

**CS4001NI PROGRAMMING**

**Assessment Weightage & Type**

**20% Individual Coursework**

**Year and Semester**

**2018-19 Autumn**

**Student Name: Krishna Budhathoki**

**London Met ID:** **18029976**

**College ID:** **np01cp4a180172**

**Assignment Due Date: 4/19/2019**

**Assignment Submission Date: 4/19/2019 Word Count (Where Required): 3434**

*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 submissions will be treated as non-submission and a marks of zero will be awarded.*

Contents

[Introduction 4](#_Toc6570986)

[Class diagram 5](#_Toc6570987)

[Method description 7](#_Toc6570988)

[Pseudo code 8](#_Toc6570989)

[Test 25](#_Toc6570990)

[Test 1 25](#_Toc6570991)

[Test 2 25](#_Toc6570992)

[Test 3 26](#_Toc6570993)

[Test 4 28](#_Toc6570994)

[Test 5 29](#_Toc6570995)

[Test 6 30](#_Toc6570996)

[Test 7 31](#_Toc6570997)

[Test 8 32](#_Toc6570998)

[Error 34](#_Toc6570999)

[Conclusion 38](#_Toc6571000)

[Bibliography 38](#_Toc6571001)

[Appendix 39](#_Toc6571002)

Table of figures

[Figure 1 test 1 25](#_Toc6571025)

[Figure 2 test 2 26](#_Toc6571026)

[Figure 3 test 3 27](#_Toc6571027)

[Figure 4 test 4.1 28](#_Toc6571028)

[Figure 5 test 4.2 28](#_Toc6571029)

[Figure 6 test 5 29](#_Toc6571030)

[Figure 7 test 6.1 30](#_Toc6571031)

[Figure 8 test 6.2 31](#_Toc6571032)

[Figure 9 test 7 31](#_Toc6571033)

[Figure 10 test 8.1 32](#_Toc6571034)

[Figure 11 test 8.2 33](#_Toc6571035)

[Figure 12 error 1 34](file:///C:\Users\DELL\Documents\coursework2.docx#_Toc6571036)

[Figure 13 debugged 1 35](file:///C:\Users\DELL\Documents\coursework2.docx#_Toc6571037)

[Figure 14 error 2 36](#_Toc6571038)

[Figure 15 debugged 2 36](file:///C:\Users\DELL\Documents\coursework2.docx#_Toc6571039)

[Figure 16 error 3 37](#_Toc6571040)

[Figure 17 debugged 3 38](file:///C:\Users\DELL\Documents\coursework2.docx#_Toc6571041)

# Introduction

The coursework is given under the module programming. This coursework contains 20 percent of total module grade. The main objective of this coursework is design GUI in java using swing and awt. This GUI was developed in blueJ using code instead of drag and drop which was very challenging. It stands for Graphical User Interface. The code for GUI was used for understanding the concept of the objects, methods, constructors, inheritance and to get use to the format of the codes for GUI. The GUI is easy to access and interact by using buttons, labels, and text panels. This coursework includes class diagram, method description, pseudo code and testing of developed program.

GUI is used in each and every programming language. GUI presents easy to use visual display to the user. To develop GUI we either use swing or javaFX. The given coursework was assigned for connecting previous coursework with GUI which we developed at first. Then we used array list to store user inputs. We used RigoTechnology class to make GUI and use array list to store data. We linked previous coursework in the rigoTechnology class to perform the action listener. (java, 2016)

# Class diagram

|  |
| --- |
| RigoTechnology |
| +lblHead1  +lblPlatformS  +lblInterviewerS  +lblWorkingHoursS  +lblSalaryS  +lblContractPeriod  +lblSeDeveloper  +lblJoiningDate  +lblAdvanceSalary  +lblStaffRoomNo  +lblSDeveloperNo  +txtPlatformS  +txtInterviewerS  +txtWorkingHoursS  +txtSalaryS  +txtContractPeriod  +txtSeDeveloper  +txtJoiningDate  +txtAdvanceSalary  +txtStaffRoomNo  +txtSDeveloperNo  +lblHead2  +lblPlatformJ  +lblInterviewerJ  +lblWorkingHoursJ  +lblSalaryJ  +lblAppointedBy  +lblTerminationDate  +lblDeveloperName  +lblAppointedDate  +lblSpecialization  +lblJDeveloperNo  +txtPlatformJ  +txtInterviewerJ  +txtWorkingHoursJ  +txtSalaryJ  +txtAppointedBy  +txtTerminationDate  +txtDeveloperName  +txtAppointedDate  +txtSpecialization  +txtJDeveloperNo  +btnAddS  +btnHireS  +btnTerminateS  +btnAddJ  +btnAppointJ  +btnDisplay  +btnClear  +ArrayList<Developer>developer |
| +main (String [] args): void  +gui(): void  + actionPerformed(ActionEvent ae): void  +addSeniorDeveloper(): void  +HireDeveloper(): void  +terminated(): void  +addJuniorDeveloper():void  +AppointedJunior(): void  +Display(): void  +Clear(): void |

# Method description

1. public void gui()

It is a public method which can be accessed by anyone and this method is used to create GUI interface

1. public void actionPerformed(ActionEvent ae)

It is a public method which can be accessed by anyone and this method is used to give function for GUI interfaces.

