID (). equals (CourseID))

coursefound = true

IF (var instanceof NonAcademicCourse)

NonAcademicCourse REGnaca = (NonAcademicCourse) var

IF (REGnaca.getisRegistered() == true)

Display Error message

Else

REGnaca.register(CourseLeader1, InstructorName1, S2, C2, E1)

Display Suitable message

End else

End IF

Else

Display Error message

END ELSE

END IF

IF (!!coursefound)

Display an ERROR message

END IF

END TRY

CATCH (Exception A)

Display suitable message

END CATCH

END IF

DO

IF (e. getSource () == Display1)

IF (term. IsSelected ())

String data1;

String data

= "Course ID: "

+ Course_ID1.getText() + "\n"

+ "Course name: "

+ Course_Name1.getText() + "\n"

String data2

```
= "Start Date: "
```

```
+ (String)date_1. getSelectedItem ()
```

```
+ "/" + (String)month_1. getSelectedItem ()
```

```
+ "/" + (String)year_1. getSelectedItem ()
```

```
+ "\n"
```

```
String data3
```

```
= "Completion Date: "
```

```
+ (String)date_2. getSelectedItem ()
```

```
+ "/" + (String)month_2. getSelectedItem ()
```

```
+ "/" + (String)year_2. getSelectedItem ()
```

```
+ "\n"
```

```
String data4
```

```
= "Course leader: "
```

```
+ Course_Leader1.getText() + "\n"
```

```
String data5= "Duration"
```

```
+ (String)d1. getText () + "\n"
```

```
String data6= "Prerequisites"
```

```
+ (String)p1. getText () + "\n"
```

```
String data7
```

```
= "Exam Date: "
```

```
+ (String)date. getSelectedItem ()
```

```
+ "/" + (String)month. getSelectedItem ()
```

```
+ "/" + (String)year. getSelectedItem ()
```

```
+ "\n"
```

```
String data8
```

```
= "Instructor Name: "
```

```
+ Instructor_Name.getText() + "\n"
```

```
tout. SetText (data + data5 + data6 + data2 + data3 + data7 + data8 + data4 );
```

tout. set Editable(false)

res. setText

END IF

END IF

DO

Else IF (e. getSource () ==Remove)

DO

TRY

String courseID = Course_ID. getText ()

IF (courseID. isEmpty ())

Display Error message

ELSE

boolean isDuplicateNaCID = false

For (course var: Array_List)

IF (var. getCourseID (). equals (CourseID))

isDuplicateNaCID = true;

IF (var instanceof NonAcademicCourse)

nonAcademic = (NonAcademicCourse) var

IF (nonAcademic.getisRemoved() ==true)

Display Error message

Else

nonacademic. Remove ()

Display Suitable message

break

End Else

End IF

Else

Display Error message

break

END ELSE

END IF

IF (! isDuplicateNaCID)

Display an ERROR message

END IF

END TRY

CATCH (NumberFormatException E)

Display Error message

END CATCH

END IF

DO

IF (e. getSource () ==Clear1)

String def = ""

Course_ID1.setText(def)

Course_Name1.setText(def)

Course_Leader1.setText(def)

term. setSelected(false)

date. SetSelectedIndex (0)

month. SetSelectedIndex (0)

year. SetSelectedIndex (0)

Instructor_Name. SetText(def)

date_1. setSelectedIndex (0)

month_1. setSelectedIndex (0)

year_1. setSelectedIndex (0)

date_2. setSelectedIndex (0)

month_2. setSelectedIndex (0)

year_2. setSelectedIndex (0)

d1. setText(def)

p1. setText(def)

tout. SetText(def)

res. setText(def)

END IF

End IF

DEFINE method getCourseID () as String type

DO

Return Course_ID

ENDDO

DEFINE method getCourseName () as String type

DO

Return Course_Name

ENDDO

DEFINE method getDuration () as String type

DO

Return duration

ENDDO

DEFINE method getCourseLeader () as String type

DO

Return Course_Leader

ENDDO

DEFINE method getLecturerName () as String type

DO

Return Lecturer_Name

ENDDO

DEFINE method getLevel () as String type

DO

Return level

ENDDO

DEFINE method getCredit () as String type

DO

Return credit

ENDDO

DEFINE method getNoOfAssessments () as String type

DO

Return No_Of_Assessments

ENDDO

DEFINE method getS1 () as String type

DO

Return date_1. getSelectedItem (). toString () + (month_1. getSelectedItem ().
toString ()) + (year_1. getSelectedItem (). toString ());

ENDDO

DEFINE method getC1 () as String type

DO

Return (date_2. getSelectedItem (). toString () + (month_2. getSelectedItem ().
toString ()) + (year_2. getSelectedItem (). toString ());

ENDDO

DEFINE method getlCourseID1 () as String type

DO

Return Course_ID1

ENDDO

DEFINE method getCourseName1 () as String type

DO

Return Course_Name1

ENDDO

DEFINE method getCourseLeader1 () as String type

DO

Return Course_Leader1

ENDDO

DEFINE method getD1 () as String type

DO

Return D1

ENDDO

DEFINE method getIS2 () as String type

DO

Return S2

ENDDO

DEFINE method getIC2 () as String type

DO

Return C2

ENDDO

DEFINE method getIE1 () as String type

DO

Return E1

ENDDO

DEFINE method getInstructorName () as String type

DO

Return Instructor_Name

ENDDO

DEFINE method getIP1 () as String type

DO

Return p1

ENDDO

DO

public static void main (String args [])

ING_College Main=new ING_College ();

Main. ING_College ();

END DO**METHOD DESCRIPTION**

- Get () Method

Method	Method Description
getCourseID ()	This method takes the value from Course_ID-text field for the course ID and returns the value.
getCourseName ()	This method takes the value from Course_Name-text field for the Course Name and returns the value.
getDuration ()	This method takes the value from duration-text field for the Duration and returns the value
getCourseLeader ()	This method takes the value from Course_Leader-text field for the course Leader and returns the value
getLecturerName ()	This method takes the value from Lecturer_Name-text field for the Lecturer Name and returns the value
getLevel ()	This method takes the value from level-text field for the Level and returns the value
getCredit ()	This method takes the value from credit-text field for the Credits and returns the value
getNoOfAssessments ()	This method takes the value from No_Of_Assessments-text field for NoOfAssessments and returns the value

getS1()	This method takes the value from data_1, month_1, year_1-combo box for the Starting Date and returns the value
getC1()	This method takes the value from data_2, month_2, year_2-combo box for the Completion Date and returns the value
getD1()	This method takes the value from duration-text field for the Duration and returns the value
getCourseID1()	This method takes the value from Course_ID1-text field for the course ID and returns the value
getCourseName1()	This method takes the value from Course_Name1-text field for the Course Name and returns the value
getCourseLeader1()	This method takes the value from Course_Leader1-text field for the CourseLeader and returns the value
getD1()	This method takes the value from d1-text field for the Duration and returns the value
getS2()	This method takes the value from date_1, month_1. Year_1-Combo box for the Starting Date and returns the value
getC2()	This method takes the value from date_2, month_2, year_2-Combo box for the Completion Date and returns the value

getE1()	This method takes the value from date, month, year-Combo box for the Exam Date and returns the value
getInstructorName ()	This method takes the value from Instructor_Name-text field for the Instructor Name and returns the value
getP1()	This method takes the value from p1-text field for Prerequisites and returns the value

Table 3:Table for describing getters method

- **ING_College () Method:**

- This is the method which forms the structure of the entire GUI as it helps to initialize the different components such as JLabel, JTextField, JButton, JTextArea, JComboBox, JCheckBox, etc. This method also helps to form the skeleton of the entire GUI by allowing us to set various panels, change the location, enhance the body by using various texts, fonts, colors and background. This method also helps to make the frame and the components visible and resizable by adjusting the layout of the whole frame.

- **actionPerformed (Action Event e) Method:**

- actionPerformed (Action Event e) Method is the method which is responsible for the working of the buttons and executing the entire commands given to the code. As the term describes, this process is responsible for all the overall action that is performed in the form. This method is used to set up error message and the correct message along with each button pressed systematically. This method initializes each button one after the other and displays correct message if the

button is working. If all the buttons are working then the correct entries are entered and is displayed in the text area.

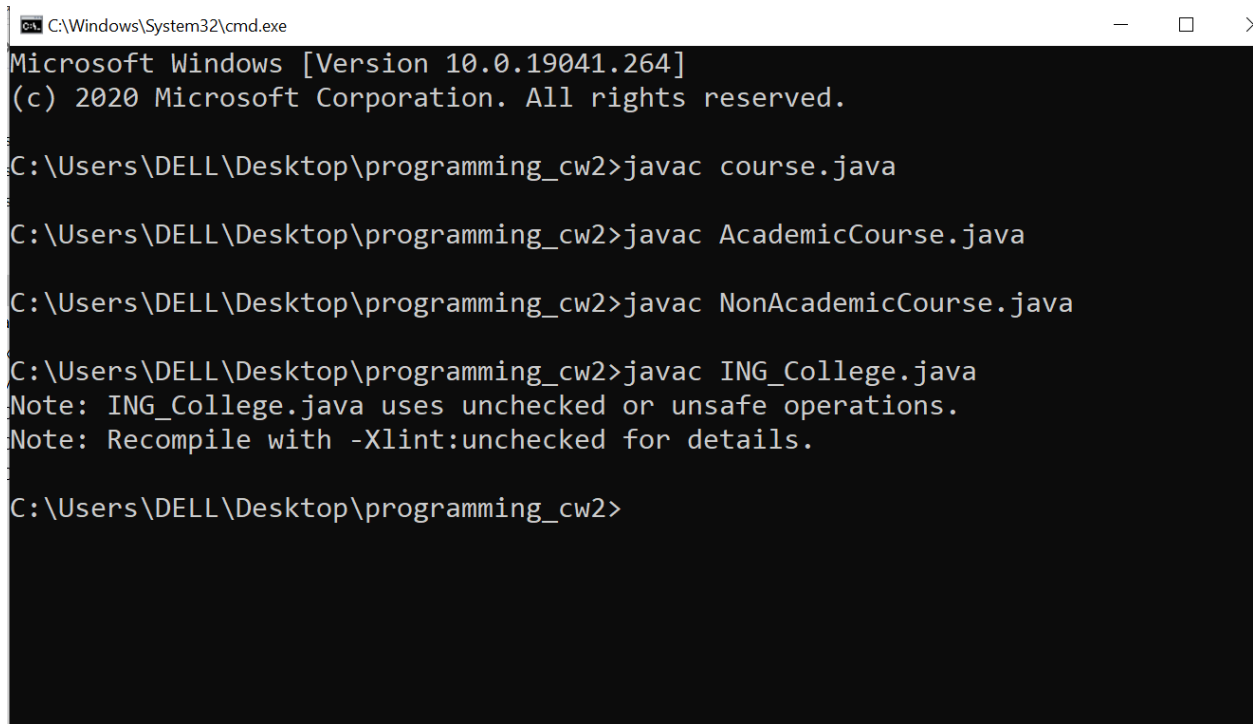
- Main (String args []) Method:
 - This method has been called as it is the main method of the class and it calls the method ING_College and creates a new object for systematic running of all the entered data and the skeleton of the form.

