LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS4001NI Programming

COURSEWORK-2

Assessment Weightage & Type

30% Individual Coursework

Semester and Year

Spring 2021

Student Name:Sahil Sainju

Group: C9

London Met ID: 20048999

College ID: NP01CP4S210243

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

Table of Tables

1. Introduction

As part of the programming module curriculum, we were given the following coursework. This is second coursework for this module. In our first coursework we created a Java project that implements a real-world scenario utilizing Java's object-oriented concepts. The project included three classes (Course, AcademicCourse, and NonAcademicCourse). Course has two subclasses: AcademicCourse and NonAcademicCourse.

In this cousework we will add a class to the project that we developed for the first part of the coursework to make a graphical user interface (GUI) for a system that stores details of Course that includes both academic and non-academic course. The class will contain a main method and will be tested using the command prompt.

We will be using BlueJ, a Java Programming Language program that was created primarily for instructional purposes. It is also appropriate for small-scale software development, which is precisely what we will be doing in this project. It runs with the help of JDK (Java Development Kit).

The Java Development Kit (JDK) is a software development environment that includes a set of tools and libraries used to create Java applications. The Java Development Kit (JDK) is needed to translate your source code into a format that the Java Runtime Environment (JRE) will execute

We will also be using MS Word in order write a report about the program. MS Word is a word processor developed by Microsoft that is used to create professional-quality documents, letters, reports, and so on. It includes comprehensive capabilities that enable you to format and modify your files and documents to the best of your ability.

We will also be using Draw.io. Draw.io is a flowchart solution that allows developers, network administrators, IT analysts, and designers to build and distribute diagrams using drag-and-drop capabilities. Professionals may use the system to build toggle layers with customized URLs and align words within the shapes.

2. Class Diagram

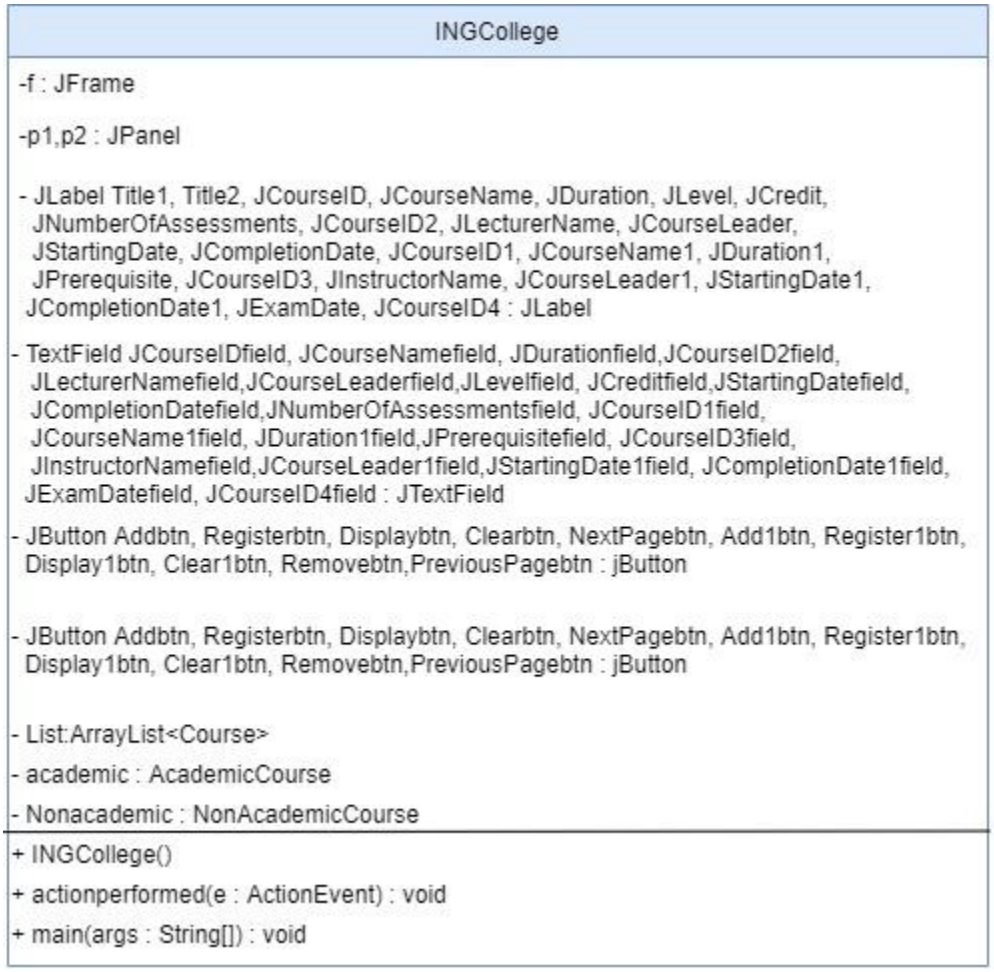

Figure 1: Fig Class Diagram for INGCollege

3. Pseudocode

Pseudocode is a casual way of describing programming that does not require any rigid programming language syntax or underlying technology considerations. It is used to create a project description or rough draft.

3.1. Pseudocode for course class

CREATE class Course

DECLARE instance variable CourseID of String type
DECLARE instance variable CourseName of String type
DECLARE instance variable CourseLeader of String type
DECLARE instance variable CourseDuration of int type

CREATE Constructor Course(**PASS** parameter CourseID of String type, CourseName of String type, CourseDuration of int type)
ASSIGN parameter CourseID to instance variable CourseID
ASSIGN parameter CourseName to instance variable CourseName
ASSIGN parameter CourseDuration to instance variable CourseDuration
ASSIGN CourseLeader to null

CREATE method getCourseID with return type String
RETURN CourseID

CREATE method getCourseName with return type String
RETURN CourseName

CREATE method getCourseLeader with return type String
RETURN CourseLeader

CREATE method getCourseDuration with return type int

RETURN CourseDuration

CREATE method setCourseLeader (**PASS** parameter newCourseLeader of String Type)

ASSIGN parameter CourseLeader to instance variable newCourseLeader

CREATE method Display

PRINT “CourseID”

PRINT” CourseName”

PRINT “CourseDuration”

IF (CourseLeader is null)

PRINT “CourseLeader”

ENDIF

3.2. Pseudocode for AcademicCourse class

CREATE class AcademicCourse

DECLARE instance variable LecturerName of String type

DECLARE instance variable Level of String type

DECLARE instance variable Credit of String type

DECLARE instance variable StartingDate of String type

DECLARE instance variable CompletionDate of String type

DECLARE instance variable NumberOfAssessments of int type

DECLARE instance variable isRegistered of boolean type

CREATE Constructor AcademicCourse (**PASS** parameter CourseID of String type, CourseName of String type, CourseDuration of int type, Level of String type, Credit of String type, NumberOfAssessments of int type)

CALL constructor from parent class

ASSIGN parameter CourseID to instance variable CourseID

ASSIGN parameter CourseName to instance variable CourseName

ASSIGN parameter CourseDuration to instance variable CourseDuration

ASSIGN parameter Level to instance variable Level

ASSIGN parameter NumberOfAssessments to instance variable NumberOfAssessments

ASSIGN parameter Credit to instance variable Credit

ASSIGN LecturerName to null

ASSIGN StartingDate to null

ASSIGN CompletionDate to null

ASSIGN isRegistered to false

CREATE method getLecturerName with return type String

RETURN LecturerName

CREATE method getLevel with return type String

RETURN Level

CREATE method getCredit with return type String

RETURN Credit

CREATE method getStartingDate with return type String

RETURN StartingDate

CREATE method getCompletionDate with return type String

RETURN CompletionDate

CREATE method getNumberOfAssessments with return type int

RETURN NumberOfAssessments

CREATE method getisRegistered with return type boolean

RETURN isRegistered

CREATE method setLecturerName (**PASS** parameter newLecturerName of String Type)

ASSIGN parameter newLecturerName to instance variable LecturerName

CREATE method setNumberOfAssessments (**PASS** parameter newNumberOfAssessments of String Type)

ASSIGN parameter newNumberOfAssessments to instance variable NumberOfAssessments

CREATE method Register (**PASS** parameter CourseLeader of String type, LecturerName of String type, StartingDate of String type, CompletionDate of String type)

IF (isRegistered is true)

PRINT “Academic course is already registered”

PRINT “LecturerName”

PRINT “StartingDate”

PRINT “CompletionDate”

ELSE

CALL setCourseLeader method from parent class

ASSIGN parameter LecturerName to instance variable LecturerName

ASSIGN parameter StartingDate to instance variable StartingDate

```
                    ASSIGN parameter CompletionDate to instance variable
                    CompletionDate
                    ASSIGN isRegistered to true

CREATE method Display
        CALL Display method from parent class
        IF (isRegistered is true)
                PRINT "LecturerName"
                PRINT "Level"
                PRINT "Credit"
                PRINT "StartingDate"
                PRINT "CompletionDate"
                PRINT "NumberOfAssessments"

 ENDIF
```

3.3. Pseudocode for NonAcademicCourse class

```
CREATE NonAcademicCourse

        DECLARE instance variable InstructorName of String type
        DECLARE instance variable CourseDuration of int type
        DECLARE instance variable StartingDate of String type
        DECLARE instance variable CompletionDate of String type
        DECLARE instance variable ExamDate of String type
        DECLARE instance variable Prerequisite of String type
        DECLARE instance variable isRegistered of boolean type
        DECLARE instance variable isRemoved of boolean type
```

CREATE Constructor NonAcademicCourse (**PASS** parameter CourseID of String type, CourseName of String type, CourseDuration of int type, Prerequisite of String type)

CALL constructor from parent class
ASSIGN parameter CourseID to instance variable CourseID
ASSIGN parameter CourseName to instance variable CourseName
ASSIGN parameter CourseDuration to instance variable CourseDuration
ASSIGN parameter Prerequisite to instance variable Prerequisite
ASSIGN StartingDate to null
ASSIGN CompletionDate to null
ASSIGN ExamDate to null
ASSIGN isRegistered to false
ASSIGN isRemoved to false

CREATE method getInstructorName with return type String
RETURN InstructorName

CREATE method getCourseDuration with int type String
RETURN CourseDuration

CREATE method getStartingDate with return type String
RETURN StartingDate

CREATE method getCompletionDate with return type String
RETURN CompletionDate

CREATE method getExamDate with return type String
RETURN ExamDate

CREATE method getPrerequisite with return type String
RETURN Prerequisite

CREATE method getisRegistered with return type boolean

RETURN isRegistered

CREATE method getisRemoved with return type boolean

RETURN isRemoved

CREATE method setInstructorName (**PASS** parameter newInstructorName of String Type)

IF (isRegistered is true)

PRINT “Since instructor is already registered, it is not possible”

ELSE

ASSIGN parameter newInstructorName to instance variable InstructorName

CREATE method Register (**PASS** parameter CourseLeader of String type, InstructorName of String type, StartingDate of String type, CompletionDate of String type, ExamDate of String type)

IF (isRegistered is true)

PRINT “Non-academic course is already registered”

ELSE

CALL method setCourseLeader from parent class

ASSIGN parameter InstructorName to instance variable LecturerName

ASSIGN parameter StartingDate to instance variable StartingDate

ASSIGN parameter CompletionDate to instance variable CompletionDate

ASSIGN parameter ExamDate to instance variable ExamDate

isRegistered to true

ASSIGN ASSIGN isRemoved to false

CREATE method Remove

```
IF (isRemoved is true)
        PRINT "Non-Academic course is already removed."
ELSE
        CALL method setCourseLeader from parent class(PASS null in instance
        variable)
        ASSIGN instance variable InstructorName to null
        ASSIGN instance variable StartingDate to null
        ASSIGN instance variable CompletionDate to null
        ASSIGN instance variable ExamDate to null
        ASSIGN instance variable isRegistered to false
        ASSIGN instance variable isRemoved to true

CREATE method Display
        CALL Display method from parent class
    IF (isRegistered is true)
        PRINT "InstructorName"
        PRINT "StartingDate"
        PRINT "CompletionDate"
        PRINT "ExamDate"
        PRINT "Prerequisite"
ENDIF
```

3.4. Pseudocode for INGCollege class

```
CREATE class INGCollege which IMPLEMENTS the interface ActionListener

        DECLARE  instance variables
        F as JFrame,
        j1,j2 as Jpanel
```

Title1, Title2, JCourseID, JCourseName, JDuration, JLevel, JCredit, JNumberOfAssessments, JCourseID2, JLecturerName, JCourseLeader,JStartingDate, JCompletionDate, JCourseID1, JCourseName1, JDuration1, JPrerequisite, JCourseID3, JInstructorName, JCourseLeader1, JStartingDate1, JCompletionDate1, JExamDate, JCourseID4 as JLabel,

JCourseIDfield, JCourseNamefield, JDurationfield, JCourseID2field, JLecturerNamefield, JCourseLeaderfield, JLevelfield, JCreditfield, JStartingDatefield, JCompletionDatefield, JNumberOfAssessmentsfield, JCourseID1field, JCourseName1field, JDuration1field, JPrerequisitefield, JCourseID3field,JInstructorNamefield, JCourseLeader1field, JStartingDate1field, JCompletionDate1field, JExamDatefield, JCourseID4field as JTextField,