1. public void addSeniorDeveloper()

It is a public method which can be accessed by anyone and this method is used to add platform, interviewer name, working hours, salary and contract period for appointing.

1. public void HireDeveloper()

It is a public method which can be accessed by anyone and this method is used to hire senior developer.

1. public void terminated()

It is a public method which can be accessed by anyone and this method is used to terminate the hired senior developer.

1. public void addJuniorDeveloper()

It is a public method which can be accessed by anyone and this method is used to add platform, interviewer name, working hour, salary, appointed by for appointing junior developer.

1. public void AppointedJunior()

It is a public method which can be accessed by anyone and this method is used to appoint senior developer.

1. public void Display()

It is a public method which can be accessed by anyone and this method is used to display all the information.

1. public void Clear()

It is a public method which can be accessed by anyone and this method is used to clear text fields in the GUI.

# Pseudo code

Define class RigoTechnology implements ActionListener

Create jframe

Create jlabel

Create jtext

Define RigoTechnology()

Call method of arraylist in RigoTechnology

Main method ()

Call the RigoTechnology

Define gui()

Do

Create jframe rigoTechnology;

Set layout

Set bounds

Set border

Add to the main frame

Create jlabel lblPlatformS

Set text "Platform:"

Set bounds

Add to frame

Create jtextfield txtPlatformS

Set bounds

Add to frame

Create jlabel lblInterviewerS

Set text "Interviewer Name:"

Set bounds

Add to frame

Create jtextfield txtInterviewerS

Set bounds

Add to frame

Create jlabel lblWorkingHoursS

Set text "Working Hours:"

Set bounds

Add to frame

Create jtextfield txtWorkingHoursS

Set bounds

Add to frame

Create jlabel lblSalaryS

Set text "Salary:"

Set bounds

Add to frame

Create jtextfield txtSalaryS

Set bounds

Add to frame

Create jlabel lblContractPeriod

Set text "Contract Period:"

Set bounds

Add to frame

Create jtextfield txtContractPeriod

Set bounds

Add to frame

Create jbutton btnAddS

Set bounds

Set text “Add"

Add to frame

Registering instance of event handler class

Create jlabel lblSeDeveloper

Set text "Developer's Name:"

Set bounds

Add to frame

Create jtextfield txtSeDeveloper

Set bounds

Add to frame

Create jlabel lblAdvanceSalary

Set text “advance salary:”

Set bounds

Add to frame

Create jtextfield txtAdvanceSalary

Set bounds

Add to frame

Create jlabel lblJoiningDate

Set text “joining date:”

Set bounds

Add to frame

Create jtextfield txtJoiningDate

Set bounds

Add to frame

Create jlabel lblStaffRoomNo

Set text “staff room no:”

Set bounds

Add to frame

Create jtextfield txtStaffRoomNo

Set bounds

Add to frame

Create jlabel lblSDeveloperNo

Set text “developer no:”

Set bounds

Add to frame

Create jtextfield txtSDeveloperNo

Set bounds

Add to frame

Create jbutton btnHireS

Set text “Hire"

Set bounds

Add to frame

Registering instance of event handler class

Create jbutton btnTerminateS

Set text “terminate”

Set bounds

Add to frame

Registering instance of event handler class

Create jlabel lblHead2

Set text "For Junior Developer"

Set bounds

Set color

Add to frame

Create jlabel lblPlatformJ

Set text "Platform:"

Set bounds

Add to frame

Create jtextfield txtPlatformJ

Set bounds

Add to frame

Create jlabel lblInterviewerJ

Set text "Interviewer Name:"

Set bounds

Add to frame

Create jtextfield txtInterviewerJ

Set bounds

Add to frame

Create jlabel lblWorkingHoursJ

Set text "Working Hours:"

Set bounds

Add to frame

Create jtextfield txtWorkingHoursJ

Set bounds

Add to frame

Create jlabel lblSalaryJ

Set text "Salary:"

Set bounds

Add to frame

Create jtextfield txtSalaryJ

Set bounds

Add to frame

Create jlabel lblAppointedBy

Set text "Appointed By:"

Set bounds

Add to frame

Create jtextfield txtAppointedBy

Set bounds

Add to frame

Create jbutton btnAddJ

Set text "Add"

Set bounds

Add to frame

Registering instance of event handler class

Create jlabel lblDeveloperName

Set text “developer name:”

Set bounds

Add to frame

Create jtextfield txtDeveloperName

Set bounds

Add to frame

Create jlabel lblAppointedDate

Set text “appointed date:”

Set bounds

Add to frame

Create jtextfield txtAppointedDate

Set bounds

Add to frame

Create jlabel lblTerminationDate

Set text “termination date:”

Set bounds

Add to frame

Create jtextfield txtTerminationDate

Set bounds

Add to frame

Create jlabel lblJDeveloperNo

Set text “developer no:”