TESTS: -

- TEST 1:

TEST Number	1
Objective	To compile and run the program in command prompt.
Action	(course, AcademicCourse, NonAcademicCourse, ING_College) classes were compiled in the command prompt and the ING_College class file was run
Expected Result	To open the GUI made in ING_College.
Actual Result	The program successfully ran using command prompt and the GUI was opened

Table 4: TEST 1 For compiling and running program in CMD prompt



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19041.264]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\DELL\Desktop\programming_cw2>javac course.java

C:\Users\DELL\Desktop\programming_cw2>javac AcademicCourse.java

C:\Users\DELL\Desktop\programming_cw2>javac NonAcademicCourse.java

C:\Users\DELL\Desktop\programming_cw2>javac ING_College.java
Note: ING_College.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

C:\Users\DELL\Desktop\programming_cw2>
```

Figure 1: Running the codes of the classes using CMD prompt.



Course Registration

Academic Course

Course ID	<input type="text"/>	Course Duration	<input type="text"/>
Course Name	<input type="text"/>	Assessments	<input type="text"/>
Level	<input type="text"/>	Credit	<input type="text"/>
<input type="button" value="Add"/>			
Lecturer Name	<input type="text"/>	Course Leader	<input type="text"/>
Start Date	<input type="text" value="1"/> <input type="text" value="Jan"/> <input type="text" value="2020"/>	Completion Date	<input type="text" value="1"/> <input type="text" value="Jan"/> <input type="text" value="2020"/>
<input type="button" value="Register"/>			
<input type="checkbox"/> Enable Text Area			
<input type="button" value="Display"/>		<input type="button" value="Clear"/>	
<input type="button" value="Add Non-Academic Course"/>			

Figure 2: Evidence that the code for Academic Course runs

Course Registration

Non Academic Course

Course ID Course Duration

Course Name Prerequisites

Instructor Name Course Leader

Start Date 1 Jan 2020 Completion Date 1 Jan 2020

Exam Date 1 Jan 2020

☐ Enable Text Area

Figure 3: Evidence that the code for Non-Academic Course runs

- TEST 2:

TEST Number 2	
Objectives	To ADD and Register Academic and NonAcademic Courses and to Remove the NonAcademic Course
Actions	Here all the fields were inserted and the buttons were pressed respectively. Firstly, ADD button adds the data REGISTER button registers the courses and then REMOVE button removes the NonAcademic course.
Expected Result	To add and Register both Academic and NonAcademic course and to remove Nonacademic Course
Actual Result	The program was successfully added, registered and removed after the buttons were pressed.

Table 5: TEST 2 for checking ADD , REGISTER and REMOVE buttons.

 Course Registration

Academic Course

Course ID	<input type="text" value="13"/>	Course Duration	<input type="text" value="3"/>
Course Name	<input type="text" value="ENGLISH"/>	Assessments	<input type="text" value="5"/>
Level	<input type="text" value="4"/>	Credit	<input type="text" value="4"/>

Figure 4: Data for adding academic course

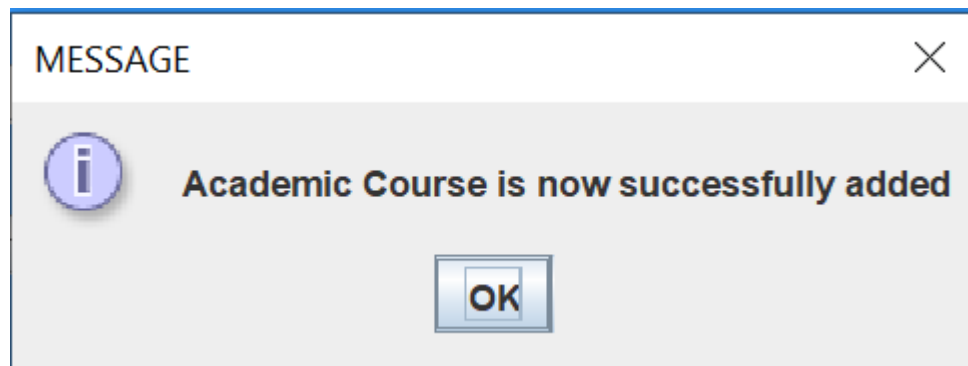


Figure 5: Dialogue Box for adding academic course

 Course Registration

Academic Course

Course ID	<input type="text" value="13"/>	Course Duration	<input type="text" value="3"/>
Course Name	<input type="text" value="ENGLISH"/>	Assessments	<input type="text" value="5"/>
Level	<input type="text" value="4"/>	Credit	<input type="text" value="4"/>

Lecturer Name	<input type="text" value="HARI BAHADUR"/>	Course Leader	<input type="text" value="RAJESH HAMAL"/>
Start Date	<input type="text" value="1"/> <input type="text" value="Jan"/> <input type="text" value="2020"/>	Completion Date	<input type="text" value="1"/> <input type="text" value="May"/> <input type="text" value="2023"/>

Figure 6: Adding data for registering academic course

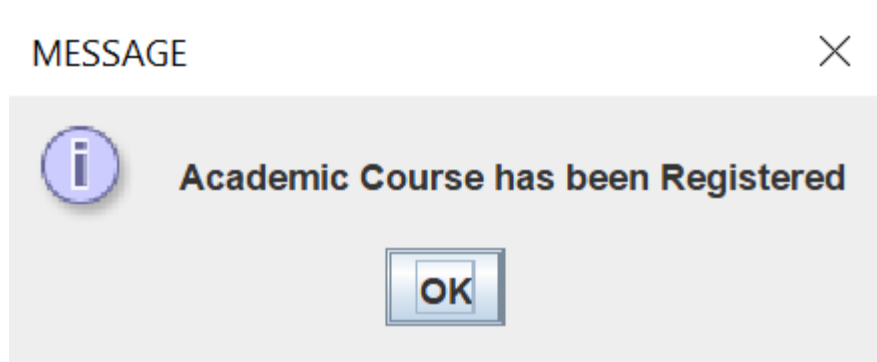


Figure 7: Dialogue box for adding academic course

Course Registration

Non Academic Course

Course ID	<input type="text" value="18"/>	Course Duration	<input type="text" value="6"/>
Course Name	<input type="text" value="LOGIC"/>	Prerequisites	<input type="text" value="join now"/>

Figure 8: Data for adding nonacademic course

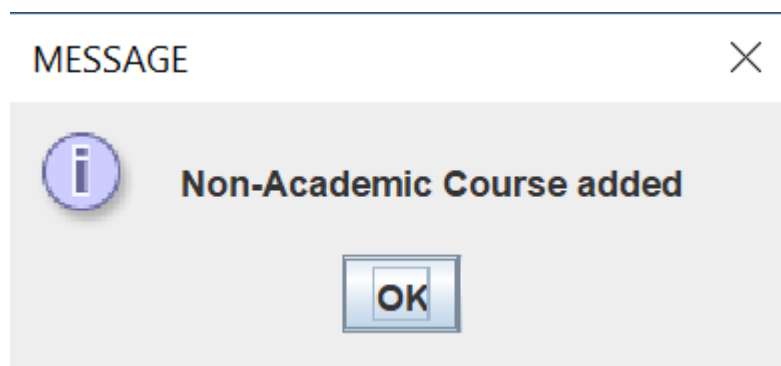


Figure 9: Dialogue Box for added non academic course

Course Registration

Non Academic Course

Course ID	<input type="text" value="18"/>	Course Duration	<input type="text" value="6"/>
Course Name	<input type="text" value="LOGIC"/>	Prerequisites	<input type="text" value="join now"/>
<input type="button" value="Add"/>			
Instructor Name	<input type="text" value="RAM"/>	Course Leader	<input type="text" value="KRISHNA"/>
Start Date	<input type="text" value="1"/> <input type="text" value="Jan"/> <input type="text" value="2020"/>	Completion Date	<input type="text" value="1"/> <input type="text" value="feb"/> <input type="text" value="2023"/>
Exam Date		<input type="text" value="1"/> <input type="text" value="May"/> <input type="text" value="2021"/>	
<input type="button" value="Register"/>			

Figure 10: Data for registering non academic course class

MESSAGE ×


 **Non Academic Course has been Registered Successfully**

Figure 11: Dialogue Box for registered non academic course

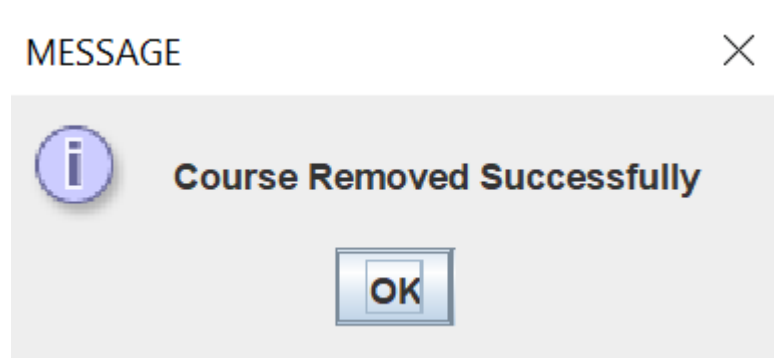


Figure 12: Dialogue Box for removing non academic course

- TEST 3:

TEST Number	3
Objective	To test by Adding duplicate Course ID and Registering the courses that is already registered and removing the NonAcademic course which is already removed
Actions	<ul style="list-style-type: none"> • In each text fields unsuitable Course were assigned and the corresponding buttons were clicked so that the dialog box appears • Course ID was repeated to get the suitable message from the dialog box. • Academic and NonAcademic Course was registered by clicking the corresponding button and NonAcademic Course was removed as well
Expected Results	To show appropriate dialog box in each case.
Actual Result	It showed the dialog box with the required message to the user so the person can reenter the course form.