Addbtn, Registerbtn, Displaybtn, Clearbtn, NextPagebtn, Add1btn, Register1btn, Display1btn, Clear1btn,,PreviousPagebtn as JButton,

INITIALIZE al as new ArrayList which holds objects of Course and its subclasses
DECLARE object academic of Academic Course
DECLARE object Nonacademic of Non-Academic Course
CREATE constructor INGCollege
    INITIALIZE JFrame frame as “Course Registration”
    SET VISIBLE JFrame j as true
    SET LAYOUT for JFrame j as null
    SET SIZE for JFrame j

    INITIALIZE p1 as new JPanel
    SET VISBILE JPanel p1 as true
    SET LAYOUT for JPanel p1 as null

SET SIZE for JPanel p1

INITIALIZE p2 as new JPanel
SET VISBILE JPanel p2 as false
SET LAYOUT for JPanel p2 as null
SET SIZE for JPanel p2

INITIALIZE JLabel Title1 as “Academic Course”
SET FONT as Arial, Bold and 25 size
SET BOUNDS to JLabel Title1
ADD Title1 in JPanel p1
SET FOREGROUND color as blue

INITIALIZE JLabel Title2 as “Non-Academic Course”
SET FONT as Arial, Bold and 25 size
SET BOUNDS to JLabel Title2

INITIALIZE JLabel CourseID as “CourseID”
SET BOUNDS to JLabel CourseID
INITIALIZE JCourseIDfield as new JTextField
SET BOUNDS to JTextField JCourseIDfield

INITIALIZE JLabel JCourseName as “Course Name”
SET BOUNDS to JLabel JCourseName
INITIALIZE JCourseNamefield as new JTextField
SET BOUNDS to JTextField JCourseNamefield

INITIALIZE JLabel JDuration as “Duration”

SET BOUNDS to JLabel JDuration

INITIALIZE JDurationfield as new JTextField

SET BOUNDS to JTextField JDurationfield

INITIALIZE JLabel JLevel as “Level”

SET BOUNDS to JLabel JLevel

INITIALIZE JLevelfield as new JTextField

SET BOUNDS to new JTextField JLevelfield

INITIALIZE JLabel JCreditas “Credit”

SET BOUNDS to JLabel JCredit

INITIALIZE JCreditfieldas new JTextField

SET BOUNDS to JTextField JCreditfield

INITIALIZE JLabel JNumberOfAssessmentsas “Assessment”

SET BOUNDS to JLabel JNumberOfAssessments

INITIALIZE JNumberOfAssessmentsfield as new JTextField

SET BOUNDS to JTextField JNumberOfAssessmentsfield

INITIALIZE JLabel JCourseID1as “CourseID”

SET BOUNDS to JLabel JCourseID1

INITIALIZE txtnonid as new JTextField

SET BOUNDS to JTextField JCourseID1field

ADD JCourseID1fieldin JPanel p2

INITIALIZE JLabel JCourseName1as “Course Name”

SET BOUNDS to JLabel JCourseName1

INITIALIZE JCourseName1fieldas new JTextField

SET BOUNDS to JTextField JCourseName1field

INITIALIZE JLabel JDuration1as “Duration”

SET BOUNDS to JLabel JDuration1

INITIALIZE JDuration1fieldas new JTextField

SET BOUNDS to JTextField JDuration1field

INITIALIZE JLabel JPrerequisiteas “Prerequisite”

SET BOUNDS to JLabel JPrerequisite

INITIALIZE JPrerequisitefield as new JTextField

SET BOUNDS to JTextField JPrerequisitefield

INITIALIZE JLabel JCourseID2as “CourseID”

SET BOUNDS to JLabel JCourseID2

INITIALIZE JCourseID2field as new JTextField

SET BOUNDS to JTextField JCourseID2field

INITIALIZE JLabel JLecturerNameas “Lecturer Name”

SET BOUNDS to JLabel JLecturerName

INITIALIZE JLecturerNamefield as new JTextField

SET BOUNDS to JTextField JLecturerNamefield

INITIALIZE JLabel JCourseLeader as“CourseLeader”

SET BOUNDS to JLabel JCourseLeader

INITIALIZE JCourseLeaderfieldas new JTextField

SET BOUNDS to JTextField JCourseLeaderfield

INITIALIZE JLabel JStartingDate as “Start Date”

SET BOUNDS to JLabel JStartingDate

INITIALIZE JStartingDatefieldas new JTextField
SET BOUNDS to JTextField JStartingDatefield

INITIALIZE JLabel JCompletionDate as “Completion Date”
SET BOUNDS to JLabel JCompletionDate
INITIALIZE JCompletionDatefieldas new JTextField
SET BOUNDS to JTextField JCompletionDatefield

INITIALIZE JLabel JCourseID3 as “CourseID”
SET BOUNDS to JLabel JCourseID3
INITIALIZE JCourseID3fieldas new JTextField
SET BOUNDS to JTextField JCourseID3field

INITIALIZE JLabel JInstructorNameas “Instructor Name”
SET BOUNDS to JLabel JInstructorName
INITIALIZE txtinstructor as new JTextField
SET BOUNDS to JTextField txtinstructor

INITIALIZE JLabel JInstructorNamefieldas “Start Date”
SET BOUNDS to JLabel JInstructorNamefield
INITIALIZE txtnonstartdate as new JTextField
SET BOUNDS to JTextField txtnonstartdate

INITIALIZE JLabel JCourseLeader1 “Course Leader”
SET BOUNDS to JLabel JCourseLeader1
INITIALIZE JCourseLeader1fieldas new JTextField
SET BOUNDS to JTextField JCourseLeader1field

INITIALIZE JLabel JStartingDate1as “Starting Date”

SET BOUNDS to JLabel JStartingDate1

INITIALIZE JStartingDate1fieldas new JTextField

SET BOUNDS to JTextField JStartingDate1field

INITIALIZE JLabel JCompletionDate1 as “Completion Date”

SET BOUNDS to JLabel JCompletionDate1

INITIALIZE JCompletionDate1 field as new JTextField

SET BOUNDS to JTextField JCompletionDate1field

INITIALIZE JLabel JExamDate as “Exam Date”

SET BOUNDS to JLabel JExamDate

INITIALIZE JExamDatefield as new JTextField

SET BOUNDS to JTextField JExamDatefield

INITIALIZE JLabel JCourseID4 as “CourseID”

SET BOUNDS to JLabel JCourseID4

INITIALIZE JCourseID4fieldas new JTextField

SET BOUNDS to JTextField JCourseID4field

INITIALIZE JButton Addbtn as “Add”

SET BOUNDS to JButton Addbtn

INITIALIZE JButton Registerbtnas “Register”

SET BOUNDS to JButton Registerbtn

INITIALIZE JButton Displaybtn as “Display”

SET BOUNDS to JButton Displaybtn

INITIALIZE JButton Clearbtn as “Clear”

SET BOUNDS to JButton Clearbtn

INITIALIZE JButton NextPagebtnas “Next Page”
SET BOUNDS to JButton NextPagebtn

INITIALIZE JButton previousbtn as “Previous Page”
SET BOUNDS to JButton previousbtn

INITIALIZE JButton Add1btn “Add”
SET BOUNDS to JButton Add1btn

INITIALIZE JButton Register1btnas “Register”
SET BOUNDS to JButton Register1btn

INITIALIZE JButton Display1btnas “Display”
SET BOUNDS to JButton Display1btn

INITIALIZE JButton Clear1btn as “Clear”
SET BOUNDS to JButton Clear1btn

INITIALIZE Removebtn Display1btnas “Remove”
SET BOUNDS to JButton Removebtn

ADD Title1 in JPanel p1
ADD JCourseID in JPanel p1
ADD JCourseName in JPanel p1
ADD JDuration in JPanel p1
ADD JCourseIDfield in JPanel p1
ADD JCourseNamefield in JPanel p1
ADD JDurationfield in JPanel p1
ADD Addbtn in JPanel p1
ADD JCourseID2in JPanel p1

ADD JLecturerName in JPanel p1

ADD JCourseLeader in JPanel p1

ADD JLevel in JPanel p1

ADD JCredit in JPanel p1

ADD JStartingDate in JPanel p1

ADD JCompletionDate in JPanel p1

ADD JNumberOfAssessments in JPanel p1

ADD JCourseID2field in JPanel p1

ADD JLecturerNamefield in JPanel p1

ADD JCourseLeaderfield in JPanel p1

ADD JLevelfield in JPanel p1

ADD JCreditfield in JPanel p1

ADD JStartingDatefield in JPanel p1

ADD JCompletionDatefield in JPanel p1

ADD JNumberOfAssessmentsfield in JPanel p1

ADD Displaybtn in JPanel p1

ADD Clearbtn in JPanel p1

ADD NextPagebtn in JPanel p1

ADD Title2 in JPanel p2

ADD JCourseID1 in JPanel p2

ADD JCourseName1 in JPanel p2

ADD JDuration1 in JPanel p2

ADD JCourseID1field in JPanel p2

ADD JCourseName1field in JPanel p2

ADD JDuration1field in JPanel p2

ADD Add1btn in JPanel p2

ADD JCourseID3 in JPanel p2

ADD JInstructorName in JPanel p2

ADD JCourseLeader1 in JPanel p2

```
ADD JStartingDate1 in JPanel p2
ADD JCompletionDate1 in JPanel p2
ADD JExamDate in JPanel p2
ADD JPrerequisite in JPanel p2
ADD JCourseID4 in JPanel p2
ADD JCourseID3field in JPanel p2
ADD JCourseLeader1fieldin JPanel p2
ADD JStartingDate1field in JPanel p2
ADD JCompletionDate1field in JPanel p2
ADD JExamDatefield in JPanel p2
ADD JPrerequisitefield in JPanel p2
ADD JCourseID4field in JPanel p2
ADD Register1btn in JPanel p2
ADD Display1btnin JPanel p2
ADD Removebtn JPanel p2
ADD Clear1btn JPanel p2
ADD PreviousPagebtn JPanel p2

DO
    IF (CALL getSource() is Addbtn)
        ASSIGN String variable courseID as empty
        ASSIGN String variable course as empty
        ASSIGN int variable duration as null
        ASSIGN String variable level as empty
        ASSIGN int variable credit as null
        ASSIGN int variable assessments as null
        DO
        TRY
            ASSIGN variable courseID as value of JTextField
            JCourseIDfield
            ASSIGN variable course as value of JTextField
```

```
JCourseNamefield
ASSIGN variable duration as int value of JLevelfield
ASSIGN variable credit as int value of JTextField JCreditfield
ASSIGN variable assessments as int value of JNumberOfAssessmentsfield
ASSIGN Boolean variable added as false
DO
IF (courseID is empty)
        DISPLAY MesageDialog"Enter the courseID."

ELSE
        FOR (all obj of Course in ArrayList al)
                DO
                        IF (CALL getCourseID() is equal to courseID)
                        ASSIGN Boolean variable added as true
                ENDDO
        DO
        IF (added is false)
                CREATE new object C AcademicCourse (PASS parameters courseID, course, duration, level, credit, and assessments)
                ADD C in the ArrayList al
                DISPLAY MesageDialog"TheAcademic course has been added."
        ELSE
                DISPLAY MesageDialog"TheAcademic course is already added."
        ENDDO
ENDDO
ENDDO
```

```
            DO
            CATCH (ASSIGN NumberFormatException as e)
                DISPLAY MesageDialog"Please fill up the form properly!."
            ENDDO
    ENDDO

    DO
    IF (CALL getSource() is Registerbtn)
        ASSIGN String variable courseID as empty
        ASSIGN String variable courseLeader as empty
        ASSIGN String variable lecturerName as empty
        ASSIGN String variable startDate as empty
        ASSIGN String variable completionDate as empty
        DO
        TRY
            ASSIGN String variable courseID as value of JTextField
JCourseID3field
            ASSIGN String variable courseLeader as value of JTextField
            JCourseLeader1field
            ASSIGN String variable lecturerName as value of JTextField
            JlecturerNameField
            ASSIGN String variable startDate as value of JTextField
JStartingDate1field
            ASSIGN String variable completionDate as value of JTextField
            JCompletionDate1field
            ASSIGN Boolean CourseID Found as false
            FOR (all obj of Course in ArrayList List)
```

```
            DO
            IF (VALUE of method getCourseID() is equal to courseID)
                ASSIGN idFound as true
                IF (obj is instance of AcademicCourse)
                    DECLARE variable C of type
                    AcademicCourse, CAST obj to
                    AcademicCourse and STORE the value in C
                    IF (VALUE of method IsRegistered() equals to
                    true)
                        DISPLAY MesageDialog"The Course is
                        already registered."
                    ELSE
                        CALL method Register (PASS
                        parameters courseleader, lecturerName,
                        startDate, completionDate)
                        DISPLAY MesageDialog"The Course is
                        registered."
                ELSE
                    DISPLAY MesageDialog"The Course is not for
                    Academic Course."


            ELSE
                DISPLAY MesageDialog"The courseID is incorrect."
            ENDDO
        ENDFOR
    ENDDO
    DO
    CATCH (ASSIGN NumberFormatException as e)
        DISPLAY MesageDialog"Do properly."
    ENDDO
```

ENDDO

DO

IF (**CALL** getSource() is displaybtn)

FOR (all obj of Course in ArrayList al)

IF (obj is instance of AcademicCourse)

DECLARE variable objAC of type AcademicCourse, **CAST** obj to AcademicCourse and **STORE** the value in objAC

CALL method display () of AcademicCourse

ENDIF

ENDFOR

ENDDO

DO

IF (**CALL** getSource() is clearbtn)

CALL method setText(**PASS** parameter empty String) for JTextField txtid

CALL method setText(**PASS** parameter empty String) for JTextField txtcourse

CALL method setText(**PASS** parameter empty String) for JTextField txtduration

CALL method setText(**PASS** parameter empty String) for JTextField txtlevel

CALL method setText(**PASS** parameter empty String) for JTextField txtcredit

CALL method setText(**PASS** parameter empty String) for JTextField txtassessment

CALL method setText(**PASS** parameter empty String) for JTextField txtleader

CALL method setText(**PASS** parameter empty String) for JTextField txtlecturer

CALL method setText(**PASS** parameter empty String) for JTextField txtstartdate

CALL method setText(**PASS** parameter empty String) for JTextField txtcomdate

ENDDO

DO

IF (**CALL** method getSource() is Add1btn)

ASSIGN String variable courseID as empty

ASSIGN String variable course as empty

ASSIGN int variable duration as null

ASSIGN String variable prerequisite as empty

DO

TRY

ASSIGN variable courseID as value of JTextField txtnonid

ASSIGN variable course as value of JTextField txtnoncourse

ASSIGN variable duration as int value of JTextField txtnonduration

ASSIGN variable prerequisite as value of JTextField txtnonpre

ASSIGN Boolean variable added as false

DO

IF (courseID is empty)

DISPLAY MesageDialog"Enter the course ID."