Set bounds

Add to frame

Create jtextfield txtJDeveloperNo

Set bounds

Add to frame

Create jlabel lblSpecialization

Set text “Specialization:"

Set bounds

Add to frame

Create jtextfield txtSpecialization

Set bounds

Add to frame

Create jbutton btnAppointJ

Set text “appoint”

Set bounds

Add to frame

Registering instance of event handler class

Create jbutton btnDisplay

Set text ”display”

Set bounds

Add to frame

Registering instance of event handler class

Create jbutton btnClear

Set text “clear”

Set bounds

Add to frame

Registering instance of event handler class

Define actionPerformed(ActionEvent ae)

If variable is equal to btnAddS

Add senior developer method

End if

If variable is equal to btnHireS

Add hire senior developer method

End if

If variable is equal to btnTerminateS

Add terminate method

End if

If variable is equal to btnAddJ

Add junior developer method

Endif

If variable is equal to btnAppointJ

Add appoint junior method

Endif

If variable is equal to btnDisplay

Add display method

Endif

If variable is equal to btnClear

Add clear method

Endif

Call addSeniorDeveloper()

Do

Try

Do

txtPlatformS.getText() is stored as variable in platform

txtInterviewerS.getText() is stored as variable in interviewer

txtWorkingHoursS.getText() is stored as variable in working hours

txtSalaryS.getText() is stored as variable in salary

txtContractPeriod.getText() is stored as variable in contract period

Try

Do

txtWorkingHoursS.getText() is converted into integer

txtContractPeriod.getText() is converted into integer

txtSalaryS.getText() is converted into integer

if platform is empty and interviewer is empty and working hours is empty and salary is empty and contract period is empty

showMessageDialog(appropriate message)

else

SeniorDeveloper hire=new SeniorDeveloper(platform,interviewer,workinghours,contractPeriod,salary)

Add hire in developer

Display appropriate message

End if

Catch nullpointer exception

Appropriate message

Catch numberformat exception

Appropriate message

Catch number format exception nfe

Appropriate message

Call hireDeveloper()

Do

Try

txtSeDeveloper.getText() is stored as variable in name

txtJoiningDate.getText() is stored as variable in joining date

txtStaffRoomNo.getText() is stored as variable in staff room no

txtAdvanceSalary.getText() is converted into integer as advance salary

txtSDeveloperNo.getText() is converted into integer

as developer no

if developer size < dev no

display appropriate message

else

name is empty and joining date is empty and staffRoomNo is empty

display appropriate message

end if

if developer.get(devNo-1) instanceof JuniorDeveloper

JuniorDeveloper objJ = (JuniorDeveloper)developer.get(devNo-1)

If isjoined is equal to false

appointDeveloper(name,appointedDate, terminationDate,specialization)

display appropriate message

else

display appropriate message

endif

else

display appropriate message

endif

catch exception

display appropriate message

call terminated()

do

try

txtSDeveloperNo.getText() is converted into integer as developer no

if developer.size()<developerNo

display appropriate message

else

if developer.get(developerNo-1) instanceof SeniorDeveloper

SeniorDeveloper objS = (SeniorDeveloper)developer.get(developerNo-1)

If isappointed is equal to true

If isterminated is equal to false

objS.contractTerminated()

display appropriate message

else

display appropriate message

endif

endif

else

display appropriate message

endif

catch exception

display appropriate message

call addJuniorDeveloper()

do

try

txtPlatformJ.getText() is stored as variable in platform

txtInterviewerJ.getText() is stored as variable in interviewer

txtWorkingHoursJ.getText() is stored as variable in working hours

txtSalaryJ.getText() is stored as variable in salary

txtAppointedBy.getText() is stored as variable in appointed by

txtTerminationDate.getText() is stored as variable in termination date

try

txtWorkingHoursJ.getText() is converted into integer as working hours

txtSalaryJ.getText() is converted into double as salary

if

platform and interviewer and working hours and salary and appointed by is empty

display appropriate message

else

JuniorDeveloper hireJ= new JuniorDeveloper(PlatformJ,InterviewerJ,WorkingHours,SalaryJe,AppointedBy,terminationDate)

Add hirej in developer

Display appropriate message

Catch exception np

Display appropriate message

Catch exception nf

Display appropriate message

Catch exception nfe

Display appropriate message

Endif

Call AppointedJunior()

Do

Try

txtDeveloperName.getText() is stored as variable in developer name

txtAppointedDate.getText() is stored as variable in appointed date

txtTerminationDate.getText() is stored as variable in termination date

txtSpecialization.getText() is stored as variable in specialization

txtJDeveloperNo.getText() is converted into integer as developer no

if developer.size() < devNo

display appropriate message

else

if name and appointed date and termination date and specialization

display appropriate message

else

if developer.get(devNo-1) instanceof JuniorDeveloper

JuniorDeveloper objJ = (JuniorDeveloper)developer.get(devNo-1)