Table 6: TEST 3 for checking repeated operations

The screenshot shows a web application window titled "Course Registration". Inside, there's a section titled "Academic Course". The form contains several input fields: "Course ID" (13), "Course Duration" (6), "Course Name" (ENGLISH), "Assessments" (5), "Level" (4), "Credit" (4), "Lecturer Name" (HARI BAHADUR), "Course Leader" (RAJESH HAMAL), "Start Date" (1 Jan 2020), and "Completion Date" (1 Jan 2020). There are buttons for "Add", "Register", "Display", "Clear", and "Add Non-Academic Course". A checkbox labeled "Enable Text Area" is checked. A modal dialog box titled "MESSAGE" is open, displaying an information icon and the text "Academic Course is now successfully added" with an "OK" button.

Figure 13: Adding academic course and getting the dialogue

The screenshot shows the same "Academic Course" registration form as Figure 13. However, the "Add" button is disabled. A modal dialog box titled "ERROR!!" is open, displaying a red error icon and the text "Course ID has been repeated" with an "OK" button. The form fields and other UI elements remain the same as in the previous figure.

Figure 14: Repeating the academic course ID and getting error box

Course Registration

Academic Course

Course ID: 13 Course Duration: 6

Course Name: ENGLISH Assessments: 5

Level: 4 Credit: 4

Lecturer Name: HARI BAHADUR

Start Date: 1 Jan 2020

Register

Enable Text Area

Display Clear

Add Non-Academic Course

MESSAGE

Academic Course has been Registered

OK

Figure 15: Dialogue box for registering academic course

Course Registration

Non Academic Course

Course ID: 18 Course Duration: 6

Course Name: LOGIC Prerequisites: on now

Instructor Name: Course Leader:

Start Date: 1 Jan 2020 Completion Date: 1 Jan 2020

Exam Date: 1 Jan 2020

Register

Enable Text Area

Display Clear

Remove

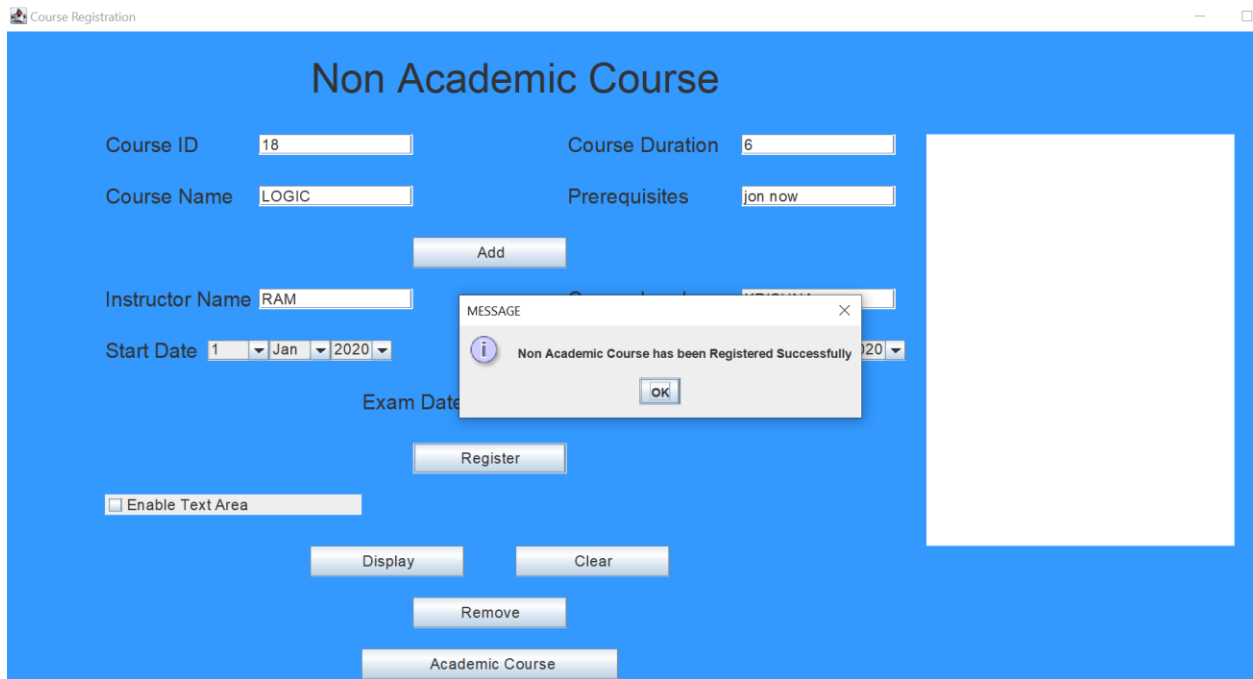
Academic Course

MESSAGE

Non-Academic Course added

OK

Figure 16: Adding non academic course and getting appropriate dialogue box

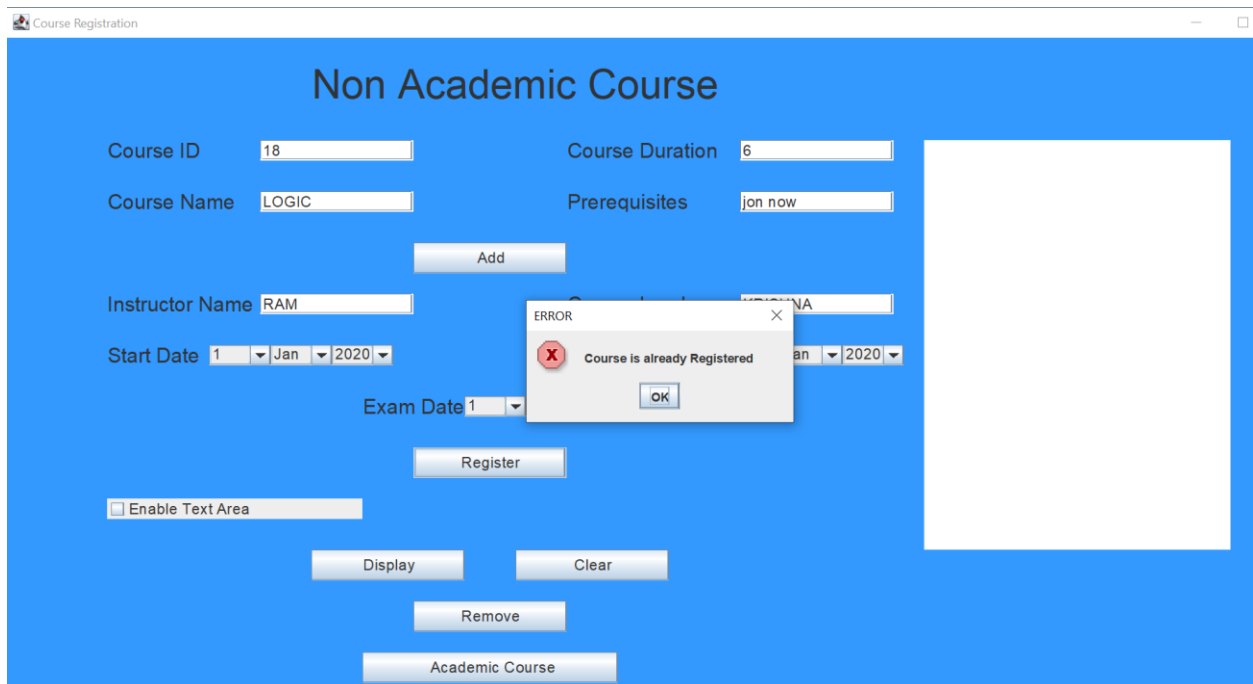


The screenshot shows a web application window titled "Course Registration". The main heading is "Non Academic Course". The form contains the following fields and controls:

- Course ID: 18
- Course Duration: 6
- Course Name: LOGIC
- Prerequisites: jon now
- Instructor Name: RAM
- Start Date: 1 Jan 2020
- Exam Date: (empty)
- Buttons: Add, Register, Display, Clear, Remove, Academic Course
- Checkbox: Enable Text Area

A "MESSAGE" dialog box is displayed in the center, stating: "Non Academic Course has been Registered Successfully". The dialog has an "OK" button.

Figure 17: registering the nonacademic course class successfully



The screenshot shows the same "Non Academic Course" registration form as Figure 17. The fields and controls are identical. However, an "ERROR" dialog box is displayed in the center, stating: "Course is already Registered". The dialog has a red "X" icon and an "OK" button.

Figure 18: trying to reregister the same curse and getting error box

 Course Registration

Non Academic Course

Course ID	<input type="text" value="18"/>	Course Duration	<input type="text" value="6"/>
Course Name	<input type="text" value="LOGIC"/>	Prerequisites	<input type="text" value="jon now"/>

Instructor Name	<input type="text" value="RAM"/>	<input type="text" value="KUSUMA"/>
Start Date	<input type="text" value="1"/> <input type="text" value="Jan"/> <input type="text" value="2020"/>	<input type="text" value="an"/> <input type="text" value="2020"/>

Exam Date

☒ Enable Text Area

Error


 Course is already removed

Figure 19: Dialogue box after removing the nonacademic course class

- TEST 4:

TEST Number	4
Objectives	To DISPLAY the entered data in the text area.
Actions	Here all the fields were inserted and the buttons were pressed respectively. Then, the display button was pressed.
Expected Result	To display all the entered data in the text area.
Actual Result	All data were displayed correctly.

Table 7: TEST 4 for displaying

The screenshot shows a web application window titled 'Course Registration'. Inside, there's a section titled 'Academic Course'. The form contains several input fields: Course ID (13), Course Name (ENGLISH), Level (4), Course Duration (6), Assessments (5), Credit (4), Lecturer Name (HARI BAHADUR), Course Leader (RAJESH HAMAL), Start Date (1 Jan 2020), and Completion Date (1 Jan 2020). There are buttons for 'Add', 'Register', 'Display', 'Clear', and 'Add Non-Academic Course'. A checkbox labeled 'Enable Text Area' is checked. On the right, a text area displays the entered data: Course ID : 13, Course name : ENGLISH, Credit: 4, Level: 4, Duration: 6, Assessments: 5, Start Date : 1/Jan/2020, Completion Date : 1/Jan/2020, Course leader : RAJESH HAMAL, and Lecturer name : HARI BAHADUR. Below the text area, it says 'Check if the displayed data is correct'.

Figure 20: Displaying all the entered data of Academic Course

Non Academic Course

Course ID: 18 Course Duration: 6

Course Name: LOGIC Prerequisites: jon now

Instructor Name: RAM Course Leader: KRISHNA

Start Date: 1 Jan 2020 Completion Date: 1 Jan 2020

Exam Date: 1 Jan 2020

Register

☒ Enable Text Area

Display Clear

Remove

Academic Course

Course ID : 18
Course name : LOGIC
Duration 6
Prerequisites jon now
Start Date : 1/Jan/2020
Completion Date : 1/Jan/2020
Exam Date : 1/Jan/2020
Instructor Name : RAM
Course leader : KRISHNA

Check if the displayed data is correct

Figure 21: Displaying all the entered data of Non-Academic Course

ERROR DETECTION

- SYNTAX ERROR: -
 - Syntax error occurs when we forget to add the appropriate syntax in our code.
 - In the code the error occurred when “;” was not added at the end of a code.

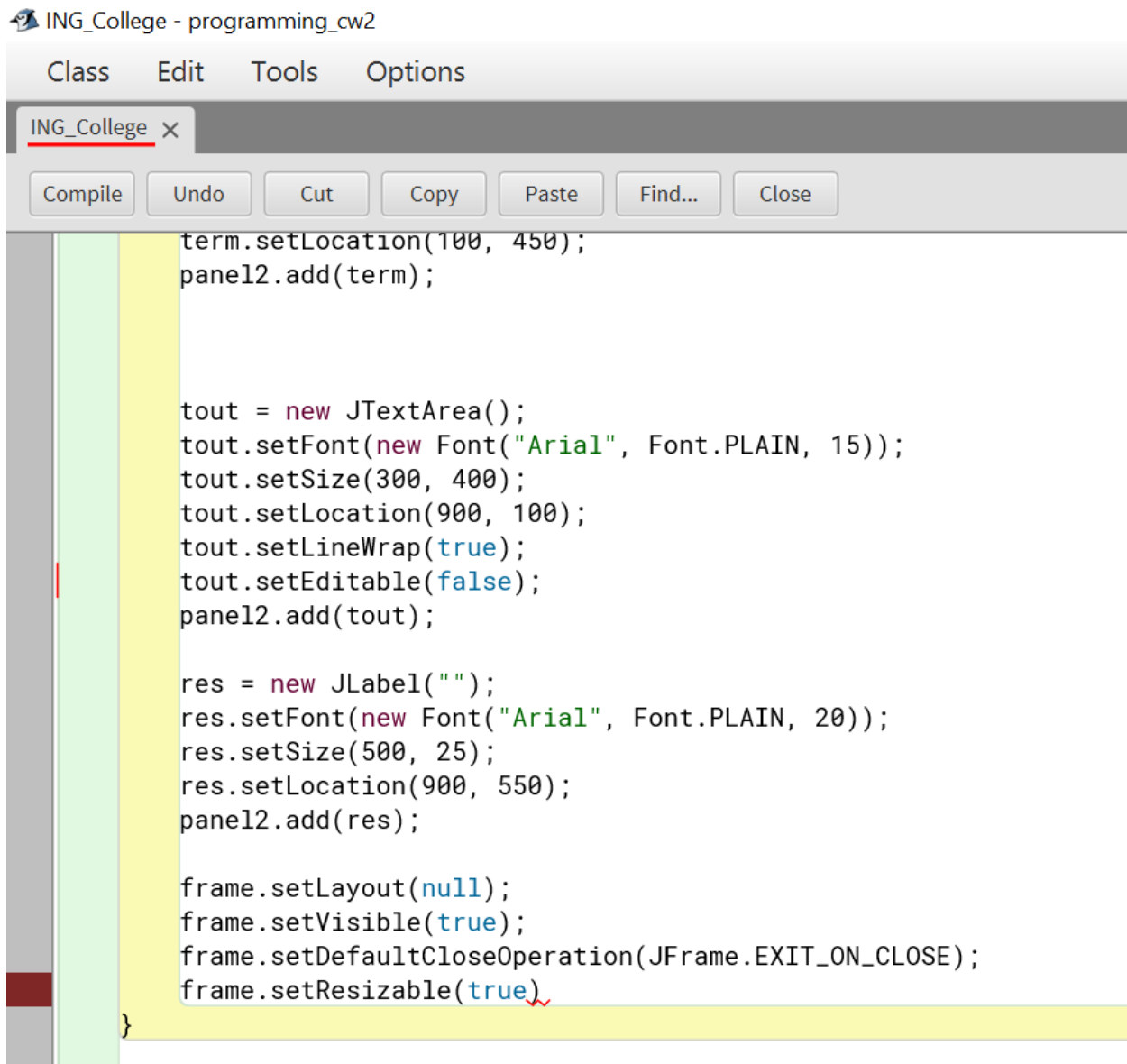
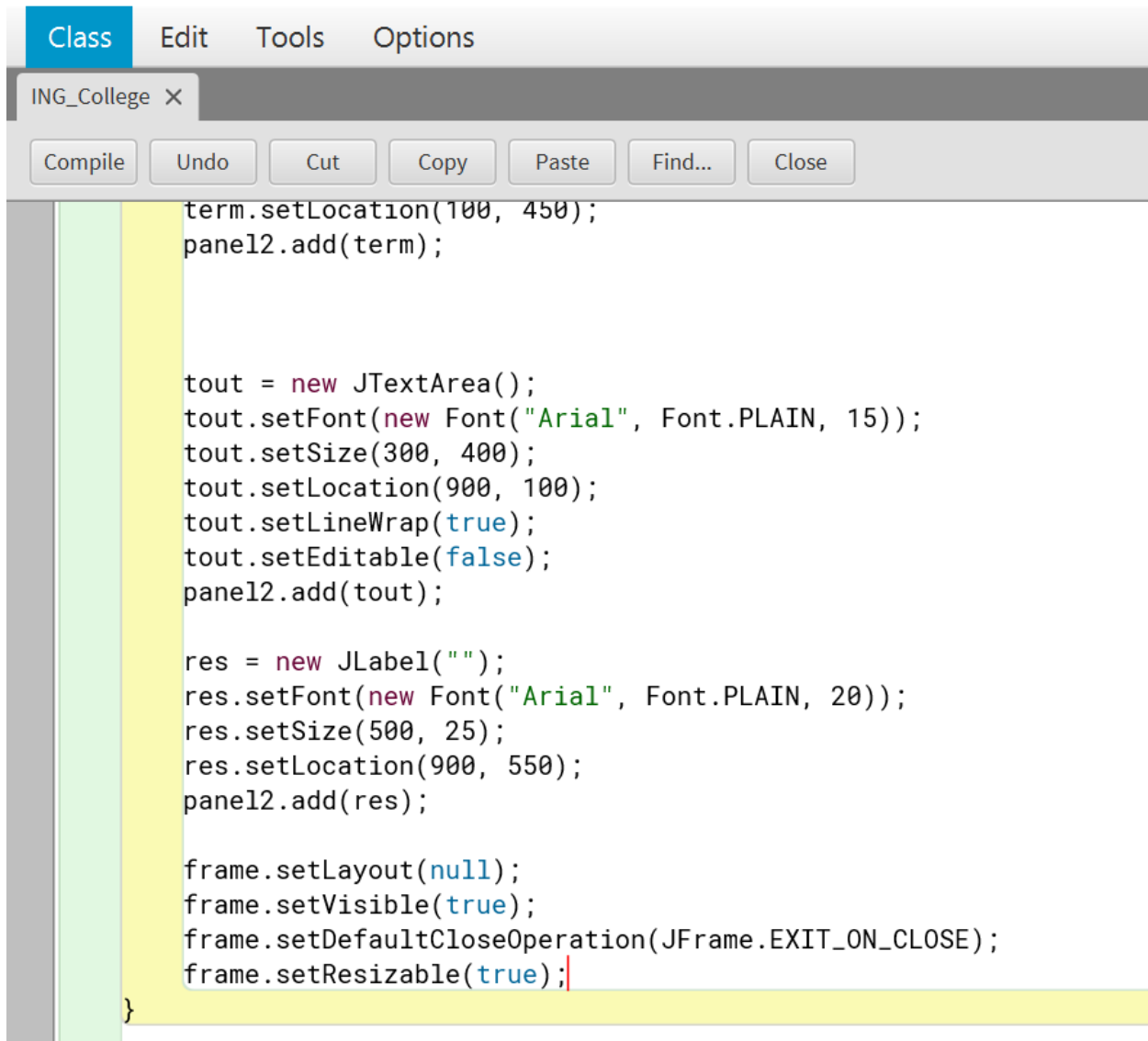


Figure 22: Syntax Error

ING_College - programming_cw2

*Figure 23: Correction of Syntax Error*

- IMPORT ERROR:
 - This error occurs when certain packages are not imported but the package data are stated in the code.


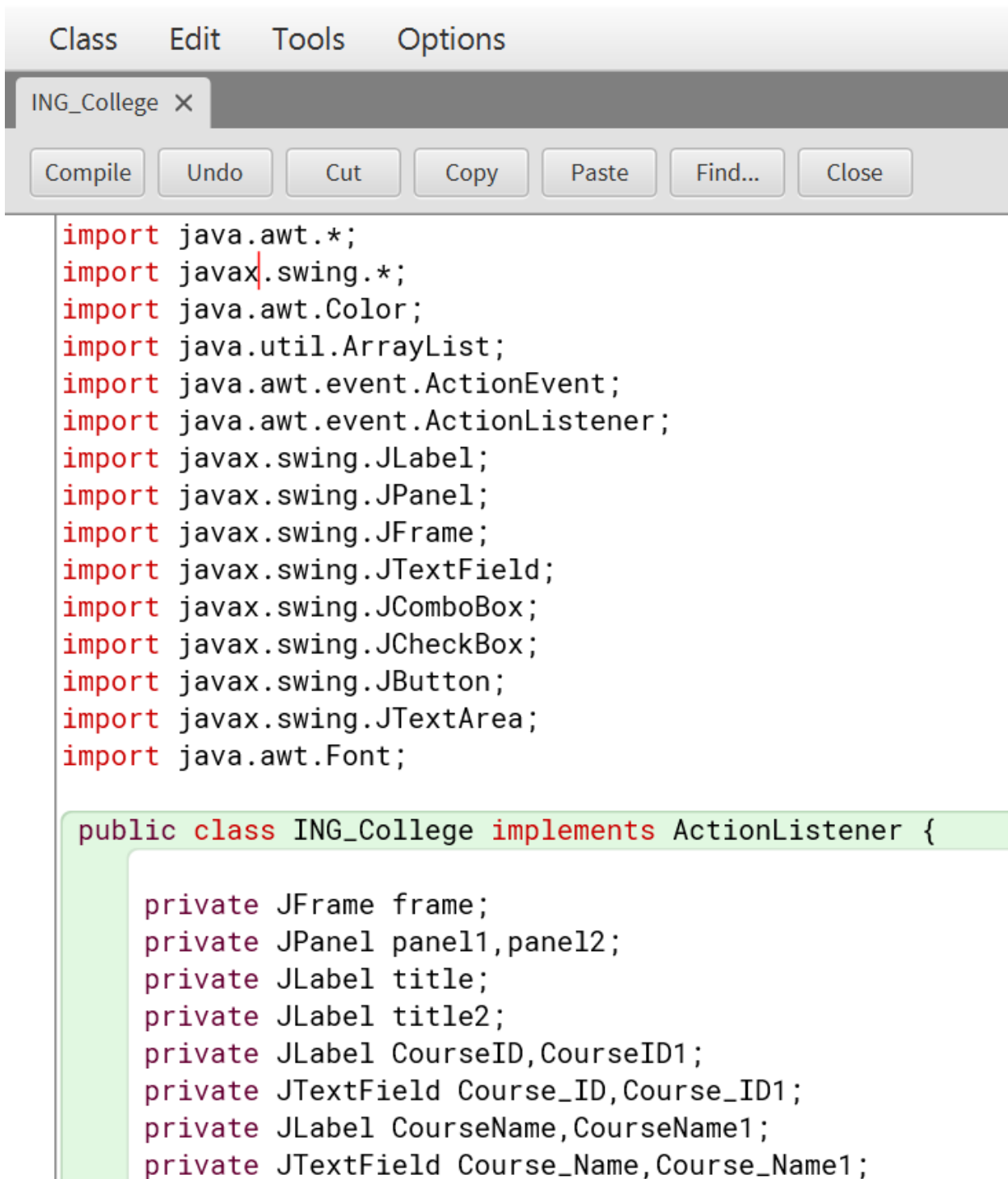
ING_College - programming_cw2