ELSE

FOR (all obj of Course in ArrayList al)

DO

IF (CALL getCourseID() is equal to courseID)

```
                    ASSIGN Boolean variable added as true
                ENDDO
            DO
            IF (added is false)
                CREATE new object objAC AcademicCourse
                (PASS parameters courseID, course, duration,
                prerequisite)
                ADD objNAC in the ArrayList al
                DISPLAY MesageDialog"The course is
                added."
            ELSE
                DISPLAY MesageDialog"The course is already
                added."
            ENDDO
        ENDDO
    ENDDO
    DO
    CATCH (ASSIGN NumberFormatException as e)
        DISPLAY MesageDialog"Do properly."
    ENDDO
ENDDO



DO
IF (CALL getSource() is nonregbtn)
    ASSIGN String variable courseID as empty
    ASSIGN String variable courseLeader as empty
    ASSIGN String variable instructorName as empty
    ASSIGN String variable startDate as empty
    ASSIGN String variable examDate as empty
```

ASSIGN String variable completionDate as empty

DO

TRY

ASSIGN String variable courseID as value of JTextField txtnonid

ASSIGN String variable courseLeader as value of JTextField txtnonleader

ASSIGN String variable instructorName as value of JTextField txtinstructor

ASSIGN String variable startDate as value of JTextField txtnonstartdate

ASSIGN String variable examDate as value of JTextField txtnonexamdate

ASSIGN String variable completionDate as value of JTextField txtnoncomdate

ASSIGN Boolean idFound as false

FOR (all obj of Course in ArrayList al)

DO

IF (**VALUE** of method getCourseID() is equal to courseID)

ASSIGN idFound as true

IF (obj is instance of NonAcademicCourse)

DECLARE variable objNAC of type Non-AcademicCourse, **CAST** obj to Non-AcademicCourse and **STORE** the value in objNAC

IF (**VALUE** of method IsRegistered() equals to true)

DISPLAY MesageDialog “The Course is already registered.”

ELSE

CALL method Register (**PASS** parameters courseleader, lecturerName, startDate, examDate, completionDate)
DISPLAY MesageDialog “The Course is registered.”

ELSE
DISPLAY MesageDialog “The Course is not for Non-Academic Course.”

ELSE
DISPLAY MesageDialog “The course ID is incorrect.”
ENDDO
ENDDO
DO
CATCH (**ASSIGN** NumberFormatException as Z)
DISPLAY MesageDialog”Do properly.”
ENDDO
ENDDO

DO
IF (**CALL** method getSource() is removebtn)
DO
TRY
ASSIGN String variable courseID as value of JTextField txtnonid
FOR (all obj of Course in ArrayList al)
DO
IF (**VALUE** of method getCourseID() is equal to courseID)
IF (obj is instance of AcademicCourse)

```
                    DECLARE variable NonAcademic of type
                    AcademicCourse, CAST obj to AcademicCourse and
                    STORE the value in objAC

                    IF (VALUE of method IsRemoved() equals to false)
                        CALL method Remove () for objNonAcademic
                        DISPLAY MesageDialog "The Course is
                        removed."
                    ELSE
                        DISPLAY MesageDialog "The Course is
                        already removed."
            ELSE
                DISPLAY MesageDialog "The course ID is incorrect."
            ENDDO
        ENDDO
        DO
        CATCH (ASSIGN NumberFormatException as Z)
            DISPLAY MesageDialog"Do properly."
        ENDDO
ENDDO
```

```
DO
IF (CALL getSource() is nondisplaybtn)
    FOR (all obj of Course in ArrayList al)
        IF (obj is instance of NonAcademicCourse)
```

DECLARE variable objNAC of type NonAcademicCourse, **CAST** obj to NonAcademicCourse and **STORE** the value in objNAC

CALL method display () of NonAcademicCourse

ENDIF

ENDFOR

ENDDO

DO

IF (**CALL** getSource() is Clear1btn)

CALL method setText(**PASS** parameter empty String) for JTextField JCourseID1field

CALL method setText(**PASS** parameter empty String) for JTextField JCourseName1field

CALL method setText(**PASS** parameter empty String) for JTextField JDuration1field

CALL method setText(**PASS** parameter empty String) for JTextField JInstructorNamefield

CALL method setText(**PASS** parameter empty String) for JTextField JCourseLeader1field

CALL method setText(**PASS** parameter empty String) for JTextField JStartingDate1field

CALL method setText(**PASS** parameter empty String) for JTextField JCompletionDate1field

CALL method setText(**PASS** parameter empty String) for JTextField JExamDatefield

CALL method setText(**PASS** parameter empty String) for JTextField JPrerequisitefield

CALL method setText(**PASS** parameter empty String) for JTextField JCourseID3field

```
        CALL method setText(PASS parameter empty String) for JTextField
        JCourseID4field

    CREATE method actionPerformed(ActionEvent e)
    DO
        IF (e.getSource() is NextPagebtn)
            SET VISIBLE Panel p1 as false
            SET VISIBLE Panel p2 as true
            ADD JPanel p2 in JFrame frame
        ELSE IF (e.getSource() is PreviousPagebtn)
            SET VISIBLE Panel p1 as true
            SET VISIBLE Panel p2 as false
    ENDDO

ENDDO

DO
CREATE method static void main (PASS parameter String args[])
    CALL constructor INGCollege()
ENDDO
```

4. Method description

INGCollege

This method includes all GUI components such as JFrame, JPanel, JLabel, JBotton, and JTextField. CourseID, Course name, Duration, CourseLeader, Lecturer name, Level, Credit, Start date, Completion date, Number of assessments, Instructor name, Exam date, Prerequisites are the fields used in previous Coursework. The GUI layout is developed using JComponents in this way. First, the frame is set to the suitable size and location. The textfield is then used to create labels, and a button is added. This method is used to complete all of the frame work. This method also includes the addition of a panel.

Action performed:

This method is all about functionality of buttons which are add, register , clear , next page , previous page, display , remove buttons. When the buttons are pressed it performs certain actions according to the scenario .The following buttons perform the following function:

Add button

If the add button is clicked, data should be collected and stored in an array list, followed by a message indicating that your course has been added. . If the add button is clicked, data should be collected and stored in an array list, followed by a message indicating that your course has been added

Register button

When the register button is hit, the previously uploaded course should be registered . When the register button is hit, the previously uploaded course should be registered. When the show button is pushed, all of the course's details should be presented

Display button

When the display button is pushed, all of the course's details should be presented.

Remove button

When the remove button is pressed, the registered course should be removed and a message indicating that your course has been removed should be displayed,

Clear button

When the clear button is pushed, all text fields should be cleaned away.

5. Test

Test 1: Test that the program can be compiled and run using the command prompt

Test No	1
Objective :	To test the program if it can be compiled and run using the command Prompt.
Action:	→ First we open command prompt from the folder where our .java file is located →now compile the program by using javac java file name command. → Now execute the program by using java class name command
Expected Result:	Program should be compiled and executed .
Actual Result:	Program was compiled and executed
Conclusion:	The test is successful.

Table 1:Test table 1

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL\Desktop\cw2>
```

Figure 2: using java classname command to execute program

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL\Desktop\cw2>javac INGCollege.java

C:\Users\DELL\Desktop\cw2>
```

Figure 3: compiling program using javac .java file name command

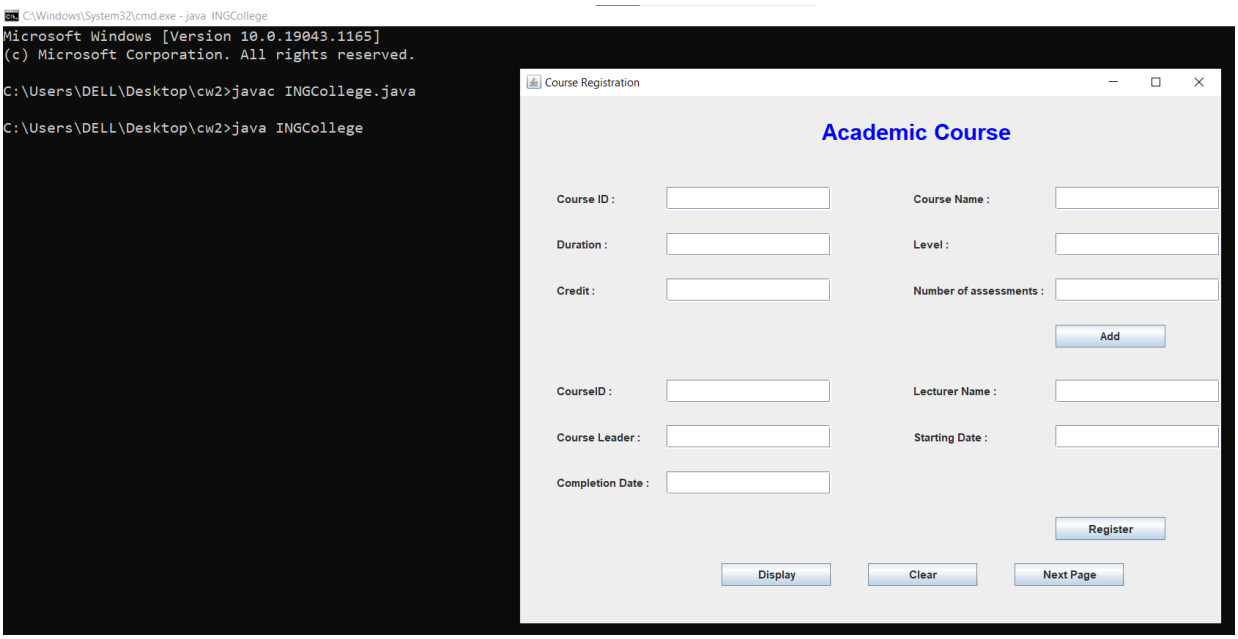

Figure 4. opening command prompt from the folder where our .java file is located

Test 2.1 : Add course for Academic Course and Non Academic Course

Test No	2.1
Objective :	To Add course for Academic Course and Non Academic Course.
Action:	→ First fill up CourseId, CourseName, Credit, duration, number of assesments and level in Academic Course and pressed add button similarly fill up courseId, CourseName, prerequisite and duration and press add button for adding course respectively → Now fill up courseID and click Remove button in NonAcademicCourse
Expected Result:	A dialog box should appear saying “The course has been added “ For both academic and non academic course
Actual Result:	A dialog box appeared saying “The course has been added “ For both academic and non academic course
Conclusion:	The test is successful.