If isjoined is equal to false

appointDeveloper(name,appointedDate, terminationDate,specialization)

display appropriate message

else

display appropriate message

endif

else

display appropriate message

endif

else

display appropriate message

endif

catch exception e

display appropriate message

call Display()

do

try

if (Developer d: developer)

if(d instanceof SeniorDeveloper)

display appropriate message

endif

if(d instanceof JuniorDeveloper)

display appropriate message

endif

call Clear()

do

try

txtPlatformJ.setText is made empty

txtInterviewerJ.setText is made empty

txtWorkingHoursJ.setText is made empty

txtSalaryJ.setText is made empty

txtAppointedBy.setText is made empty

txtTerminationDate.setText is made empty

txtDeveloperName.setText is made empty

txtAppointedDate.setText is made empty

txtSpecialization.setText is made empty

txtJDeveloperNo.setText is made empty

txtPlatformS.setText is made empty

txtInterviewerS.setText is made empty

txtWorkingHoursS.setText is made empty

txtSalaryS.setText is made empty

txtContractPeriod.setText is made empty

txtSeDeveloper.setText is made empty

txtJoiningDate.setText is made empty

txtAdvanceSalary.setText is made empty

txtStaffRoomNo.setText is made empty

txtSDeveloperNo.setText is made empty

# Test

## Test 1

Objective: Compile program using command prompt

Action Type: javac (filename).java by giving file location

Expected Result: Program to be compiled and GUI being displayed

Actual: Result Program is compiled and GUI is displayed

Conclusion : Test Successful

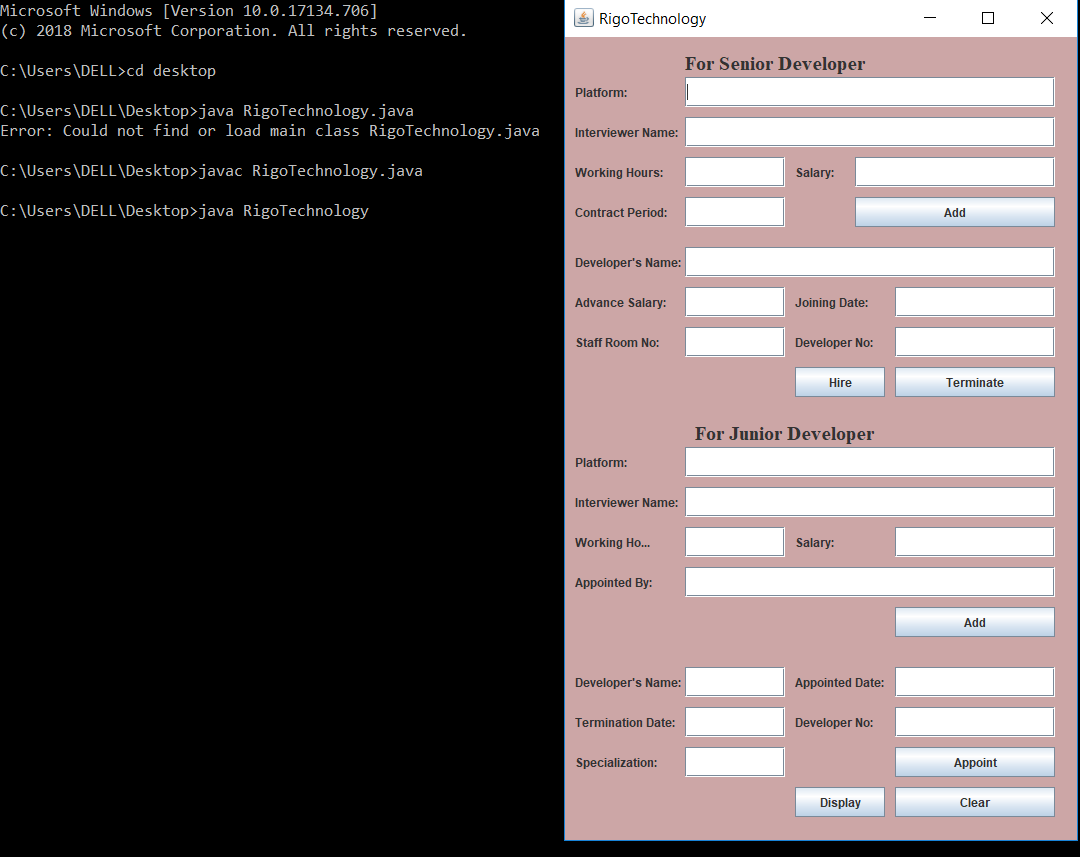


Figure 1 test 1

## Test 2

Objective: add the information of senior developer

Action Type : add the platform, interviewer name, working hours, salary, contract period