```
import java.awt.Color;
import java.util.ArrayList;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ING_College implements ActionListener {

    private JFrame frame;
    private JPanel panel1, panel2;
    private JLabel title;
    private JLabel title2;
    private JLabel CourseID, CourseID1;
    private JTextField Course_ID, Course_ID1;
    private JLabel CourseName, CourseName1;
    private JTextField Course_Name, Course_Name1;
    private JLabel CourseLeader, CourseLeader1;
    private JTextField Course_Leader, Course_Leader1;
    private JLabel LecturerName;
```

Figure 24: Import error

 ING_College - programming_cw2

```
import java.awt.*;
import javax.swing.*;
import java.awt.Color;
import java.util.ArrayList;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JFrame;
import javax.swing.JTextField;
import javax.swing.JComboBox;
import javax.swing.JCheckBox;
import javax.swing.JButton;
import javax.swing.JTextArea;
import java.awt.Font;

public class ING_College implements ActionListener {

    private JFrame frame;
    private JPanel panel1, panel2;
    private JLabel title;
    private JLabel title2;
    private JLabel CourseID, CourseID1;
    private JTextField Course_ID, Course_ID1;
    private JLabel CourseName, CourseName1;
    private JTextField Course_Name, Course_Name1;
```

Figure 25: Correction of import error

- SEMANTIC ERROR:

- This type of error occurs when there is a mistake in the constructor of certain codes.
- Here, Semantic error occurs when + is used instead of = in the code.

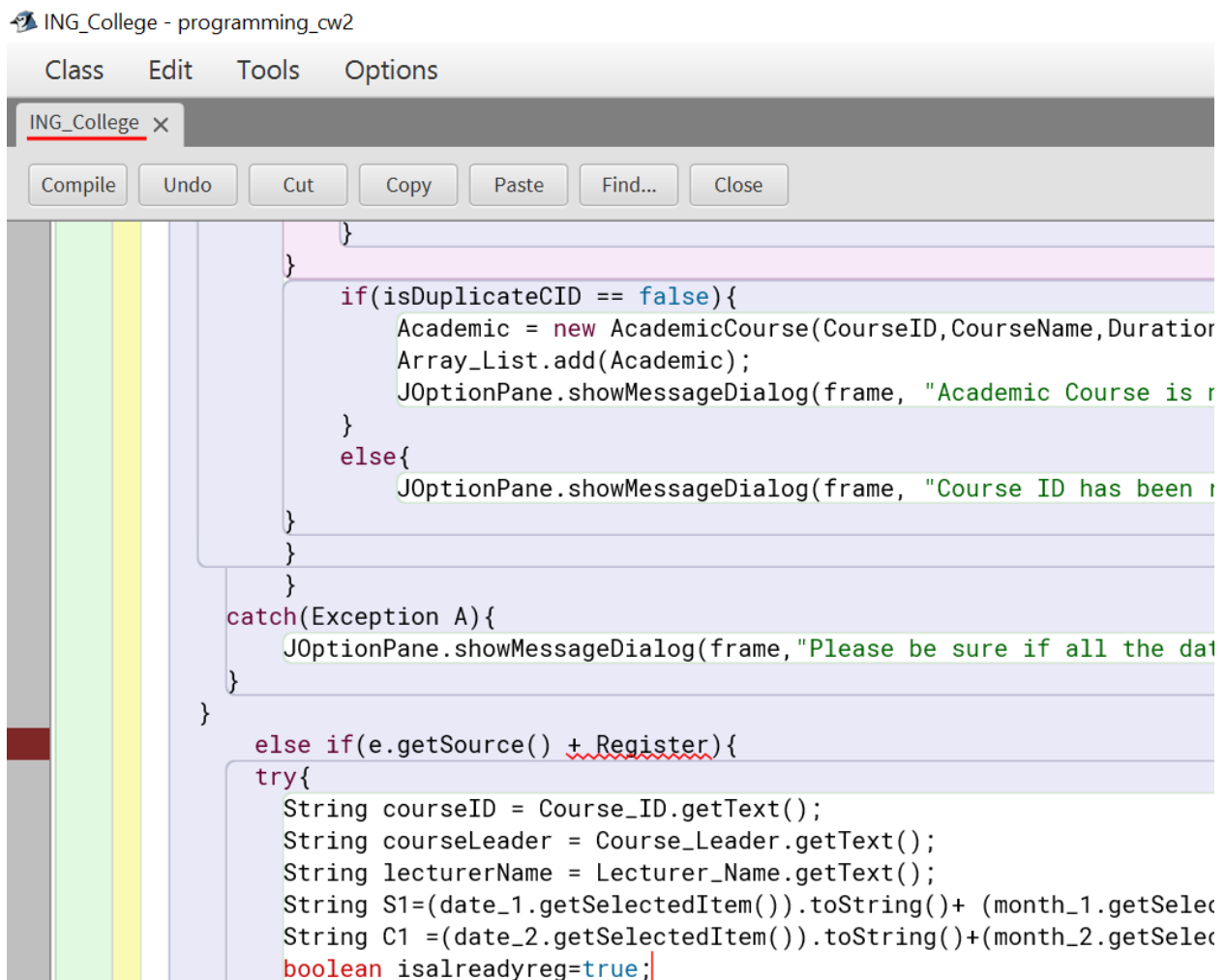
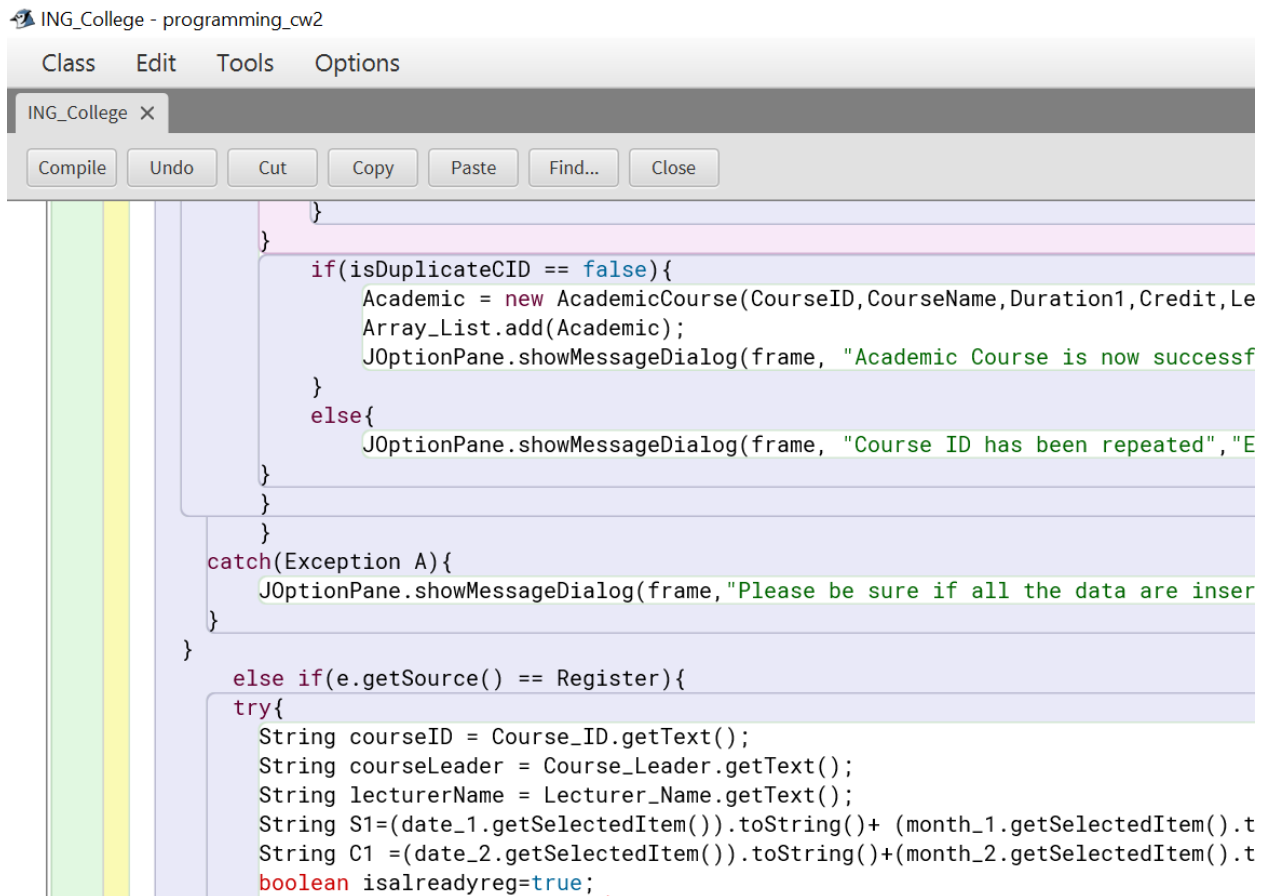


Figure 26: SEMANTIC ERROR