Table 2 :Test table 2.1

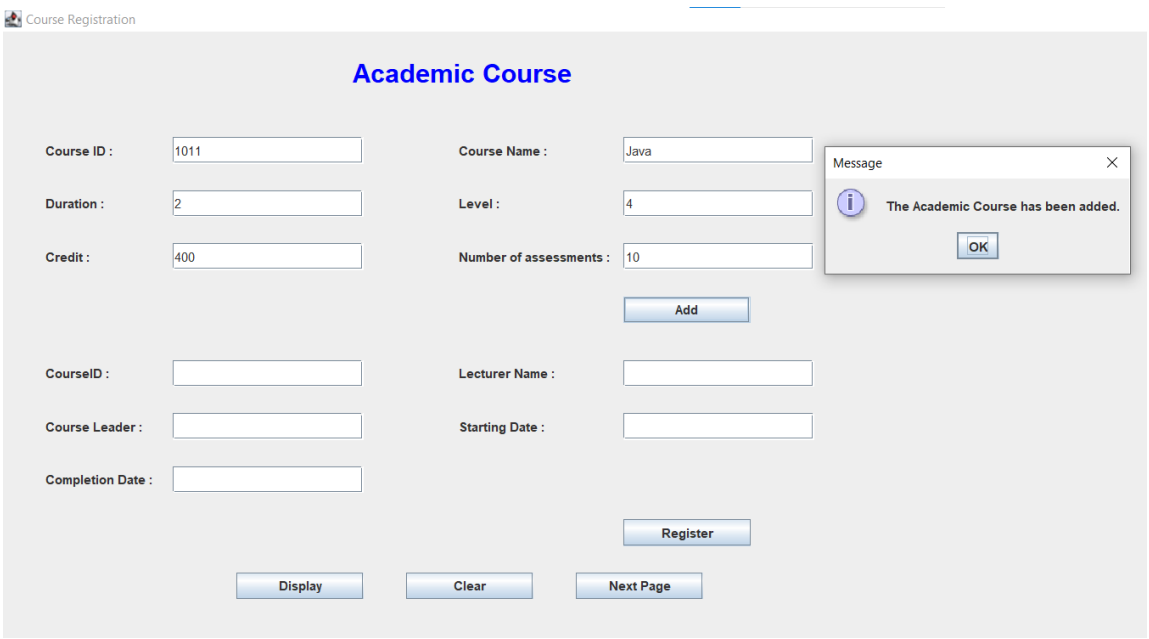

Figure 5: adding AcademicCourse

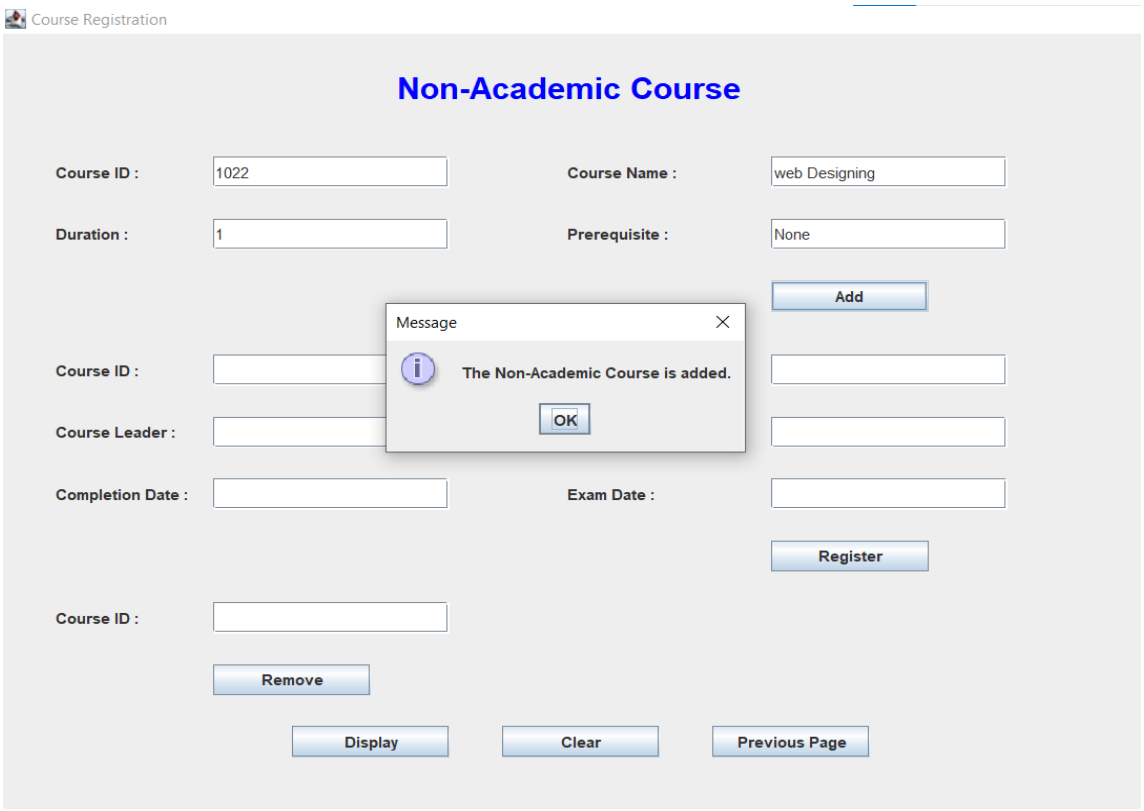

Figure 6 : adding NonAcademicCourse

Test 2.2 : To Register academic Course and non Academic course

Test	2.2
Objectives:	To Register academic Course and non Academic course
Action :	→ fill up CourseID, Lecturer name, Course leader, starting date Completion date in academic course. →fill up CourseID , Instructor name, Course leader, starting date Completion date , Exam date in Non academic course → Click the register button
Expected Result :	A dialog box appear saying “The course has been registered “ For both academic and non academic course
Actual Result :	A dialog box appeared saying “The course has been registered “ For both academic and non academic course
Conclusion :	The test is successful

Table 3 : test table 2.2

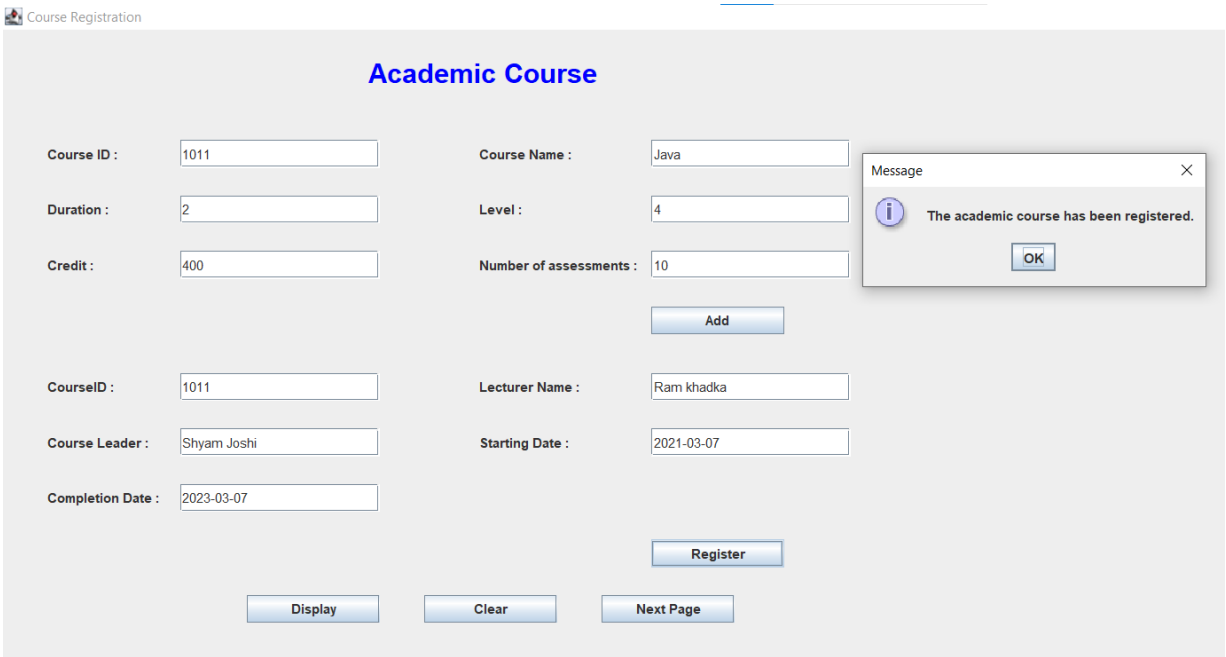

Figure 7 : Registering AcademicCourse

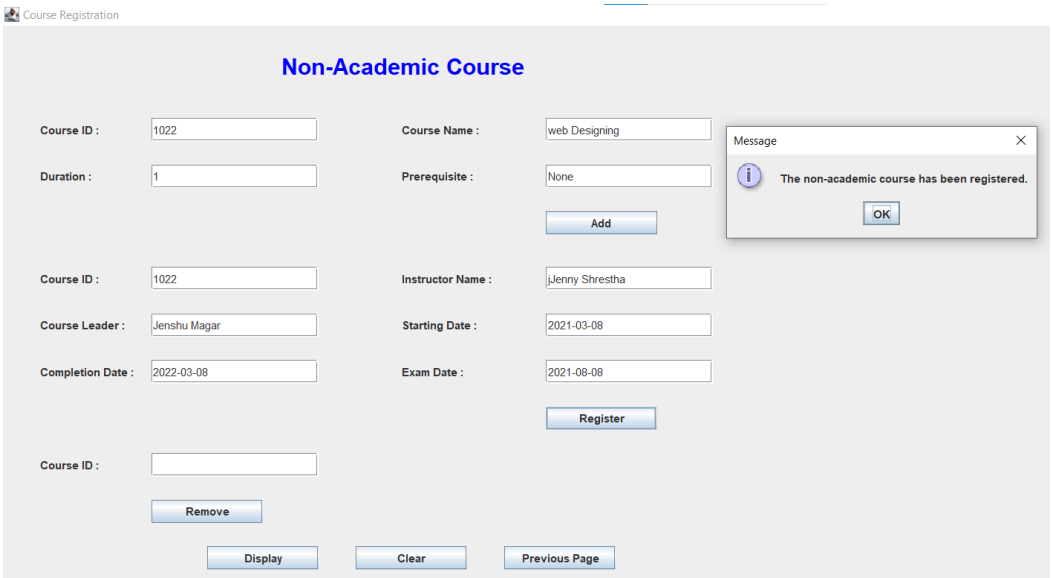

Figure 8 :Registering NonAcademicCourse

Test 2.3 :To remove non-academic course.

Test	2.3
Objectives:	To remove non-academic course.
Action :	→fill up CourseID in Non academic course → Click the Remove button
Expected Result :	A dialog box saying “The course has been removed “ Should appear in non academic course
Actual Result :	A dialog box appeared saying “The course has been removed “
Conclusion :	The test is successful

Table 4 : Test table 2.3

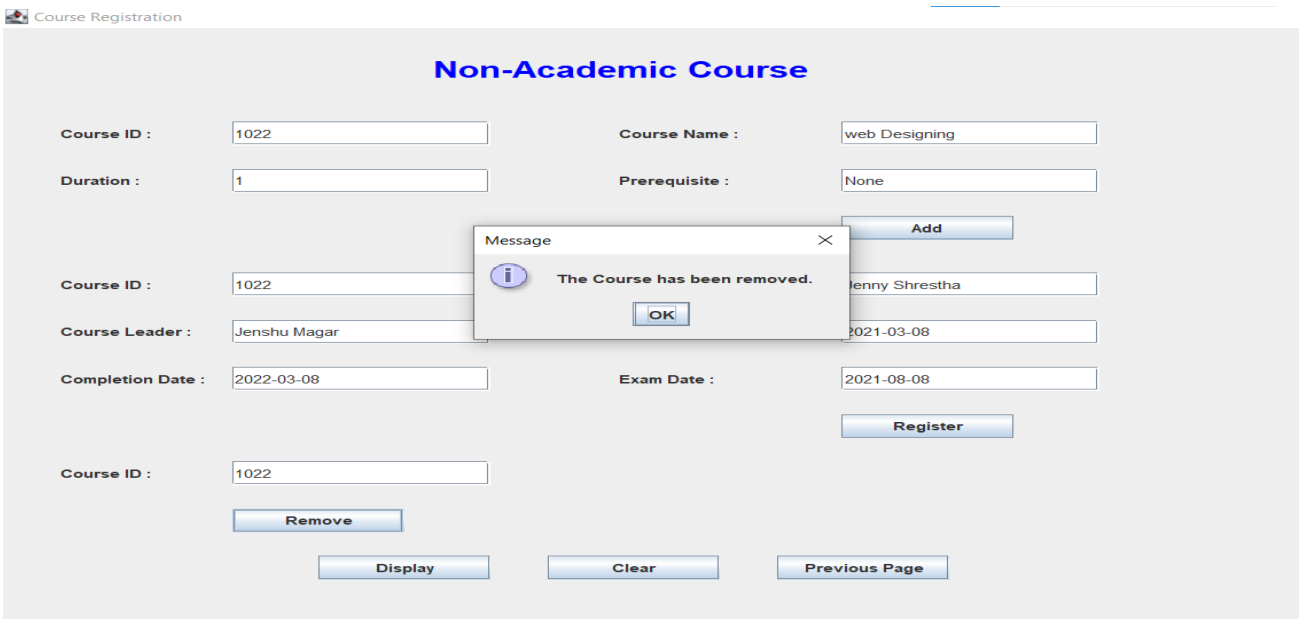

Figure 9 : Removing NonAcademicCourse

Test 3.1 To Check if appropriate dialog boxes appear when: Trying to add duplicate courseID,

Test No	3.1
Objective :	To Check if appropriate dialog boxes appear when: Trying to add duplicate courseID Trying to remove the non-academic course which is already removed..
Action:	→ Fill up duplicate CourseId, CourseName, Credit, duration, number of oursed nts and level in Academic Course and pressed add button similarly fill up duplicate course, CourseName, prerequisite and duration and press add button for adding course respectively
Expected Result:	Appropriate Dialog box should appear while trying to add duplicate courseID ,
Actual Result:	Appropriate Dialog box appeared trying to add duplicate courseID ,
Conclusion:	The test is successful.