Expected Result : information added for senior developer

Actual Result : information added for senior developer

Conclusion Test Successful

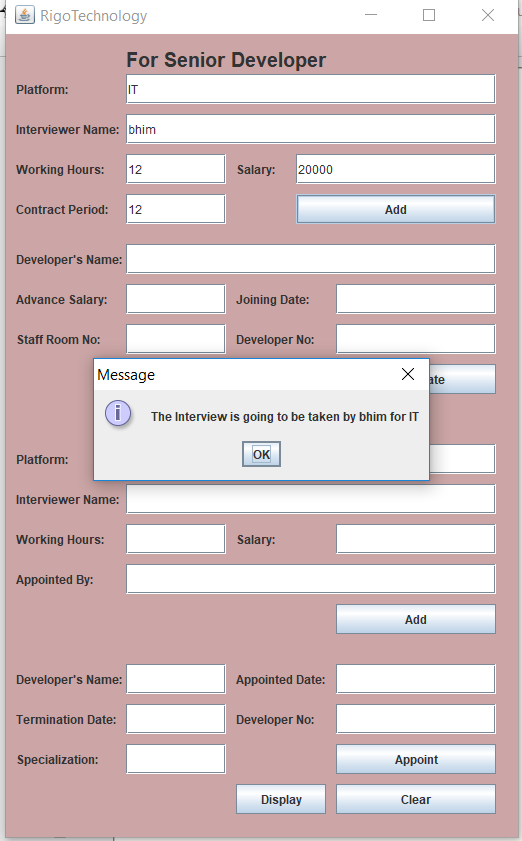


Figure 2 test 2

## Test 3

Objective: add the information to appoint senior developer

Action Type : add developer name, advance salary, joining date, staff room no

,developer no

Expected Result : information added to appoint senior developer

Actual Result : information is stored for appointing the senior developer

Conclusion: test successful

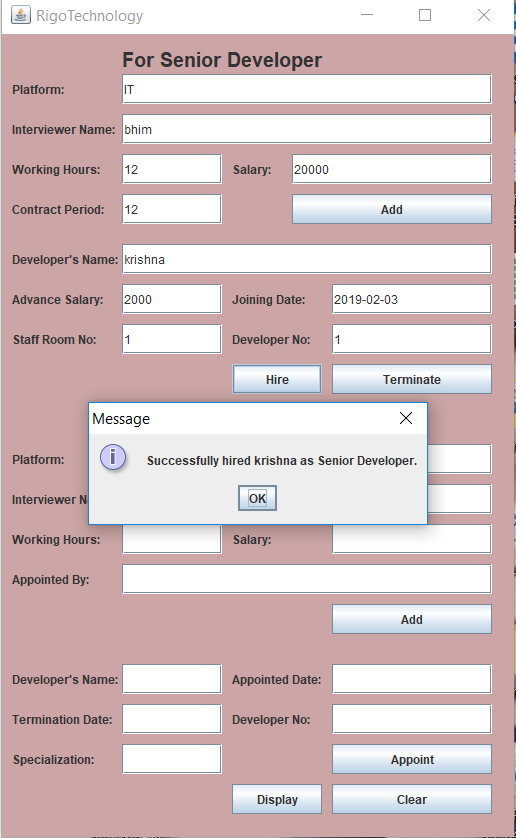


Figure 3 test 3

## Test 4

Objective: to terminate senior developer after being appointed

Action Type : information of senior developer being removed

Expected Result : information of senior developer should be removed

Actual result : the senior developer is terminated

Conclusion: successful

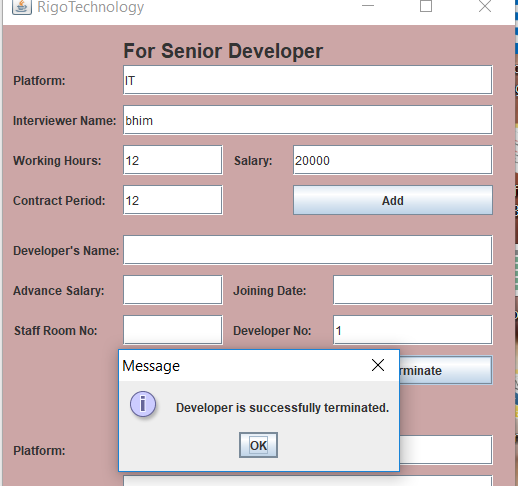


Figure 4 test 4.1

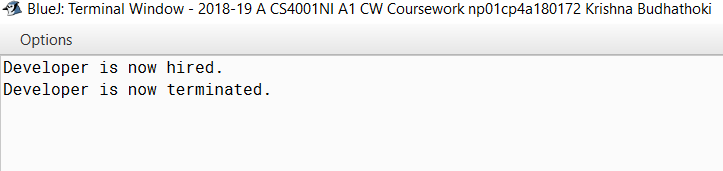


Figure 5 test 4.2

## Test 5

Objective: to add information for junior developer after being appointed

Action Type : platform, interviewer name, working hours, salary, appointed by is stored for appointing