```

}
}
if(isDuplicateCID == false){
    Academic = new AcademicCourse(CourseID, CourseName, Duration1, Credit, Le
    Array_List.add(Academic);
    JOptionPane.showMessageDialog(frame, "Academic Course is now successf
}
else{
    JOptionPane.showMessageDialog(frame, "Course ID has been repeated", "E
}
}
}
catch(Exception A){
    JOptionPane.showMessageDialog(frame, "Please be sure if all the data are inser
}
}
else if(e.getSource() == Register){
    try{
        String courseID = Course_ID.getText();
        String courseLeader = Course_Leader.getText();
        String lecturerName = Lecturer_Name.getText();
        String S1=(date_1.getSelectedItem()).toString()+ (month_1.getSelectedItem()).t
        String C1 =(date_2.getSelectedItem()).toString()+(month_2.getSelectedItem()).t
        boolean isAlreadyreg=true;
    }
}

```

Figure 27: Correction of SEMANTIC ERROR

CONCLUSION:

- This Coursework in general has taught me a lot about the various aspects of coding. In this assignment the main focus is in the ING_College class where the main GUI is located. The GUI functions through various buttons and different permission that is granted after pressing the buttons. At this stage of programming we have faced various different problems and have been introduced to various small aspects that can make or break a program that has to be created. Since the start of our module, we have started from arranging basic strings, arrays and by carrying out small operations. But at this stage, we have worked with GUI, functionality of buttons, displaying messages through the help of dialogue box and also have added event handling qualities to our work which makes our work responsive. I have learned various different things through this coursework such as adding the action listener which makes the buttons work, I have learned about making the GUI more attractive and how to add various different new things into the GUI. Creation of array list to connect the course class, academic course class and the nonacademic course class to the main ING_College class.

The overall analysis about the difficulties that I faced while coding helps me conclude that with proper knowledge and research, all the problems can be solved easily without errors. I had difficulties that were acting as a set back since the start mainly during the functionality of the buttons and while adjusting the location of the entire GUI components. Many of my problems were solved by my friends and teachers along with my own personal studies and researches as well.

In conclusion, I can state that this assignment has helped me learn about various new components that are important while coding and have also taught me to face the difficulties with less worry and look forward to finding a solution to the

problems that have been encountered. I have tried my best and have made my work presentable for the better understanding about what I have done.

APPENDIX:

- COURSE CLASS:

```
-  
- public class course  
- {  
-     //Attributes of course class  
-     private String courseID;  
-     private String courseName;  
-     private String courseLeader;  
-     private int duration;  
-     /*  
-     * defining constructor of course class  
-     */  
-     public course(String courseID,String courseName, int duration)  
-     {  
-         this. courseID = courseID;  
-         this. courseName=courseName;  
-         this. duration = duration;  
-         this.courseLeader = " ";  
-     }  
-     //Accessor method--Getters method  
-     public String getcourseID()  
-     {  
-         return this.courseID;  
-     }  
-  
-     public String getcourseName()
```

```
- {  
-     return this.courseName;  
- }  
-  
- public int getduration()  
- {  
-     return this.duration;  
- }  
-  
- public String getcourseLeader()  
- {  
-     return this.courseLeader;  
- }  
- //setters method  
- public void setcourseLeader(String courseLeader)  
- {  
-     this.courseLeader = courseLeader;  
- }  
-  
- /*defining method to display the details of the course class  
-  * courseID , course name, duration, courseleader.  
-  */  
- public void Display()  
- {  
-     System.out.println("The ID is"+getcourseID());  
-     System.out.println("The name of the course is"+getcourseName());  
-     System.out.println("The duration of course is"+getduration());  
-     if(courseLeader!=" ")  
-     {  
-         System.out.println("The leader of the course is"+getcourseLeader());  
-     }  
- }  
- }
```

- ACADEMIC COURSE CLASS:

```
-  
- public class AcademicCourse extends course  
- {  
-     //Attributes of class:AcademicCourse  
-     private String LecturerName;  
-     private String Level;  
-     private String Credit;  
-     private String StartingDate;  
-     private String CompletionDate;  
-     private int NumberOfAssesments;  
-     private boolean isRegistered;  
-  
-     /*  
-     * Constructor of Academic Course Class which has six parameters.  
-     */  
-     AcademicCourse(String CourseID, String CourseName,  
-     int duration,String level,String Credit,int NumberOfAssesments)  
-     {  
-         super(CourseID,CourseName,duration);  
-         this.Level=level;  
-         this.Credit=Credit;  
-         this.NumberOfAssesments=NumberOfAssesments;  
-         this. LecturerName="";  
-         this. StartingDate="";  
-         this.CompletionDate="";  
-         this.isRegistered= true;  
-     }  
-  
-     //accessor method--getters method  
-  
-     public String getlecturerName()
```

```
- {  
-     return this.LecturerName;  
- }  
-  
- public String getlevel()  
- {  
-     return this.Level;  
- }  
-  
- public String getcredit()  
- {  
-     return this.Credit;  
- }  
-  
- public String getstartingdate()  
- {  
-     return this.StartingDate;  
- }  
-  
- public String getcompletiondate()  
- {  
-     return this.CompletionDate;  
- }  
-  
- public int getNumberOfAssessment()  
- {  
-     return this.NumberOfAssesments;  
- }  
-  
- public boolean getisRegistered()  
- {  
-     return this.isRegistered;  
- }  
- //setters method is defined
```

```
- public void setLecturerName(String lecturer)
- {
-     this.LecturerName = LecturerName;
- }
-
- public void setNoofassesments(int number)
- {
-     this.NumberOfAssesments=NumberOfAssesments;
- }
-
- /*deifining method to register
-  * lecturername, startingdate, completiondate
-  */
- public void register(String courseLeader,String lecturer,
- String start,String completion)
- {
-     if(this.isRegistered ==false)
-     {
-         System.out.println("Instructor Name is:" +this.LecturerName);
-         System.out.println("Starting date:" + this.StartingDate);
-         System.out.println("Completion date:"+this.CompletionDate);
-     }
-     else if(this.isRegistered==true)
-     {
-         super. setcourseLeader(courseLeader);
-
-         this.LecturerName = lecturer;
-         this.StartingDate= start;
-         this.CompletionDate = completion;
-         this.isRegistered = true;
-     }
- }
-
- /*defining method to display the details of academic course class
```

```

-      * lecturer name, level, credit, startingdate, completiondate, numberofassesments
-      */
-      public void display()
-      {
-          super. Display();
-          if(this.isRegistered==false)
-          {
-              System.out.println("Lecturer name is:"+ getlecturerName());
-              System.out.println("Level is:"+ getlevel());
-              System.out.println("Credit is:"+getcredit());
-              System.out.println("Starting Date is:"+getstartingdate());
-              System.out.println("Completion Date is:"+getcompletiondate());
-              System.out.println("NoofAssesments is:"+getNumberOfAssessment());
-          }
-      }
-  }
- }

```

- NON-ACADEMIC COURSE CLASS:

```

-
-      public class NonAcademicCourse extends course
-      {
-          //defining attributes
-          private String InstructorName;
-          private int duration;
-          private String startingDate;
-          private String completionDate;
-          private String examDate;
-          private String prerequisite;

```



```
- private boolean isRegistered;
- private boolean isRemoved;
-
- //defining constructors for Non Academic Course
- public NonAcademicCourse(String courseID, String courseName, int duration, String
prerequisite){
-     super(courseID,courseName,duration);
-     this.duration= duration;
-     this.prerequisite=prerequisite;
-     this.startingDate="";
-     this.examDate="";
-     this.completionDate="";
-     this.isRegistered = false;
-     this.isRemoved= false;
- }
-
- //defining corresponding accessor methods
- public String getInstructorName(){
-     return InstructorName ;
- }
-
- public int getDuration(){
-     return duration ;
- }
-
- public String getStartingDate(){
-     return startingDate;
- }
-
- public String getCompletionDate(){
-     return completionDate;
- }
-
- public String getExamDate(){
```

```
-         return examDate;
-     }
-
-     public String getPrerequisite(){
-         return examDate;
-     }
-
-     public boolean getIsRegistered(){
-         return isRegistered;
-     }
-
-     public boolean getIsRemoved(){
-         return isRemoved;
-     }
-
-     /*Defining method to set instructorName
-      * if the name is not registered it will be registered or a message will appear inthe
-      screen.
-      */
-     public void setInstructorName(String instructorName){
-         if (isRegistered == false){
-             this.InstructorName = instructorName;
-         }
-         else{
-             System.out.println("Already registered; Changing the instructor name is not
- allowed");
-         }
-     }
-
-     /*defining method to register
-      * instructorName along with starting date completion date and exam date are
-      registered
-      * and if they are already registered a suitable message will be shown
-      */
```

```
- public void register( String courseLeader, String InstructorName, String startingDate,
String completionDate, String examDate){
-     if (isRegistered == false){
-         setInstructorName(InstructorName);
-         this.startingDate= startingDate;
-         this.completionDate = completionDate;
-         this.examDate = examDate;
-         this.isRegistered= true;
-     }
-     else{
-         System.out.println("The course has already been registered.");
-     }
- }
-
- /* method to remove the non-academic courses
-  * if it is already removed a suitable message will display
-  */
- public void remove(){
-     if(isRemoved= true){
-         super.setcourseLeader("");
-         this.InstructorName = "";
-         this.startingDate = "";
-         this.completionDate = "";
-         this.examDate = "";
-         this.isRegistered = false;
-         this.isRemoved = true;
-     }
-     else{
-         System.out.println("The course is already removed");
-     }
- }
-
- /*defining method to display details of non academic course that includes
-  * courseID, coursename and duration if rnot registered then
```