Table 5 : Test table 3.1

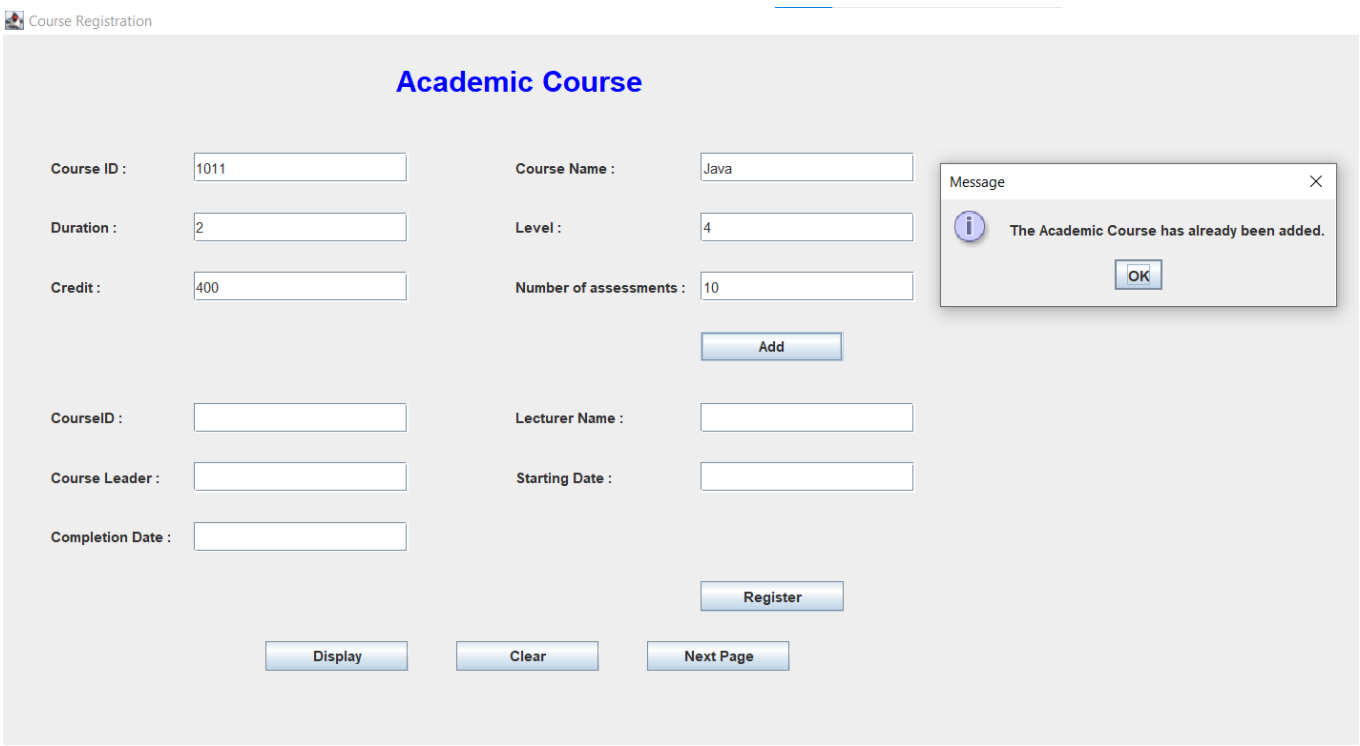

Figure 10 : Adding Duplicate courseID in Academic Course

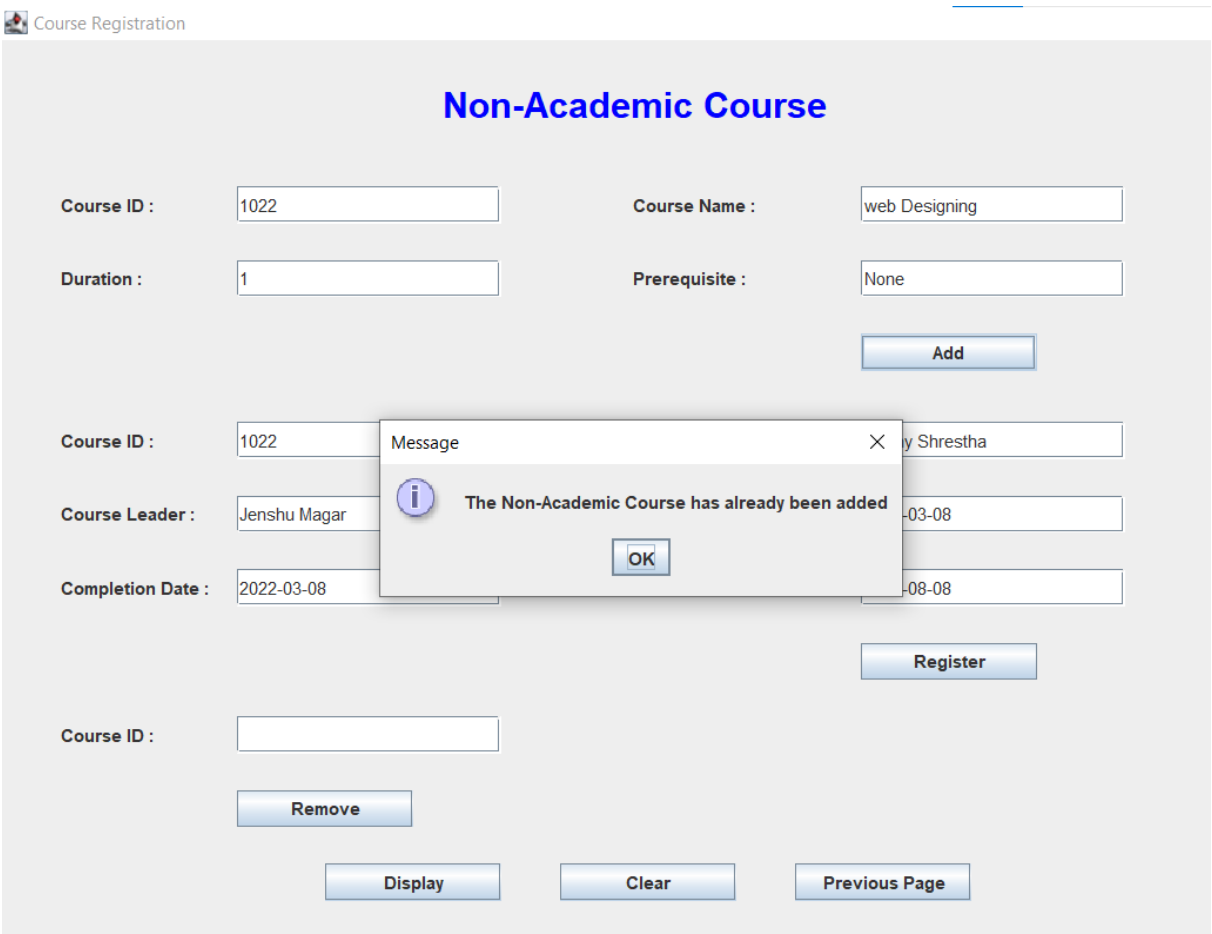

Figure 11 : Adding duplicate courseID in NonAcademicCourse

Test 3.2 : To Check if appropriate dialog boxes appear when:,Trying to register already registered course

Test	3.2
Objectives:	To Check if appropriate dialog boxes appear when:,Trying to register already registered course
Action :	→try to register the course that has already been registered
Expected Result :	Appropriate Dialog box should appear while trying to register already registered course
Actual Result :	Appropriate Dialog box appeared while trying to register already registered course
Conclusion :	The test is successful

Table 6: Test table 3.2

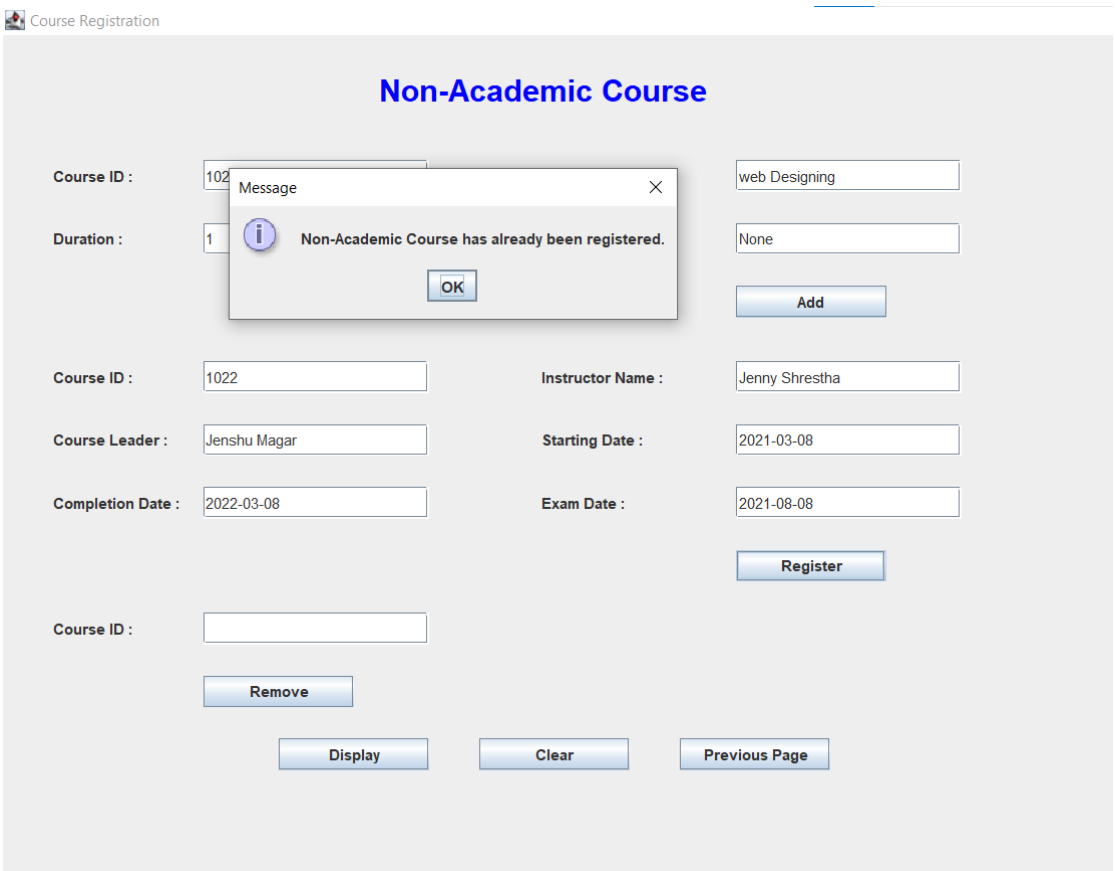

Figure 12 : Registering Already registered course in NonAcademicCourse

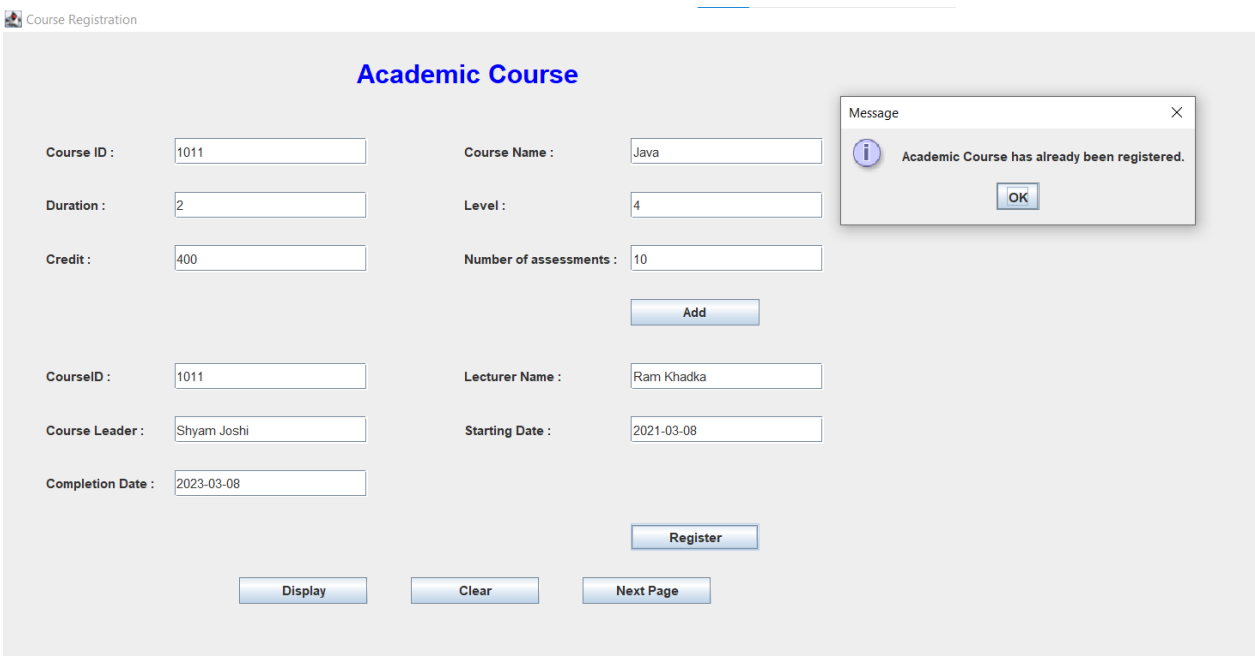

Figure 13 : Registering registered Course in AcademicCourse

Test 3.3 : To Check if appropriate dialog boxes appear when: trying to remove the non-academic course which is already removed.