Expected Result : information should be added for junior developer

Actual result : information is added for appointing junior developer

Conclusion: successful

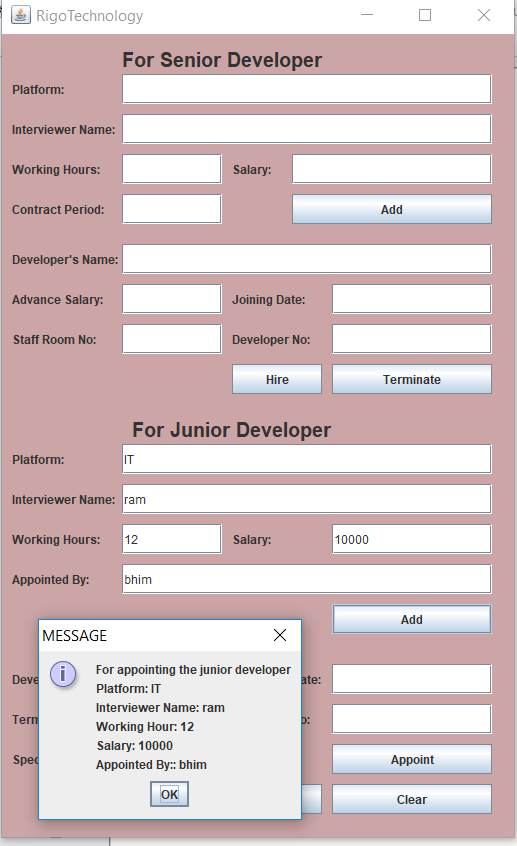


Figure 6 test 5

## Test 6

Objective: to appoint junior developer

Action Type : information of junior developer for appointing

Expected Result : junior developer should be appointed

Actual result : junior developer is appointed

Conclusion: successful

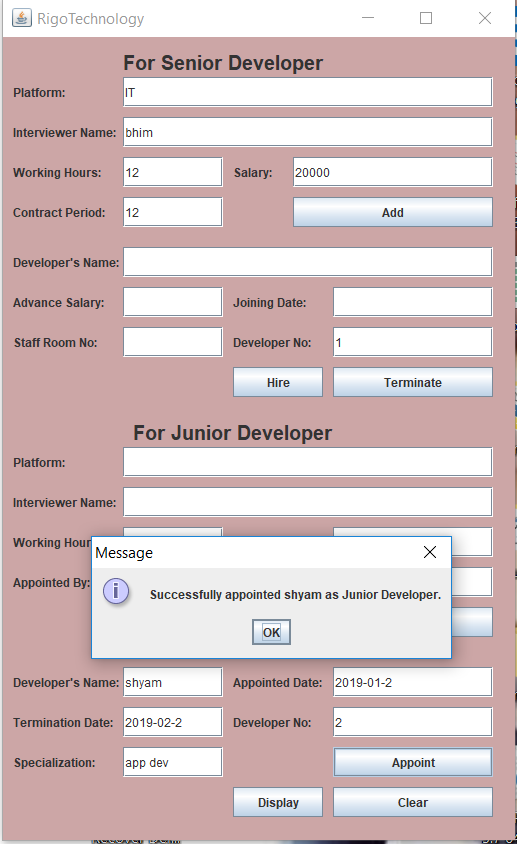


Figure 7 test 6.1

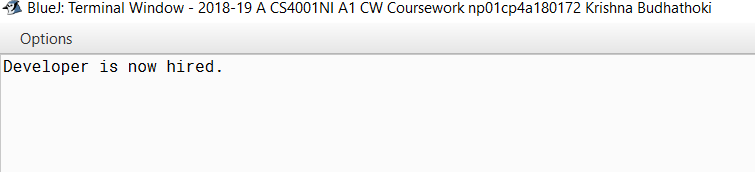


Figure 8 test 6.2

## Test 7

Objective: to display all the information after all the test

Action Type: displaying all the information

Expected Result : all the information of senior and junior developer should be displayed

Actual result : all the information of senior and junior developer after appointed, termination is displayed