```
-      * instructor name starting date completion date and exam date is also displayed
-      */
-      public void display()
-      {
-          super.Display();
-          if (isRegistered == true)
-          {
-              System.out.println("instructor name is: "+ this.InstructorName);
-              System.out.println("starting date is: "+ this.startingDate);
-              System.out.println("completion date is: "+ this.completionDate);
-              System.out.println("exam date is "+ this.examDate);
-          }
-      }
-  }
- }
```

- **ING_College Course Appendix:**

```
- import java.awt.*;
- import javax.swing.*;
- import java.awt.Color;
- import java.util.ArrayList;
- import java.awt.event.ActionEvent;
- import java.awt.event.ActionListener;
- import javax.swing.JLabel;
- import javax.swing.JPanel;
- import javax.swing.JFrame;
- import javax.swing.JTextField;
- import javax.swing.JComboBox;
- import javax.swing.JCheckBox;
- import javax.swing.JButton;
- import javax.swing.JTextArea;
- import java.awt.Font;
-
- public class ING_College implements ActionListener {
```

- //Declaring Variables
- JFrame frame;
- JPanel panel1,panel2;
- JLabel title;
- JLabel title2;
- JLabel CourseID,CourseID1;
- JTextField Course_ID,Course_ID1;
- JLabel CourseName,CourseName1;
- JTextField Course_Name,Course_Name1;
- JLabel CourseLeader,CourseLeader1;
- JTextField Course_Leader,Course_Leader1;
- JLabel LecturerName;
- JTextField Lecturer_Name;
- JLabel Level;
- JTextField level;
- JLabel Credit;
- JTextField credit;
- JLabel P1;
- JTextField p1;
- JLabel InstructorName;
- JTextField Instructor_Name;
- JLabel NoOfAssessments;
- JTextField No_Of_Assessments;
- JLabel S1,S2;
- JComboBox date_1;
- JComboBox month_1;
- JComboBox year_1;
- JLabel C1,C2;
- JComboBox date_2;
- JComboBox month_2;
- JComboBox year_2;
- JLabel E1;
- JComboBox date;
- JComboBox month;

- JComboBox year;
- JCheckBox term;
- JCheckBox term1;
- JLabel Duration,D1;
- JTextField duration,d1;
- JButton Display;
- JButton Clear;
- JButton Add;
- JButton Register;
- JButton Add1;
- JButton Register1;
- JButton Display1;
- JButton Clear1;
- JButton Remove;
- JButton Add_NC;
- JButton Add_C;
- JLabel res;
- JTextArea tout;
- JLabel res1;
- JTextArea tout1;
-
- String dates[]
- = { "1", "2", "3", "4", "5",
- "6", "7", "8", "9", "10",
- "11", "12", "13", "14", "15",
- "16", "17", "18", "19", "20",
- "21", "22", "23", "24", "25",
- "26", "27", "28", "29", "30",
- "31" };
- String months[]
- = { "Jan", "feb", "Mar", "Apr",
- "May", "Jun", "July", "Aug",
- "Sep", "Oct", "Nov", "Dec" };
- String years[]

```
-      = { "2020", "2021", "2022", "2023" };
-
-
-
-
-      //Creating ArrayList
-      ArrayList<course> Array_List = new ArrayList();
-      AcademicCourse Academic;
-      NonAcademicCourse nonAcademic;
-
-
-
-      //Creating the GUI Method
-      public void ING_College()
-      {
-          frame= new JFrame();
-          frame.setTitle("Course Registration");
-          frame.setBounds(0, 0, 1300, 700);
-          frame.setLayout(null);
-
-          /*
-           * Academic Course
-           */
-
-          panel1 = new JPanel();
-          panel1.setLayout(null);
-          Color CPanel1 = new Color(51,153,255);
-          panel1.setBackground(CPanel1);
-          panel1.setBounds(0, 0, 1400, 700);
-          panel1.setVisible(true);
-          frame.add(panel1);
-
-          title = new JLabel("Academic Course");
-          title.setFont(new Font("Arial", Font.PLAIN, 40));
-          title.setSize(500, 30);
-          title.setLocation(300, 30);
```

```
- panel1.add(title);  
-  
- CourseID = new JLabel("Course ID");  
- CourseID.setFont(new Font("Arial", Font.PLAIN, 20));  
- CourseID.setSize(100, 20);  
- CourseID.setLocation(100, 100);  
- panel1.add(CourseID);  
-  
- Course_ID = new JTextField();  
- Course_ID.setFont(new Font("Arial", Font.PLAIN, 15));  
- Course_ID.setSize(150, 20);  
- Course_ID.setLocation(250, 100);  
- panel1.add(Course_ID);  
-  
- CourseName = new JLabel("Course Name");  
- CourseName.setFont(new Font("Arial", Font.PLAIN, 20));  
- CourseName.setSize(250, 20);  
- CourseName.setLocation(100, 150);  
- panel1.add(CourseName);  
-  
- Course_Name= new JTextField();  
- Course_Name.setFont(new Font("Arial", Font.PLAIN, 15));  
- Course_Name.setSize(150, 20);  
- Course_Name.setLocation(250, 150);  
- panel1.add(Course_Name);  
-  
- CourseLeader = new JLabel("Course Leader");  
- CourseLeader.setFont(new Font("Arial", Font.PLAIN, 20));  
- CourseLeader.setSize(150, 20);  
- CourseLeader.setLocation(550, 300);  
- panel1.add(CourseLeader);  
-  
- Course_Leader = new JTextField();  
- Course_Leader.setFont(new Font("Arial", Font.PLAIN, 15));
```



```
- Course_Leader.setSize(150, 20);
- Course_Leader.setLocation(720, 300);
- panel1.add(Course_Leader);
-
- LecturerName = new JLabel("Lecturer Name");
- LecturerName.setFont(new Font("Arial", Font.PLAIN, 20));
- LecturerName.setSize(150, 20);
- LecturerName.setLocation(100, 300);
- panel1.add(LecturerName);
-
- Lecturer_Name = new JTextField();
- Lecturer_Name.setFont(new Font("Arial", Font.PLAIN, 15));
- Lecturer_Name.setSize(150, 20);
- Lecturer_Name.setLocation(250, 300);
- panel1.add(Lecturer_Name);
-
- Level = new JLabel("Level");
- Level.setFont(new Font("Arial", Font.PLAIN, 20));
- Level.setSize(150, 20);
- Level.setLocation(100, 200);
- panel1.add(Level);
-
- level = new JTextField();
- level.setFont(new Font("Arial", Font.PLAIN, 15));
- level.setSize(150, 20);
- level.setLocation(250, 200);
- panel1.add(level);
-
- Credit = new JLabel("Credit");
- Credit.setFont(new Font("Arial", Font.PLAIN, 20));
- Credit.setSize(150, 20);
- Credit.setLocation(550, 200);
- panel1.add(Credit);
-
```

```
- credit = new JTextField();
- credit.setFont(new Font("Arial", Font.PLAIN, 15));
- credit.setSize(150, 20);
- credit.setLocation(720, 200);
- panel1.add(credit);
-
- NoOfAssessments = new JLabel("Assessments");
- NoOfAssessments.setFont(new Font("Arial", Font.PLAIN, 20));
- NoOfAssessments.setSize(200, 20);
- NoOfAssessments.setLocation(550, 150);
- panel1.add(NoOfAssessments);
-
- No_Of_Assessments = new JTextField();
- No_Of_Assessments.setFont(new Font("Arial", Font.PLAIN, 15));
- No_Of_Assessments.setSize(150, 20);
- No_Of_Assessments.setLocation(720, 150);
- panel1.add(No_Of_Assessments);
-
- S1 = new JLabel("Start Date");
- S1.setFont(new Font("Arial", Font.PLAIN, 20));
- S1.setSize(150, 20);
- S1.setLocation(100, 350);
- panel1.add(S1);
-
- date_1 = new JComboBox(dates);
- date_1.setFont(new Font("Arial", Font.PLAIN, 15));
- date_1.setSize(60, 20);
- date_1.setLocation(200, 350);
- panel1.add(date_1);
-
- month_1 = new JComboBox(months);
- month_1.setFont(new Font("Arial", Font.PLAIN, 15));
- month_1.setSize(60, 20);
- month_1.setLocation(260, 350);
```

```
- panel1.add(month_1);  
-  
- year_1 = new JComboBox(years);  
- year_1.setFont(new Font("Arial", Font.PLAIN, 15));  
- year_1.setSize(60, 20);  
- year_1.setLocation(320, 350);  
- panel1.add(year_1);  
-  
- C1 = new JLabel("Completion Date");  
- C1.setFont(new Font("Arial", Font.PLAIN, 20));  
- C1.setSize(200, 20);  
- C1.setLocation(550, 350);  
- panel1.add(C1);  
-  
- date_2 = new JComboBox(dates);  
- date_2.setFont(new Font("Arial", Font.PLAIN, 15));  
- date_2.setSize(60, 20);  
- date_2.setLocation(700, 350);  
- panel1.add(date_2);  
-  
- month_2 = new JComboBox(months);  
- month_2.setFont(new Font("Arial", Font.PLAIN, 15));  
- month_2.setSize(60, 20);  
- month_2.setLocation(760, 350);  
- panel1.add(month_2);  
-  
- year_2 = new JComboBox(years);  
- year_2.setFont(new Font("Arial", Font.PLAIN, 15));  
- year_2.setSize(60, 20);  
- year_2.setLocation(820, 350);  
- panel1.add(year_2);  
-  
- Duration = new JLabel("Course Duration");  
- Duration.setFont(new Font("Arial", Font.PLAIN, 20));
```

```
- Duration.setSize(200, 20);
- Duration.setLocation(550, 100);
- panel1.add(Duration);
-
- duration= new JTextField();
- duration.setFont(new Font("Arial", Font.PLAIN, 15));
- duration.setSize(150, 20);
- duration.setLocation(720, 100);
- panel1.add(duration);
-
- Display= new JButton("Display");
- Display.setFont(new Font("Arial", Font.PLAIN, 15));
- Display.setSize(150, 30);
- Display.setLocation(300, 500);
- Display.addActionListener(this);
- panel1.add(Display);
-
- Clear = new JButton("Clear");
- Clear.setFont(new Font("Arial", Font.PLAIN, 15));
- Clear.setSize(150, 30);
- Clear.setLocation(500, 500);
- Clear.addActionListener(this);
- panel1.add(Clear);
-
- Add= new JButton("Add");
- Add.setFont(new Font("Arial", Font.PLAIN, 15));
- Add.setSize(150, 30);
- Add.setLocation(400,250);
- Add.addActionListener(this);
- panel1.add(Add);
-
- Register= new JButton("Register");
- Register.setFont(new Font("Arial", Font.PLAIN, 15));
- Register.setSize(150, 30);
```

```
- Register.setLocation(400,400);
- Register.addActionListener(this);
- panel1.add(Register);
-
- term1 = new JCheckBox("Enable Text Area");
- term1.setFont(new Font("Arial", Font.PLAIN, 15));
- term1.setSize(250, 20);
- term1.setLocation(100, 450);
- panel1.add(term1);
-
-
- Add_NC= new JButton("Add Non-Academic Course");
- Add_NC.setFont(new Font("Arial", Font.PLAIN, 15));
- Add_NC.setSize(250, 30);
- Add_NC.setLocation(350, 550);
- Add_NC.addActionListener(this);
- panel1.add(Add_NC);
-
-
- tout1 = new JTextArea();
- tout1.setFont(new Font("Arial", Font.PLAIN, 15));
- tout1.setSize(300, 400);
- tout1.setLocation(900, 100);
- tout1.setLineWrap(true);
- tout1.setEditable(false);
- panel1.add(tout1);
-
- res1 = new JLabel("");
- res1.setFont(new Font("Arial", Font.PLAIN, 20));
- res1.setSize(500, 25);
- res1.setLocation(900, 550);
- panel1.add(res1);
-
- /*
```