Test	3.3
Objectives:	To Check if appropriate dialog boxes appear when:, : trying to remove the non-academic course which is already removed..
Action :	→: try to remove the non-academic course which is already removed.
Expected Result :	Appropriate Dialog box should appear while trying to remove the non-academic course which is already removed
Actual Result :	Appropriate Dialog box appeared while trying to remove the non-academic course which is already removed
Conclusion :	The test is successful

Table 7 Test table 3.3

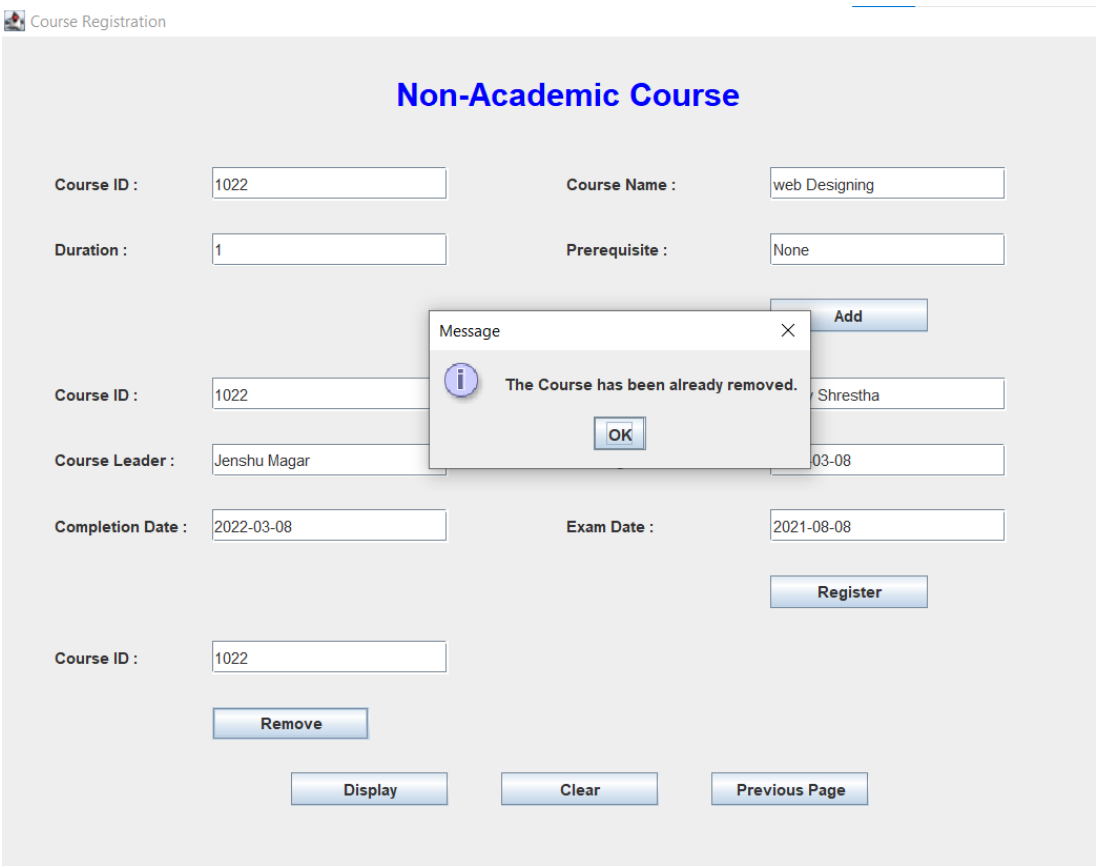

Figure 14 : Removing already removed course In NonAcademicCourse

6. Error

Error1 Syntax Error

Detection : The error was due to the missing of semi-column in JFrame variable , the error was corrected by adding semi column in its correct position

```
INGCollege - cw2
Class   Edit   Tools   Options
INGCollege ×
Compile   Undo   Cut   Copy   Paste   Find...   Close

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class INGCollege implements ActionListener //Implements ActionListener
{
    //instance reference variables
    private JFrame f
    private JPanel p
    private JLabel T                                  ';' expected                    ID, JCourseName, JDuration, JLevel, JCredit, JNumberOfAsse
            JLecturerName, JCourseLeader,JStartingDate, JCompletionDate, JCourseID1, JCourseName1, J
            JCourseID3, JInstructorName, JCourseLeader1, JStartingDate1, JCompletionDate1, JExamDate
    private JTextField JCourseIDfield, JCourseNamefield, JDurationfield, JCourseID2field, JLecturerN
            JLevelfield, JCreditfield, JStartingDatefield, JCompletionDatefield, JNumberOfAssessment
            JCourseName1field, JDuration1field, JPrerequisitefield, JCourseID3field,JInstructorNamef
            JStartingDate1field, JCompletionDate1field, JExamDatefield, JCourseID4field;
    private JButton Addbtn, Registerbtn, Displaybtn, Clearbtn, NextPagebtn, Add1btn, Register1btn, D

    private ArrayList<Course>list = new ArrayList<Course>(); //creating arraylist of Course class ty
    private AcademicCourse academic;
```

Figure 15: Syntax Error

```
INGCollege - cw2
Class   Edit   Tools   Options
INGCollege ×
Compile   Undo   Cut   Copy   Paste   Find...   Close

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class INGCollege implements ActionListener //Implements ActionListener
{
    //instance reference variables
    private JFrame f;
    private JPanel p1,p2;
    private JLabel Title1, Title2, JCourseID, JCourseName, JDuration, JLevel, JCredit, JNumberOfAssessments, JCourseI
            JLecturerName, JCourseLeader,JStartingDate, JCompletionDate, JCourseID1, JCourseName1, JDuration1, JPrere
            JCourseID3, JInstructorName, JCourseLeader1, JStartingDate1, JCompletionDate1, JExamDate, JCourseID4;
    private JTextField JCourseIDfield, JCourseNamefield, JDurationfield, JCourseID2field, JLecturerNamefield, JCourse
            JLevelfield, JCreditfield, JStartingDatefield, JCompletionDatefield, JNumberOfAssessmentsfield, JCourseID
            JCourseName1field, JDuration1field, JPrerequisitefield, JCourseID3field,JInstructorNamefield, JCourseLead
            JStartingDate1field, JCompletionDate1field, JExamDatefield, JCourseID4field;
    private JButton Addbtn, Registerbtn, Displaybtn, Clearbtn, NextPagebtn, Add1btn, Register1btn, Display1btn, Clear

    private ArrayList<Course>list = new ArrayList<Course>(); //creating arraylist of Course class type
    private AcademicCourse academic;
    private NonAcademicCourse Nonacademic;

    public INGCollege()
    {
```

Figure 16:Syntax error Correction

Error2: Simatic error

This error was occurred due to the incompatible of data type in the parameter while creating object called from Academic course. So , Correction was made using compatible data type in the parameter

```
INGCollege - cw2
Class  Edit  Tools  Options
INGCollege ×  AcademicCourse ×
Compile  Undo  Cut  Copy  Paste  Find...  Close
    {
        CourseID = JCourseIDfield.getText();
        CourseName = JCourseNamefield.getText();
        Duration = Integer.parseInt(JDurationfield.getText());
        Level = JLevelfield.getText();
        Credit = Integer.parseInt(JCreditfield.getText());
        Credit1 = Integer.parseInt(JCreditfield.getText());
        NumberOfAssessments = Integer.parseInt(JNumberOfAssessmentsfield.getText());
        boolean Sameacademic = false;
        for(Course C:list)
        {
            if (C.getCourseID().equals(CourseID))
            {
                Sameacademic = true;
            }
        }
        if(Sameacademic==false)
        {
            academic = new AcademicCourse(Duration, CourseID,CourseName,Level,Credit,NumberOfAssessments);
            list.add(academic);
            JOptionPane.showMessageDialog(f
            incompatible types: int cannot be converted to java.lang.String
        }
        else
        {
            JOptionPane.showMessageDialog(f,"The Academic Course has already been added.");
```

Figure 17: Semantic Error

```
INGCollege - cw2
Class  Edit  Tools  Options
INGCollege ×  AcademicCourse ×
Compile  Undo  Cut  Copy  Paste  Find...  Close
    {
        CourseID = JCourseIDfield.getText();
        CourseName = JCourseNamefield.getText();
        Duration = Integer.parseInt(JDurationfield.getText());
        Level = JLevelfield.getText();
        Credit = Integer.parseInt(JCreditfield.getText());
        Credit1 = Integer.parseInt(JCreditfield.getText());
        NumberOfAssessments = Integer.parseInt(JNumberOfAssessmentsfield.getText());
        boolean Sameacademic = false;
        for(Course C:list)
        {
            if (C.getCourseID().equals(CourseID))
            {
                Sameacademic = true;
            }
        }
        if(Sameacademic==false)
        {
            academic = new AcademicCourse(CourseID,CourseName,Duration,Level,Credit,NumberOfAssessments);
            list.add(academic);
            JOptionPane.showMessageDialog(f,"The Academic Course has been added.");
        }
        else
        {
            JOptionPane.showMessageDialog(f,"The Academic Course has already been added.");
Class compiled - no syntax errors
```

Figure 18 : : Semantic Error Correction

Error3 : Runtime error

The error was detected when the program was running. Here as you can see the value of courseID and Course name was interchanged.So, Correction was made after interchanging the CourseID and Coursename.

```
public class AcademicCourse extends Course
{
    //attributes of AcademicCourse
    private String LecturerName;
    private String Level;
    private int Credit;
    private String StartingDate;
    private String CompletionDate;
    private int NumberOfAssessments;
    private boolean IsRegistered;

    public AcademicCourse(String CourseName, String CourseId, int Duration, String Level, int Credit, int NumberOfAssessments)
    {
        super(CourseID, CourseName, Duration);
        this.Level=Level;
        this.Credit=Credit;
        this.NumberOfAssessments=NumberOfAssessments;
        LecturerName="";
        StartingDate="";
        CompletionDate="";
        IsRegistered=false;
```

Figure 19: Error in code

```
The courseID is Java
The coursename is 0110
The duration is 2 years
```

Figure 20: Runtime Error

```java
public class AcademicCourse extends Course
{
    //attributes of AcademicCourse
    private String LecturerName;
    private String Level;
    private int Credit;
    private String StartingDate;
    private String CompletionDate;
    private int NumberOfAssessments;
    private boolean IsRegistered;

    public AcademicCourse(String CourseID, String CourseName, int Duration, String Level, int Credit, int NumberOfAssessments)
    {
        super(CourseID, CourseName, Duration);
        this.Level=Level;
        this.Credit=Credit;
        this.NumberOfAssessments=NumberOfAssessments;
        LecturerName="";
```

Figure 21: Correction in code

```
The courseID is 0110
The coursename is Java
The duration is 2 years
```

Figure 22 : Run time Error Correction

7. Conclusion

This coursework was the coursework for the Programming module. . We were able to write a small program in the BlueJ programming language. We used all of the JAVA coding programming that we had learned from Online courses

During this course we learned a lot of things. We created a GUI layout is developed using JComponents . In this coursework all GUI components such as JFrame, JPanel, JLabel, JBotton, and JTextField. CourseID, Course name, Duration, CourseLeader, Lecturer name, Level, Credit, Start date, Completion date, Number of assessments, Instructor name, Exam date, Prerequisites are the fields used in previous Coursework. We also learned event handling.

We learn a lot terms used in the java programing language and also how to use it. A lot of dougths were clear throughout the course work by the guildlines given by the teachers and surfing the internet.

There were a lots of difficulties throught the course work. As this is the first course work for programming module it was confusing for me as I didn't know where to start. I also encountered a lot of errors while creating program in blue j. Syntax error and semantic errors were easy to find and solve but logical errors were very hard to find.