Conclusion: successful

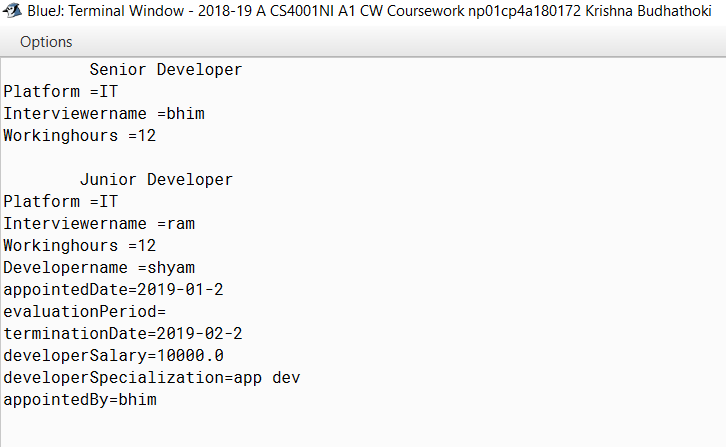


Figure 9 test 7

## Test 8

Objective: to show errors of each false value input in developer no

Action Type : error for developer no

Expected Result : pop message of error for developer no wrong input

Actual result : pop message was displayed in the screen for error

Conclusion: successful

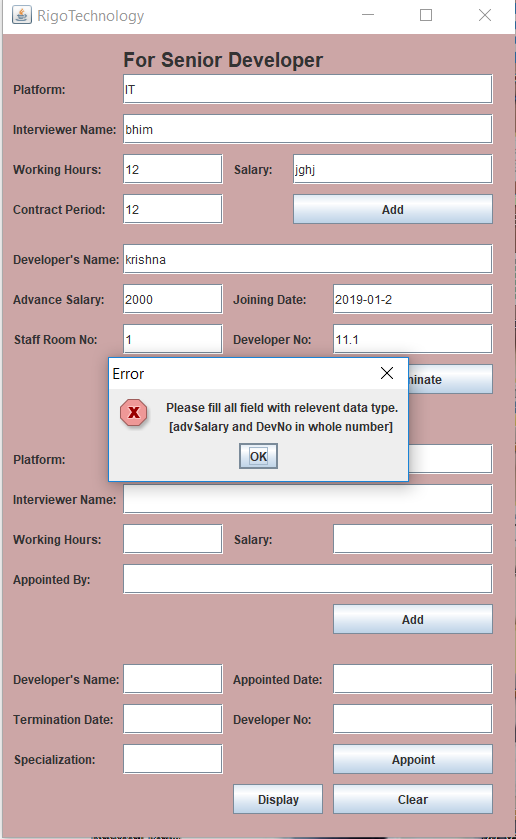


Figure 10 test 8.1

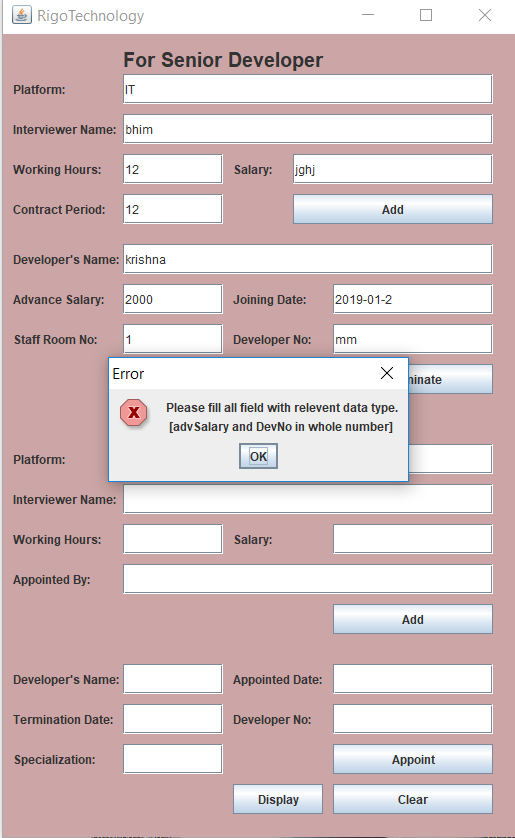


Figure 11 test 8.2

# Error

Figure 12 error 1

The given error was encountered by using wrong variable on the methods for action event which was solved.