```
-      * Non Academic Course
-      */
-
-
-
-
-      panel2 = new JPanel();
-      panel2.setLayout(null);
-      Color CPanel2 = new Color(175,255,175);
-      panel2.setBackground(CPanel1);
-      panel2.setBounds(0, 0, 1400, 700);
-      panel2.setVisible(false);
-      frame.add(panel2);
-
-      title2 = new JLabel("Non Academic Course");
-      title2.setFont(new Font("Arial", Font.PLAIN, 40));
-      title2.setSize(500, 30);
-      title2.setLocation(300, 30);
-      panel2.add(title2);
-
-      CourseID1 = new JLabel("Course ID");
-      CourseID1.setFont(new Font("Arial", Font.PLAIN, 20));
-      CourseID1.setSize(100, 20);
-      CourseID1.setLocation(100, 100);
-      panel2.add(CourseID1);
-
-      Course_ID1 = new JTextField();
-      Course_ID1.setFont(new Font("Arial", Font.PLAIN, 15));
-      Course_ID1.setSize(150, 20);
-      Course_ID1.setLocation(250, 100);
-      panel2.add(Course_ID1);
-
-      CourseName1 = new JLabel("Course Name");
-      CourseName1.setFont(new Font("Arial", Font.PLAIN, 20));
-      CourseName1.setSize(250, 20);
```

```
- CourseName1.setLocation(100, 150);
- panel2.add(CourseName1);
-
- Course_Name1= new JTextField();
- Course_Name1.setFont(new Font("Arial", Font.PLAIN, 15));
- Course_Name1.setSize(150, 20);
- Course_Name1.setLocation(250, 150);
- panel2.add(Course_Name1);
-
- S2 = new JLabel("Start Date");
- S2.setFont(new Font("Arial", Font.PLAIN, 20));
- S2.setSize(150, 20);
- S2.setLocation(100, 300);
- panel2.add(S2);
-
- date_1 = new JComboBox(dates);
- date_1.setFont(new Font("Arial", Font.PLAIN, 15));
- date_1.setSize(60, 20);
- date_1.setLocation(200, 300);
- panel2.add(date_1);
-
- month_1 = new JComboBox(months);
- month_1.setFont(new Font("Arial", Font.PLAIN, 15));
- month_1.setSize(60, 20);
- month_1.setLocation(260, 300);
- panel2.add(month_1);
-
- year_1 = new JComboBox(years);
- year_1.setFont(new Font("Arial", Font.PLAIN, 15));
- year_1.setSize(60, 20);
- year_1.setLocation(320, 300);
- panel2.add(year_1);
-
- C2 = new JLabel("Completion Date");
```

```
- C2.setFont(new Font("Arial", Font.PLAIN, 20));
- C2.setSize(200, 20);
- C2.setLocation(550, 300);
- panel2.add(C2);
-
- date_2 = new JComboBox(dates);
- date_2.setFont(new Font("Arial", Font.PLAIN, 15));
- date_2.setSize(60, 20);
- date_2.setLocation(700, 300);
- panel2.add(date_2);
-
- month_2 = new JComboBox(months);
- month_2.setFont(new Font("Arial", Font.PLAIN, 15));
- month_2.setSize(60, 20);
- month_2.setLocation(760, 300);
- panel2.add(month_2);
-
- year_2 = new JComboBox(years);
- year_2.setFont(new Font("Arial", Font.PLAIN, 15));
- year_2.setSize(60, 20);
- year_2.setLocation(820, 300);
- panel2.add(year_2);
-
- E1= new JLabel("Exam Date");
- E1.setFont(new Font("Arial", Font.PLAIN, 20));
- E1.setSize(200, 20);
- E1.setLocation(350, 350);
- panel2.add(E1);
-
- date = new JComboBox(dates);
- date.setFont(new Font("Arial", Font.PLAIN, 15));
- date.setSize(60, 20);
- date.setLocation(450, 350);
- panel2.add(date);
```



```
-  
- month = new JComboBox(months);  
- month.setFont(new Font("Arial", Font.PLAIN, 15));  
- month.setSize(60, 20);  
- month.setLocation(510, 350);  
- panel2.add(month);  
-  
- year = new JComboBox(years);  
- year.setFont(new Font("Arial", Font.PLAIN, 15));  
- year.setSize(60, 20);  
- year.setLocation(570, 350);  
- panel2.add(year);  
-  
- InstructorName = new JLabel("Instructor Name");  
- InstructorName.setFont(new Font("Arial", Font.PLAIN, 20));  
- InstructorName.setSize(150, 20);  
- InstructorName.setLocation(100, 250);  
- panel2.add(InstructorName);  
-  
- Instructor_Name = new JTextField();  
- Instructor_Name.setFont(new Font("Arial", Font.PLAIN, 15));  
- Instructor_Name.setSize(150, 20);  
- Instructor_Name.setLocation(250, 250);  
- panel2.add(Instructor_Name);  
-  
- D1 = new JLabel("Course Duration");  
- D1.setFont(new Font("Arial", Font.PLAIN, 20));  
- D1.setSize(500, 20);  
- D1.setLocation(550, 100);  
- panel2.add(D1);  
-  
- d1= new JTextField();  
- d1.setFont(new Font("Arial", Font.PLAIN, 15));  
- d1.setSize(150, 20);
```

```
- d1.setLocation(720, 100);
- panel2.add(d1);
-
-
- CourseLeader1 = new JLabel("Course Leader");
- CourseLeader1.setFont(new Font("Arial", Font.PLAIN, 20));
- CourseLeader1.setSize(500, 20);
- CourseLeader1.setLocation(550, 250);
- panel2.add(CourseLeader1);
-
- Course_Leader1= new JTextField();
- Course_Leader1.setFont(new Font("Arial", Font.PLAIN, 15));
- Course_Leader1.setSize(150, 20);
- Course_Leader1.setLocation(720, 250);
- panel2.add(Course_Leader1);
-
- P1 = new JLabel("Prerequisites");
- P1.setFont(new Font("Arial", Font.PLAIN, 20));
- P1.setSize(500, 20);
- P1.setLocation(550, 150);
- panel2.add(P1);
-
- p1 = new JTextField();
- p1.setFont(new Font("Arial", Font.PLAIN, 15));
- p1.setSize(150, 20);
- p1.setLocation(720, 150);
- panel2.add(p1);
-
-
- Display1= new JButton("Display");
- Display1.setFont(new Font("Arial", Font.PLAIN, 15));
- Display1.setSize(150, 30);
- Display1.setLocation(300, 500);
- Display1.addActionListener(this);
```

```
- panel2.add(Display1);  
-  
- Clear1 = new JButton("Clear");  
- Clear1.setFont(new Font("Arial", Font.PLAIN, 15));  
- Clear1.setSize(150, 30);  
- Clear1.setLocation(500, 500);  
- Clear1.addActionListener(this);  
- panel2.add(Clear1);  
-  
- Add1= new JButton("Add");  
- Add1.setFont(new Font("Arial", Font.PLAIN, 15));  
- Add1.setSize(150, 30);  
- Add1.setLocation(400,200);  
- Add1.addActionListener(this);  
- panel2.add(Add1);  
-  
- Register1= new JButton("Register");  
- Register1.setFont(new Font("Arial", Font.PLAIN, 15));  
- Register1.setSize(150, 30);  
- Register1.setLocation(400,400);  
- Register1.addActionListener(this);  
- panel2.add(Register1);  
-  
- Remove= new JButton("Remove");  
- Remove.setFont(new Font("Arial", Font.PLAIN, 15));  
- Remove.setSize(150, 30);  
- Remove.setLocation(400, 550);  
- Remove.addActionListener(this);  
- panel2.add(Remove);  
-  
- Add_C= new JButton(" Academic Course");  
- Add_C.setFont(new Font("Arial", Font.PLAIN, 15));  
- Add_C.setSize(250, 30);  
- Add_C.setLocation(350, 600);
```

```
- Add_C.addActionListener(this);
- panel2.add(Add_C);
-
- term = new JCheckBox("Enable Text Area");
- term.setFont(new Font("Arial", Font.PLAIN, 15));
- term.setSize(250, 20);
- term.setLocation(100, 450);
- panel2.add(term);
-
-
-
- tout = new JTextArea();
- tout.setFont(new Font("Arial", Font.PLAIN, 15));
- tout.setSize(300, 400);
- tout.setLocation(900, 100);
- tout.setLineWrap(true);
- tout.setEditable(false);
- panel2.add(tout);
-
-
- res = new JLabel("");
- res.setFont(new Font("Arial", Font.PLAIN, 20));
- res.setSize(500, 25);
- res.setLocation(900, 550);
- panel2.add(res);
-
-
- frame.setLayout(null);
- frame.setVisible(true);
- frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
- frame.setResizable(true);
- }
-
- //Method For The Functionality Of The Buttons
-
- public void actionPerformed(ActionEvent e)
```

```
- {  
-     if(e.getSource()==Add_NC){  
-         panel1.setVisible(false);  
-         panel2.setVisible(true);  
-     }  
-  
-     if(e.getSource()==Add_C){  
-         panel1.setVisible(true);  
-         panel2.setVisible(false);  
-     }  
-  
-  
-     //For Adding Academic Course  
-     if (e.getSource() == Add)  
-     {  
-         try{  
-             String CourseID = Course_ID.getText();  
-             String CourseName = Course_Name.getText();  
-             String Credit = credit.getText();  
-             String Level = level.getText();  
-             String NoOfAssessments = No_Of_Assessments.getText();  
-             String Duration = duration.getText();  
-             boolean isDuplicateCID = false;  
-  
-             if(CourseID.isEmpty() || CourseName.isEmpty() || Credit.isEmpty() ||  
-             Level.isEmpty() || NoOfAssessments.isEmpty() || Duration .isEmpty()){  
-                 JOptionPane.showMessageDialog(frame, "Please enter all the text  
-                 fields!", "Error!", JOptionPane.ERROR_MESSAGE);  
-             }  
-             else{  
-                 int Duration1 = Integer.parseInt(duration.getText());  
-                 int NoOfAssessments1 = Integer.parseInt(No_Of_Assessments.getText());  
-  
-                 for(course var:Array_List){  
-                     if(Integer.parseInt(var.getcourseID()) == (Integer.parseInt(CourseID))){
```

```

-         isDuplicateCID = true;
-         break;
-     }
- }
-     if(isDuplicateCID == false){
-         Academic = new
AcademicCourse(CourseID,CourseName,Duration1,Credit,Level,NoOfAssessments1);
-         Array_List.add(Academic);
-         JOptionPane.showMessageDialog(frame, "Academic Course is now
successfully added","MESSAGE",JOptionPane.INFORMATION_MESSAGE);
-     }
-     else{
-         JOptionPane.showMessageDialog(frame, "Course ID has been
repeated","ERROR!!",JOptionPane.ERROR_MESSAGE);
-     }
-     }
-     }
-     catch(Exception A){
-         JOptionPane.showMessageDialog(frame,"Please be sure if all the data are
inserted correctly","ERROR!!",JOptionPane.ERROR_MESSAGE);
-     }
- }
- //For Registering Academic Course
-     else if(e.getSource() == Register){
-         try{
-             String courseID = Course_ID.getText();
-             String courseLeader = Course_Leader.getText();
-             String lecturerName = Lecturer_Name.getText();
-             String S1=(date_1.getSelectedItemId()).toString()+
(month_1.getSelectedItemId()).toString() + (year_1.getSelectedItemId()).toString());
-             String C1
=(date_2.getSelectedItemId()).toString()+(month_2.getSelectedItemId()).toString()+(year_2.
getSelectedItemId()).toString());
-             boolean isalreadyreg=true;

```

```
-         for(course var:Array_List){
-             if(var.getCourseID() .equals (courseID)){
-                 if(var instanceof AcademicCourse){
-                     isAlreadyreg=true;
-                     AcademicCourse REGaca = (AcademicCourse) var;
-                     if(REGaca.getisRegistered() == false){
-                         JOptionPane.showMessageDialog(frame,"Course is already
Registered","ERROR",JOptionPane.ERROR_MESSAGE);
-                     }
-                     else{
-                         REGaca.register(courseLeader,lecturerName,S1,C1);
-                         JOptionPane.showMessageDialog(frame,"Academic Course has
been Registered ","MESSAGE",JOptionPane.INFORMATION_MESSAGE);
-                     }
-                 }
-             }
-             else{
-                 JOptionPane.showMessageDialog(frame,"Course not
Found!","ERROR",JOptionPane.ERROR_MESSAGE);
-             }
-         }
-         if(!isAlreadyreg){
-             JOptionPane.showMessageDialog(frame,"Course ID
Invalid","ERROR",JOptionPane.ERROR_MESSAGE);
-         }
-     }
-     catch(Exception A){
-         JOptionPane.showMessageDialog(frame,"Please enter the correct
details","ERROR",JOptionPane.ERROR_MESSAGE);
-     }
- }
-
- //For Displaying The Data Of Academic Course
-
```