At last we were able to overcome all the difficulties we faced by interacting with our respected module teachers. Referencing the online classes that have been recorded and uploaded in google classroom. Self-Practice, planning ahead of the project and lots of lots of time was invested for the completion of this particular project. All in all this was an awesome learning experience

8.Appendix

Course Class

```
public class Course
   {
   // four attributes of Course
   private String CourseID;
   private String CourseName;
   private String CourseLeader;
   private int Duration;

   public Course( String CourseID, String CourseName, int Duration)
   {
      this.CourseID=CourseID;
      this.CourseName=CourseName;
      this.Duration=Duration;
      CourseLeader="";
   }
   //getter method
   public String getCourseID()
   {
      return CourseID;
   }

   public String getCourseName()
   {
      return CourseName;
   }

   public String getCourseLeader()
   {
      return CourseLeader;
   }

   public int getDuration()
   {
      return Duration;
   }
   //setter method
   public void setCourseLeader(String CourseLeader)
   {
      this.CourseLeader= CourseLeader;
   }
   //Display method
```

```
    public void display()
    {
        System.out.println("The courseID is "+CourseID);
        System.out.println("The coursename is "+CourseName);
        System.out.println("The duration is " +Duration + " years");
        //If statement
        if (CourseLeader!="")
        {
            System.out.println("The courseleader is " +CourseLeader);
        }
    }
}
```

Academic Course Class

```
public class AcademicCourse extends Course
{
    //attributes of AcademicCourse
    private String LecturerName;
    private String Level;
    private int Credit;
    private String StartingDate;
    private String CompletionDate;
    private int NumberOfAssessments;
    private boolean IsRegistered;

    public AcademicCourse(String CourseID, String CourseName, int Duration, String
Level, int Credit, int NumberOfAssessments)
    {
        super(CourseID, CourseName, Duration);
        this.Level=Level;
        this.Credit=Credit;
        this.NumberOfAssessments=NumberOfAssessments;
        LecturerName="";
        StartingDate="";
        CompletionDate="";
        IsRegistered=false;
    }
    //getter method for all attributes
    public String getLectureName()
    {
        return LecturerName;
    }

    public String getLevel()
```

```
    {
        return Level;
    }

    public int getCredit()
    {
        return Credit;
    }

    public String getStartingIDate()
    {
        return StartingDate;
    }

    public String getCompletionDate()
    {
        return CompletionDate;
    }

    public int getNumberOfAssessments()
    {
        return NumberOfAssessments;
    }

    public boolean getIsRegistered()
    {
        return IsRegistered;
    }
    //setter method
    public void setLecturerName(String LectureName)
    {
        this.LecturerName=LecturerName;
    }
    //setter method
    public void setNumberOfAssessment(int NumberOfAssessments)
    {
        this.NumberOfAssessments=NumberOfAssessments;
    }
    //method to register
    public void Register(String CourseLeader, String LecturerName, String StartingDate,
String CompletionDate)
    {
        if(IsRegistered==true)
        {
            System.out.println("The Academic Course is Registered.");
            System.out.println("Lecturer name is " +LecturerName);
```

```
            System.out.println("Starting date is " +StartingDate);
            System.out.println("Completion date is " +CompletionDate);
        }else{
            super.setCourseLeader(CourseLeader);//calls the super class method
            this.LecturerName=LecturerName;
            this.StartingDate=StartingDate;
            this.CompletionDate=CompletionDate;
            IsRegistered=true;
        }
    }
    //method to display
    public void display()
    {
        super.display();//calls the super class method
        if(IsRegistered==true)
        {
            System.out.println("The lecturer name is " +LecturerName);
            System.out.println("The level of this course is " +Level );
            System.out.println("The Credit of this course is " +Credit + " credits");
            System.out.println("The number of assessments is " +NumberOfAssessments);
            System.out.println("The starting date is "  +StartingDate);
            System.out.println("The completion date is "  +CompletionDate);
        }
    }
}
```

NonAcademicCourse Class

```
public class NonAcademicCourse extends Course
{
    //Attributes of NonAcademicCourse
    private String InstructorName;
    private String StartDate;
    private String CompletionDate;
    private String ExamDate;
    private String Prerequisite;
    private boolean IsRegistered;
    private boolean IsRemoved;

    public NonAcademicCourse (String CourseID, String CourseName, int Duration,
String Prerequisite )
```

```
{
   super (CourseID, CourseName, Duration);
   this.Prerequisite=Prerequisite;
   StartDate="";
   CompletionDate="";
   ExamDate="";
   IsRegistered=false;
   IsRemoved=false;
}
//getter method for all Attributes of NonAcademicCourse
public String getInstructorName()
{
   return InstructorName;
}

public String getStartingDate()
{
   return StartDate;
}

public String getCompletionDate()
{
   return CompletionDate;
}

public String getExamDate()
{
   return ExamDate;
}

public String getPrerequisite()
{
   return Prerequisite;
}

public boolean getIsRegistered()
{
   return IsRegistered;
}

public boolean getIsRemoved()
{
   return IsRemoved;
}
//setter method
public void setInstructorName (String InstructorName)
```

```
    {
        if(IsRegistered==false)
        {
            this.InstructorName=InstructorName;
        }else{
            System.out.println("The non academic course is already resistered.");
        }
    }
    //method to Register
    public void Register(String CourseLeader,String InstructorName, String StartDate,
String CompletionDate, String ExamDate)
    {
        if(IsRegistered==false)
        {
            setInstructorName(InstructorName);
            super.setCourseLeader(CourseLeader);//calls the super class method
            this.StartDate=StartDate;
            this.CompletionDate=CompletionDate;
            this.ExamDate=ExamDate;
            IsRegistered=true;
        }else{
            System.out.println("The Course is already registered.");
        }
    }
    //method to remove
    public void Remove()
    {
        if(IsRemoved==true)
        {
            System.out.println("The Course is removed.");
        }else{
            super.setCourseLeader("");//calls the super class method
            InstructorName="";
            StartDate="";
            CompletionDate="";
            ExamDate="";
            IsRegistered=false;
            IsRemoved=true;
        }
    }
    //method to display
    public void Display()
    {
        super.display();//calls the super class method
        if(IsRegistered==true)
        {
```

```
            System.out.println("The Instructor Name is " +InstructorName);
            System.out.println("The Start Date is " +StartDate);
            System.out.println("The Completion Date is " +CompletionDate);
            System.out.println("The Exam Date is " +ExamDate);
        }
    }
}
```

INGCollege Class

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

public class INGCollege implements ActionListener //Implements ActionListener
{
    //instance reference variables
    private JFrame f;
    private JPanel p1,p2;
    private JLabel Title1, Title2, JCourseID, JCourseName, JDuration, JLevel, JCredit,
JNumberOfAssessments, JCourseID2,
            JLecturerName, JCourseLeader,JStartingDate, JCompletionDate, JCourseID1,
JCourseName1, JDuration1, JPrerequisite,
            JCourseID3, JInstructorName, JCourseLeader1, JStartingDate1,
JCompletionDate1, JExamDate, JCourseID4;
    private JTextField JCourseIDfield, JCourseNamefield, JDurationfield,
JCourseID2field, JLecturerNamefield, JCourseLeaderfield,
            JLevelfield, JCreditfield, JStartingDatefield, JCompletionDatefield,
JNumberOfAssessmentsfield, JCourseID1field,
            JCourseName1field, JDuration1field, JPrerequisitefield,
JCourseID3field,JInstructorNamefield, JCourseLeader1field,
            JStartingDate1field, JCompletionDate1field, JExamDatefield, JCourseID4field;
    private JButton Addbtn, Registerbtn, Displaybtn, Clearbtn, NextPagebtn, Add1btn,
Register1btn, Display1btn, Clear1btn,
             Removebtn,PreviousPagebtn;

    private ArrayList<Course>list = new ArrayList<Course>(); //creating arraylist of
Course class type
    private AcademicCourse academic;
    private NonAcademicCourse Nonacademic;
```

```
public INGCollege()
{
    /**
     * For Frame
     */
    f = new JFrame("Course Registration");
    f.setVisible(true);// setting frame visible
    f.setLayout(null);// setting layout null
    f.setSize(860,640);
    /**
     * For panel p1 (Academic Course)
     */
    p1 = new JPanel();
    p1.setVisible(true);
    p1.setLayout(null);
    p1.setSize(860,640);
    /**
     * For panel p2 (Non Academic Course)
     */
    p2 = new JPanel();
    p2.setVisible(false);
    p2.setLayout(null);
    p2.setSize(860,640);
    /**
     * for Academic Course Title1
     */
    Title1 = new JLabel("Academic Course");
    Title1.setFont(new Font("Arial",Font.BOLD,25));//setting label font and size

    Title1.setBounds(330,15,400,50);// x axis , y axis , width, height
    Title1.setForeground(Color.BLUE);//setting text color
    /**
     * For NonAcademicCourse Title2
     */
    Title2 = new JLabel("Non-Academic Course");
    Title2.setFont(new Font("Arial",Font.BOLD,25));

    Title2.setBounds(300,20,400,50);
    Title2.setForeground(Color.BLUE);
    /**
     * For CourseID(AcademicCourse)
     */
    JCourseID = new JLabel("Course ID : ");
    JCourseID.setBounds(40,100,120,25);

    JCourseIDfield = new JTextField();
```

```
JCourseIDfield.setBounds(160,100,180,25);
/**
 * For CourseName(AcademicCourse)
 */
JCourseName = new JLabel("Course Name : ");
JCourseName.setBounds(430,100,120,25);

JCourseNamefield = new JTextField();
JCourseNamefield.setBounds(585,100,180,25);
/**
 * For Duration(AcademicCourse)
 */
JDuration = new JLabel("Duration : ");
JDuration.setBounds(40,150,120,25);

JDurationfield = new JTextField();
JDurationfield.setBounds(160,150,180,25);
/**
 * For Level(AcademicCourse)
 */
JLevel = new JLabel("Level : ");
JLevel.setBounds(430,150,120,25);

JLevelfield = new JTextField();
JLevelfield.setBounds(585,150,180,25);
/**
 * For Credit (AcademcCourse)
 */
JCredit = new JLabel("Credit : ");
JCredit.setBounds(40,200,120,25);

JCreditfield= new JTextField();
JCreditfield.setBounds(160,200,180,25);
/**
 * For NumberOfAssessments(AcademicCourse)
 */
JNumberOfAssessments = new JLabel("Number of assessments :");
JNumberOfAssessments.setBounds(430,200,150,25);

JNumberOfAssessmentsfield = new JTextField();
JNumberOfAssessmentsfield.setBounds(585,200,180,25);
/**
 * For CourseID1(NonAcademicCourse)
 */
JCourseID1 = new JLabel("Course ID : ");
JCourseID1.setBounds(40,100,120,25);
```

```
JCourseID1field = new JTextField();
JCourseID1field.setBounds(160,100,180,25);

/**
 * for CourseName1(NonAcademicCourse)
 */
JCourseName1 = new JLabel("Course Name : ");
JCourseName1.setBounds(430,100,120,25);

JCourseName1field = new JTextField();
JCourseName1field.setBounds(585,100,180,25);
/**
 * For Duration1(NonAcademicCourse)
 */
JDuration1 = new JLabel("Duration : ");
JDuration1.setBounds(40,150,120,25);

JDuration1field = new JTextField();
JDuration1field.setBounds(160,150,180,25);
/**
 * For Prerequisite(NonAcademicCourse)
 */
JPrerequisite = new JLabel("Prerequisite : ");
JPrerequisite.setBounds(430,150,120,25);

JPrerequisitefield = new JTextField();
JPrerequisitefield.setBounds(585,150,180,25);
/**
 * For CourseID2(AcademicCourse)
 */
JCourseID2 = new JLabel ("CourseID : ");
JCourseID2.setBounds(40,310,120,25);

JCourseID2field = new JTextField();
JCourseID2field.setBounds(160,310,180,25);
/**
 * For LectureName(AcademicCourse)
 */
JLecturerName = new JLabel("Lecturer Name : ");
JLecturerName.setBounds(430,310,120,25);

JLecturerNamefield = new JTextField();
JLecturerNamefield.setBounds(585,310,180,25);
/**
 * For CourseLeader(AcademicCourse)
```

```
 */
JCourseLeader = new JLabel("Course Leader : ");
JCourseLeader.setBounds(40,360,120,25);

JCourseLeaderfield = new JTextField();
JCourseLeaderfield.setBounds(160,360,180,25);
/**
 * For StartingDate(AcademicCourse)
 */
JStartingDate = new JLabel("Starting Date : ");
JStartingDate.setBounds(430,360,120,25);

JStartingDatefield = new JTextField();
JStartingDatefield.setBounds(585,360,180,25);
/**
 * For CompletionDate(AcademicCourse)
 */
JCompletionDate = new JLabel("Completion Date : ");
JCompletionDate.setBounds(40,410,120,25);

JCompletionDatefield = new JTextField();
JCompletionDatefield.setBounds(160,410,180,25);
/**
 * For JCourseID3(NonAcademicCourse)
 */
JCourseID3 = new JLabel("Course ID : ");
JCourseID3.setBounds(40,260,120,25);

JCourseID3field = new JTextField();
JCourseID3field.setBounds(160,260,180,25);
/**
 * For InstructorName(NonAcademicCourse)
 */
JInstructorName = new JLabel("Instructor Name : ");
JInstructorName.setBounds(430,260,120,25);

JInstructorNamefield = new JTextField();
JInstructorNamefield.setBounds(585,260,180,25);
/**
 * For CourseLeader1(NonAcademicCourse)
 */
JCourseLeader1 = new JLabel("Course Leader : ");
JCourseLeader1.setBounds(40,310,120,25);

JCourseLeader1field = new JTextField();
JCourseLeader1field.setBounds(160,310,180,25);
```

```
/**
 * For StartingDate1(NonAcademicCourse)
 */
JStartingDate1 = new JLabel("Starting Date : ");
JStartingDate1.setBounds(430,310,120,25);

JStartingDate1field = new JTextField();
JStartingDate1field.setBounds(585,310,180,25);
/**
 * For CompletionDate1(NonAcademicCourse)
 */
JCompletionDate1 = new JLabel("Completion Date : ");
JCompletionDate1.setBounds(40,360,120,25);

JCompletionDate1field = new JTextField();
JCompletionDate1field.setBounds(160,360,180,25);
/**
 * For ExamDate(NonAcademicCourse)
 */
JExamDate = new JLabel("Exam Date : ");
JExamDate.setBounds(430,360,120,25);

JExamDatefield = new JTextField();
JExamDatefield.setBounds(585,360,180,25);
/**
 * For CourseID4(NonAcademicCourse)
 */
JCourseID4 = new JLabel("Course ID : ");
JCourseID4.setBounds(40,460,120,25);

JCourseID4field = new JTextField();
JCourseID4field.setBounds(160,460,180,25);

/**
 * For buttons (AcdemicCourse)
 */
Addbtn = new JButton("Add");
Addbtn.setBounds(585,250,120,25);

Registerbtn = new JButton("Register");
Registerbtn.setBounds(585,460,120,25);

Displaybtn = new JButton("Display");
Displaybtn.setBounds(220,510,120,25);

Clearbtn = new JButton("Clear");
```

```
Clearbtn.setBounds(380,510,120,25);

NextPagebtn = new JButton("Next Page");
NextPagebtn.setBounds(540,510,120,25);

PreviousPagebtn = new JButton("Previous Page");
PreviousPagebtn.setBounds(540,560,120,25);
/**
 * For buttons (NonAcademicCourse)
 */
Add1btn = new JButton("Add");
Add1btn.setBounds(585,200,120,25);

Register1btn = new JButton("Register");
Register1btn.setBounds(585,410,120,25);

Display1btn = new JButton("Display");
Display1btn.setBounds(220,560,120,25);

Clear1btn = new JButton("Clear");
Clear1btn.setBounds(380,560,120,25);

Removebtn = new JButton("Remove");
Removebtn.setBounds(160,510,120,25);




/**
 * Adding labels in panel(AcademicCourse)
 */
p1.add(Title1);
p1.add(JCourseID);
p1.add(JCourseName);
p1.add(JDuration);
p1.add(JCourseIDfield);
p1.add(JCourseNamefield);
p1.add(JDurationfield);
p1.add(Addbtn);
p1.add(JCourseID2);
p1.add(JLecturerName);
p1.add(JCourseLeader);
p1.add(JLevel);
p1.add(JCredit);
p1.add(JStartingDate);
```

```
p1.add(JCompletionDate);
p1.add(JNumberOfAssessments);
p1.add(JCourseID2field);
p1.add(JLecturerNamefield);
p1.add(JCourseLeaderfield);
p1.add(JLevelfield);
p1.add(JCreditfield);
p1.add(JStartingDatefield);
p1.add(JCompletionDatefield);
p1.add(JNumberOfAssessmentsfield);
p1.add(Registerbtn);
p1.add(Displaybtn);
p1.add(Clearbtn);
p1.add(NextPagebtn);
/**
 * Adding labels in panel(NonAcademicCourse)
 */
p2.add(Title2);
p2.add(JCourseID1);
p2.add(JCourseName1);
p2.add(JDuration1);
p2.add(JCourseID1field);
p2.add(JCourseName1field);
p2.add(JDuration1field);
p2.add(Add1btn);
p2.add(JCourseID3);
p2.add(JInstructorName);
p2.add(JCourseLeader1);
p2.add(JStartingDate1);
p2.add(JCompletionDate1);
p2.add(JExamDate);
p2.add(JPrerequisite);
p2.add(JCourseID4);
p2.add(JCourseID3field);
p2.add(JInstructorNamefield);
p2.add(JCourseLeader1field);
p2.add(JStartingDate1field);
p2.add(JCompletionDate1field);
p2.add(JExamDatefield);
p2.add(JPrerequisitefield);
p2.add(JCourseID4field);
p2.add(Register1btn);
p2.add(Display1btn);
p2.add(Removebtn);
p2.add(Clear1btn);
p2.add(PreviousPagebtn);
```

```
        f.add(p1);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        /**
         * Adding action listener
         */
        Addbtn.addActionListener(this);
        Add1btn.addActionListener(this);
        Registerbtn.addActionListener(this);
        Register1btn.addActionListener(this);
        Displaybtn.addActionListener(this);
        Display1btn.addActionListener(this);
        Removebtn.addActionListener(this);
        Clearbtn.addActionListener(this);
        Clear1btn.addActionListener(this);
        NextPagebtn.addActionListener(this);
        PreviousPagebtn.addActionListener(this);
    }

    public void actionPerformed(ActionEvent A)
    {
        /**
         * For Addbtn
         */
        if(A.getSource()==Addbtn)
        {
            String CourseID="";
            String CourseName="";
            int Duration = 0;
            String Level = "";
            int Credit = 0;
            int Credit1 =Integer.valueOf(Credit);
            int NumberOfAssessments = 0;
            try
            {
                CourseID = JCourseIDfield.getText();
                CourseName = JCourseNamefield.getText();
                Duration = Integer.parseInt(JDurationfield.getText());
                Level = JLevelfield.getText();
                Credit = Integer.parseInt(JCreditfield.getText());
                Credit1 = Integer.parseInt(JCreditfield.getText());
                NumberOfAssessments =
Integer.parseInt(JNumberOfAssessmentsfield.getText());
                boolean Sameacademic = false;
                for(Course C:list)
```

```
            {
                if (C.getCourseID().equals(CourseID))
                {
                    Sameacademic = true;
                }
            }
            if(Sameacademic==false)
            {
                academic = new
AcademicCourse(CourseID,CourseName,Duration,Level,Credit,NumberOfAssessments
);
                list.add(academic);
                JOptionPane.showMessageDialog(f,"The Academic Course has been
added.");
            }
            else
            {
                JOptionPane.showMessageDialog(f,"The Academic Course has already
been added.");
            }
        }

        catch(Exception e)
        {
            JOptionPane.showMessageDialog(p1,"Please fill up the form properly !");
        }
    }
    else if (A.getSource()==Add1btn)
    {
        String CourseID = "";
        String CourseName = "";
        int Duration=0;
        String Prerequisite = "";
        try
        {
            CourseID = JCourseID1field.getText();
            CourseName = JCourseName1field.getText();
            Duration = Integer.parseInt(JDuration1field.getText());
            Prerequisite = JPrerequisitefield.getText();
            boolean SameNonacademic = false;
            for(Course C:list)
            {
                if(C.getCourseID().equals(CourseID))
                {
                    SameNonacademic = true;
                }
```

```
            }
            if(SameNonacademic == false)
            {
                Nonacademic = new NonAcademicCourse(CourseID, CourseName,
Duration, Prerequisite);
                list.add(Nonacademic);
                JOptionPane.showMessageDialog(f,"The Non-Academic Course is
added.");
            }
            else
            {
                JOptionPane.showMessageDialog(f,"The Non-Academic Course has
already been added");
            }
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(f,"Please fill up the form properly !");
        }
    }
    /**
     * for Register button
     */
    else if (A.getSource()==Registerbtn)
    {
        String CourseID = "";
        String CourseLeader = "";
        String LecturerName = "";
        String StartingDate = "";
        String CompletionDate = "";
        try
        {
            CourseID = JCourseID2field.getText();
            CourseLeader = JCourseLeaderfield.getText();
            LecturerName = JLecturerNamefield.getText();
            StartingDate = JStartingDatefield.getText();
            CompletionDate = JCompletionDatefield.getText();
            boolean Sameacademic1 = false;
            for(Course CO:list)
            {
                if(CO.getCourseID().equals(CourseID))
                {
                    Sameacademic1 = true;
                    if (CO instanceof AcademicCourse)
                    {
                        academic = (AcademicCourse) CO;
```

```
                    if (academic.getIsRegistered()==true)
                    {
                        JOptionPane.showMessageDialog(f,"Academic Course has already
been registered.");
                    }
                    else
                    {
                        academic.Register(CourseLeader, LecturerName, StartingDate,
CompletionDate);
                        JOptionPane.showMessageDialog(f,"The academic course has
been registered.");
                    }
                }
            }
            else
            {
                JOptionPane.showMessageDialog(f,"The CourseID do not match.");
                break;
            }
        }
    }

    catch (Exception E)
    {
        JOptionPane.showMessageDialog(p1,"Please fill up the forms properly !");
    }
}

else if (A.getSource()==Register1btn)
{
    String CourseID = "";
    String CourseLeader = "";
    String InstructorName = "";
    String StartingDate = "";
    String CompletionDate = "";
    String ExamDate = "";
    try
    {
        CourseID = JCourseID3field.getText();
        CourseLeader = JCourseLeader1field.getText();
        InstructorName = JInstructorNamefield.getText();
        StartingDate = JStartingDate1field.getText();
        CompletionDate = JCompletionDate1field.getText();
        ExamDate = JExamDatefield.getText();
        boolean SameNonacademic1 = false;
        for(Course CO:list)
```

```
            {
                if(CO.getCourseID().equals(CourseID))
                {
                    SameNonacademic1 = true;
                    if (CO instanceof NonAcademicCourse)
                    {
                        Nonacademic = (NonAcademicCourse)CO;
                        if (Nonacademic.getIsRegistered()==true)
                        {
                            JOptionPane.showMessageDialog(f,"Non-Academic Course has
already been registered.");
                        }
                        else
                        {
                            Nonacademic.Register(CourseLeader, InstructorName,
StartingDate, CompletionDate, ExamDate);
                            JOptionPane.showMessageDialog(f,"The non-academic course has
been registered.");
                        }
                    }
                }
                else
                {
                    JOptionPane.showMessageDialog(f,"The CourseID do not match.");
                    break;
                }
            }
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(f,"Please fill up the form properly !");
        }
    }
    /**
     * For Remove button
     */
    else if (A.getSource()==Removebtn)
    {
        String CourseID = JCourseID4field.getText();
        try{
            for(Course C:list){
                if(C.getCourseID().equals(CourseID))
                {
                    if(C instanceof NonAcademicCourse)
                    {
                        Nonacademic=(NonAcademicCourse)C;
```

```
                    if(Nonacademic.getIsRemoved()==false)
                    {
                        Nonacademic.Remove();
                        JOptionPane.showMessageDialog(f,"The Course has been
removed.");

                    }
                    else if(Nonacademic.getIsRemoved()==true)
                    {
                        JOptionPane.showMessageDialog(f,"The Course has been already
removed.");
                    }
                }
            }
            else
            {
                JOptionPane.showMessageDialog(f,"Enter valid CourseID");
                break;
            }
        }
    }
    catch (Exception e)
    {
        JOptionPane.showMessageDialog(f,"Please fill up the form properly !");
    }
}
/**
 * For display button
 */
else if (A.getSource()==Displaybtn)
{
    for(Course CO:list)
    {
        if(CO instanceof AcademicCourse)
        {
            AcademicCourse academic = (AcademicCourse)CO;

            academic.display();
        }
    }
}
else if (A.getSource()==Display1btn)
{
    for(Course CO:list)
    {
        if(CO instanceof NonAcademicCourse)
```

```
        {
            NonAcademicCourse Nonacademic = (NonAcademicCourse)CO;

            Nonacademic.Display();
        }
    }
}
/**
 * for Clear button
 */
else if (A.getSource()==Clearbtn)
{
    JCourseIDfield.setText("");
    JCourseNamefield.setText("");
    JDurationfield.setText("");
    JLecturerNamefield.setText("");
    JCourseLeaderfield.setText("");
    JLevelfield.setText("");
    JCreditfield.setText("");
    JStartingDatefield.setText("");
    JCompletionDatefield.setText("");
    JNumberOfAssessmentsfield.setText("");
    JCourseID2field.setText("");
}
else if (A.getSource()==Clear1btn)
{
    JCourseID1field.setText("");
    JCourseName1field.setText("");
    JDuration1field.setText("");
    JInstructorNamefield.setText("");
    JCourseLeader1field.setText("");
    JStartingDate1field.setText("");
    JCompletionDate1field.setText("");
    JExamDatefield.setText("");
    JPrerequisitefield.setText("");
    JCourseID3field.setText("");
    JCourseID4field.setText("");
}
/**
 * for next page Button
 */
else if(A.getSource()==NextPagebtn)
{
    p1.setVisible(false);
    p2.setVisible(true);
    f.add(p2);
```

```
        }
        /**
         * for previous page button
         */
        else if(A.getSource()==PreviousPagebtn)
        {
            p2.setVisible(false);
            p1.setVisible(true);
        }
    }

    public static void main(String[]args)
    {
        new INGCollege();
    }
}
```