```
-     else if (e.getSource() == Display) {  
-         if (term1.isSelected()) {  
-             String data1;  
-             String data  
-                 = "Course ID : "  
-                 + Course_ID.getText() + "\n"  
-                 + "Course name : "  
-                 + Course_Name.getText() + "\n";  
-  
-  
-             String data2  
-                 = "Start Date : "  
-                 + (String)date_1.getSelectedItemAt()  
-                 + "/" + (String)month_1.getSelectedItemAt()  
-                 + "/" + (String)year_1.getSelectedItemAt()  
-                 + "\n";  
-  
-             String data3  
-                 = "Completion Date : "  
-                 + (String)date_2.getSelectedItemAt()  
-                 + "/" + (String)month_2.getSelectedItemAt()  
-                 + "/" + (String)year_2.getSelectedItemAt()  
-                 + "\n";  
-  
-             String data4  
-                 = "Course leader : "  
-                 + Course_Leader.getText() + "\n"  
-                 + "Lecturer name : "  
-                 + Lecturer_Name.getText() + "\n";  
-  
-             String data5= "Credit: "  
-                 + credit.getText() + "\n"  
-                 + "Level: "  
-                 + level.getText() + "\n";
```



```
-  
-  
-      String data6= "Duration"  
-      + (String)duration.getText() + "\n";  
-  
-  
-      String data7= "Assessments"  
-      + (String)No_Of_Assessments.getText() + "\n";  
-  
-      tout1.setText(data +data5 + data6 + data7+ data2 + data3 + data4 );  
-      tout1.setEditable(false);  
-  
-  
-      res1.setText("Check if the displayed data is correct");  
-  }  
- }  
-  
-  
- //For Clearing The Data Of Academic Course From The GUI  
-  
- else if (e.getSource() == Clear) {  
-   String def = "";  
-   Course_ID.setText(def);  
-   Course_Name.setText(def);  
-   Course_Leader.setText(def);  
-   Lecturer_Name.setText(def);  
-   level.setText(def);  
-   credit.setText(def);  
-   term1.setSelected(false);  
-   date_1 .setSelectedIndex(0);  
-   month_1 .setSelectedIndex(0);  
-   year_1 .setSelectedIndex(0);  
-   date_2 .setSelectedIndex(0);  
-   month_2 .setSelectedIndex(0);  
-   year_2 .setSelectedIndex(0);
```

```
-         duration.setText(def);
-         No_Of_Assessments.setText(def);
-         tout1.setText(def);
-         res1.setText(def);
-     }
-
-     //Adding Button To Add Non Academic Course
-     if (e.getSource() == Add1)
-     {
-         try{
-             String CourseID1 = Course_ID1.getText();
-             String CourseName1 = Course_Name1.getText();
-             String P1 = p1.getText();
-             String D1= d1.getText();
-             boolean isDuplicateCID = false;
-
-             if(CourseID1.isEmpty() || CourseName1.isEmpty() || P1.isEmpty() ||
D1.isEmpty()){
-                 JOptionPane.showMessageDialog(frame, "Please enter all the text
feilds!","Error!",JOptionPane.ERROR_MESSAGE);
-             }
-             else{
-                 int Duration1 = Integer.parseInt(d1.getText());
-
-                 for(course var:Array_List){
-                     if(var.getcourseID().equals (CourseID)){
-                         isDuplicateCID = true;
-                         break;
-                     }
-                 }
-
-                 if(isDuplicateCID == false){
-                     nonAcademic = new NonAcademicCourse(CourseID1,
CourseName1,Duration1, P1);
-                     Array_List.add(nonAcademic);
```

```

-             JOptionPane.showMessageDialog(frame, "Non-Academic Course
added","MESSAGE",JOptionPane.INFORMATION_MESSAGE);
-         }
-         else{
-             JOptionPane.showMessageDialog(frame, "Course ID has been
repeated","ERROR!!",JOptionPane.ERROR_MESSAGE);
-         }
-     }
- }
- }
- catch(Exception A){
-     JOptionPane.showMessageDialog(frame, "Please make sure that all the data
are inserted correctly","ERROR!!",JOptionPane.ERROR_MESSAGE);
- }
- }
-
- //Adding Button To Register Non Academic Course
- else if(e.getSource() == Register1){
-     try{
-         String CourseID1 = Course_ID1.getText();
-         String CourseLeader1 = Course_Leader1.getText();
-         String InstructorName = Instructor_Name.getText();
-         String S2=(date_1.getSelectedItemAt()).toString()+
(month_1.getSelectedItemAt()).toString() + (year_1.getSelectedItemAt()).toString());
-         String C2
=(date_2.getSelectedItemAt()).toString()+(month_2.getSelectedItemAt()).toString()+(year_2.
getSelectedItem().toString());
-         String E1
=(date.getSelectedItemAt()).toString()+(month.getSelectedItemAt()).toString()+(year.getSele
ctedItemAt()).toString());
-         boolean coursefound = false;
-
-         for(course var:Array_List){
-             if(var.getCourseID().equals (CourseID1)){
-                 coursefound = true;

```

```

-         if(var instanceof NonAcademicCourse){
-             NonAcademicCourse REGnaca = (NonAcademicCourse) var;
-             if(REGnaca.getIsRegistered()== true){
-                 JOptionPane.showMessageDialog(frame,"Course is already
Registered","ERROR",JOptionPane.ERROR_MESSAGE);
-             }
-             else{
-                 REGnaca.register(CourseLeader1,InstructorName,S2,C2,E1);
-                 JOptionPane.showMessageDialog(frame,"Non Academic Course
has been Registered
Successfully","MESSAGE",JOptionPane.INFORMATION_MESSAGE);
-             }
-         }
-         else{
-             JOptionPane.showMessageDialog(frame,"Course not
Found!","ERROR",JOptionPane.ERROR_MESSAGE);
-         }
-     }
-     if (!coursefound){
-         JOptionPane.showMessageDialog(frame,"CourseID
Invalid","ERROR",JOptionPane.ERROR_MESSAGE);
-     }
- }
- catch(Exception A){
-     JOptionPane.showMessageDialog(frame,"Please enter valid
details","ERROR",JOptionPane.ERROR_MESSAGE);
- }
- }
-
- //To Display Non Academic Course
- else if (e.getSource() == Display1) {
-     if (term.isSelected()) {
-         String data1;

```

```
- String data
- = "Course ID : "
- + Course_ID1.getText() + "\n"
- + "Course name : "
- + Course_Name1.getText() + "\n";
-
-
- String data2
- = "Start Date : "
- + (String)date_1.getSelectedItem()
- + "/" + (String)month_1.getSelectedItem()
- + "/" + (String)year_1.getSelectedItem()
- + "\n";
-
- String data3
- = "Completion Date : "
- + (String)date_2.getSelectedItem()
- + "/" + (String)month_2.getSelectedItem()
- + "/" + (String)year_2.getSelectedItem()
- + "\n";
-
- String data4
- = "Course leader : "
- + Course_Leader1.getText() + "\n";
-
- String data5= "Duration"
- + (String)d1.getText() + "\n";
-
-
- String data6= "Prerequisites"
- + (String)p1.getText() + "\n";
-
- String data7
- = "Exam Date : "
```

```
-         + (String)date.getSelectedltem()
-         + "/" + (String)month.getSelectedltem()
-         + "/" + (String)year.getSelectedltem()
-         + "\n";
-
-     String data8
-         = "Instructor Name : "
-         + Instructor_Name.getText() + "\n";
-
-     tout.setText(data + data5 + data6 + data2 + data3 + data7 + data8 + data4 );
-     tout.setEditable(false);
-
-     res.setText("Check if the displayed data is correct");
-     }
- }
-
- //To Remove Non Academic Course
- else if(e.getSource() == Remove){
-     try{
-         String courseID = Course_ID1.getText();
-         if(courseID.isEmpty()){
-             JOptionPane.showMessageDialog(frame, "Please insert the Course ID
before removing Course","Error!!!", JOptionPane.ERROR_MESSAGE);
-         }
-         else{
-             boolean isDuplicateNaCID = false;
-             for(course var:Array_List){
-                 if(var.getcourseID().equals (courseID)){
-                     isDuplicateNaCID = true;
-                     if(var instanceof NonAcademicCourse){
-                         nonAcademic = (NonAcademicCourse) var;
-                         if(nonAcademic.getIsRemoved()==true){
-                             JOptionPane.showMessageDialog(frame,"Course is already
removed","Error", JOptionPane.ERROR_MESSAGE);
```

```
-         }
-         else{
-             nonAcademic.remove();
-             JOptionPane.showMessageDialog(frame,"Course Removed
Successfully","MESSAGE",JOptionPane.INFORMATION_MESSAGE);
-             break;
-         }
-     }
-     else{
-         JOptionPane.showMessageDialog(frame," Error has
occured","ERROR",JOptionPane.ERROR_MESSAGE);
-         break;
-     }
- }
- }
-     if(!isDuplicateNaCID){
-         JOptionPane.showMessageDialog(frame,"Course ID Invalid
","ERROR",JOptionPane.ERROR_MESSAGE);
-     }
- }
- }
-     catch(NumberFormatException E){
-         JOptionPane.showMessageDialog(frame, "check Course ID. ","ERROR!!",
JOptionPane.ERROR_MESSAGE);
-     }
- }
-
- //To Clear Non Academic Course
-
-     else if (e.getSource() == Clear1) {
-         String def = "";
-         Course_ID1.setText(def);
-         Course_Name1.setText(def);
-         Course_Leader1.setText(def);
```

```
-      term.setSelected(false);
-      date.setSelectedIndex(0);
-      month.setSelectedIndex(0);
-      year.setSelectedIndex(0);
-      Instructor_Name.setText(def);
-      date_1.setSelectedIndex(0);
-      month_1.setSelectedIndex(0);
-      year_1.setSelectedIndex(0);
-      date_2.setSelectedIndex(0);
-      month_2.setSelectedIndex(0);
-      year_2.setSelectedIndex(0);
-      d1.setText(def);
-      p1.setText(def);
-      tout.setText(def);
-      res.setText(def);
-  }
-  }
-  public String getCourseID(){
-      return Course_ID.getText();
-  }
-
-  public String getCourseName(){
-      return Course_Name.getText();
-  }
-
-  public String getDuration(){
-      return duration.getText();
-  }
-
-  public String getCourseLeader(){
-      return Course_Leader.getText();
-  }
-
-  public String getLecturerName(){
```



```
-         return Lecturer_Name.getText();  
-     }  
-  
-     public String getLevel(){  
-         return level.getText();  
-     }  
-  
-     public String getCredit(){  
-         return credit.getText();  
-     }  
-  
-     public String getNoOfAssessments(){  
-         return No_Of_Assessments.getText();  
-     }  
-  
-     public String getS1(){  
-         return (date_1.getSelectedItemId()).toString()+  
-         (month_1.getSelectedItemId()).toString() + (year_1.getSelectedItemId()).toString();  
-     }  
-  
-     public String getC1(){  
-         return (date_2.getSelectedItemId()).toString()+  
-         (month_2.getSelectedItemId()).toString() + (year_2.getSelectedItemId()).toString();  
-     }  
-  
-     public String getCourseID1(){  
-         return Course_ID1.getText();  
-     }  
-  
-     public String getCourseName1(){  
-         return Course_Name1.getText();  
-     }  
-  
-     public String getCourseLeader1(){
```

```
-         return Course_Leader1.getText();
-     }
-
-     public String getD1(){
-         return d1.getText();
-     }
-
-     public String getS2(){
-         return (date_1.getSelectedItem().toString()+
- (month_1.getSelectedItem().toString()) + (year_1.getSelectedItem().toString()));
-     }
-
-     public String getC2(){
-         return (date_2.getSelectedItem().toString()+
- (month_2.getSelectedItem().toString()) + (year_2.getSelectedItem().toString()));
-     }
-
-     public String getE1(){
-         return (date.getSelectedItem().toString()+ (month.getSelectedItem().toString()) +
- (year.getSelectedItem().toString()));
-     }
-
-     public String getInstructorName(){
-         return Instructor_Name.getText();
-     }
-
-     public String getP1(){
-         return p1.getText();
-     }
-
-     //Main Method
-
-     public static void main(String args[]){
-         ING_College Main=new ING_College();
```

```
-      Main.ING_College();  
-    }  
- }
```