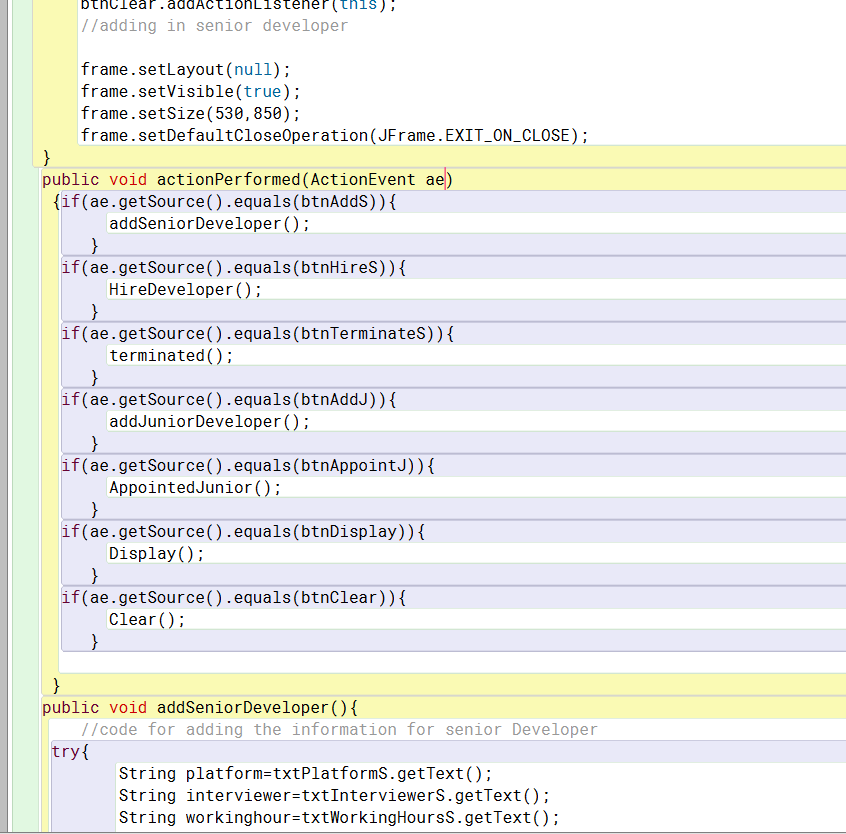


Figure 13 debugged 1

Error 2

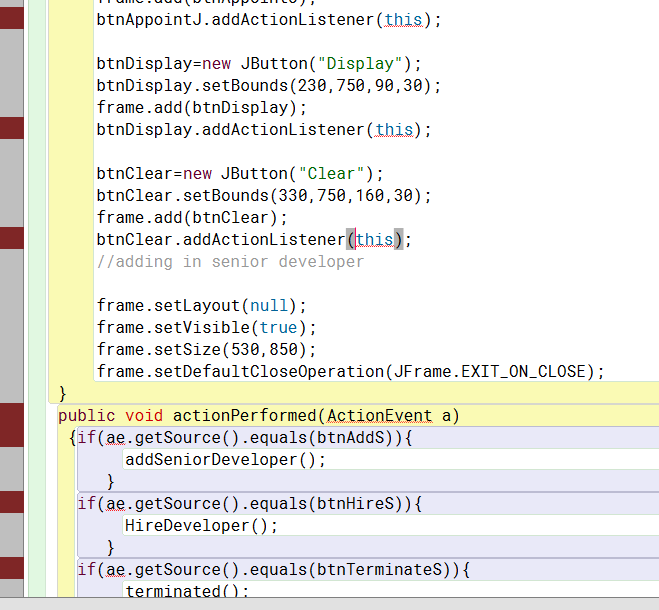


Figure 14 error 2

The given error was encountered by not using reliable java source which is import java.awt.event.\* which was debugged.

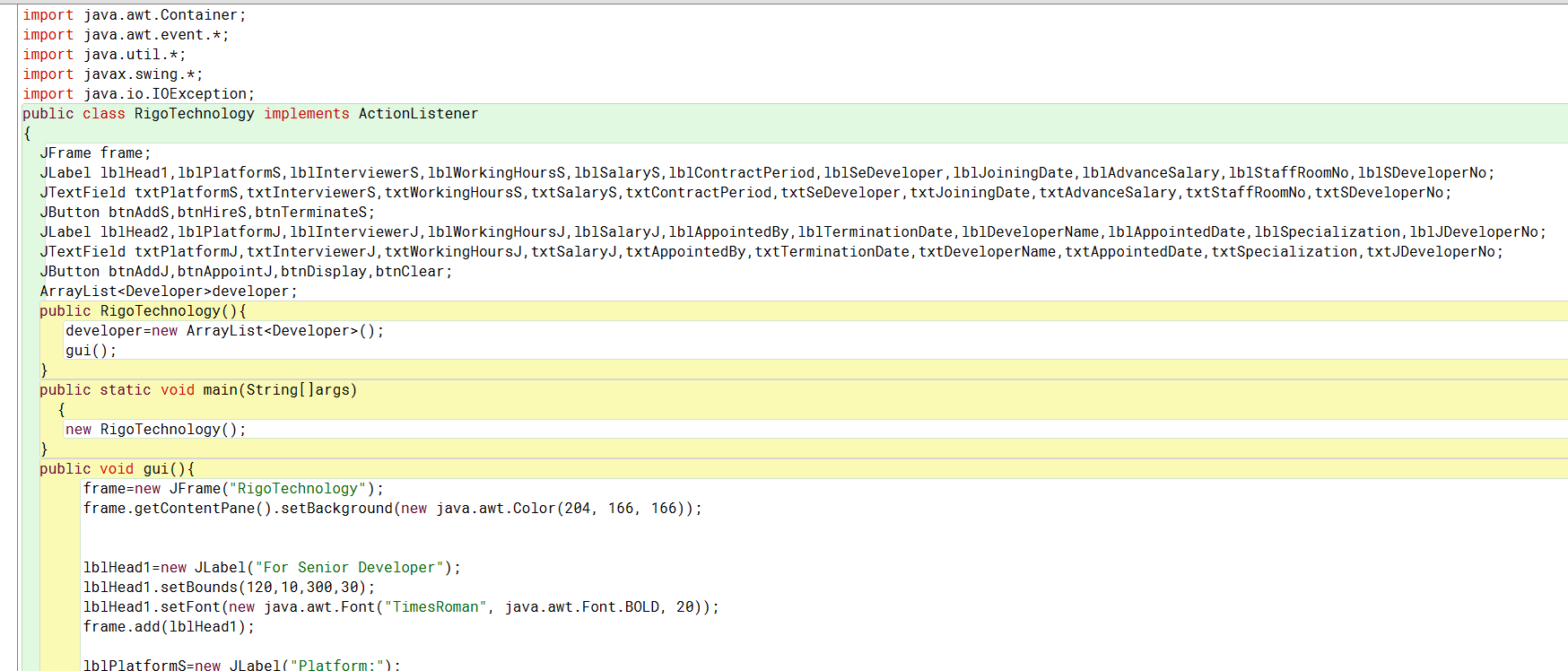


Figure 15 debugged 2

Error 3

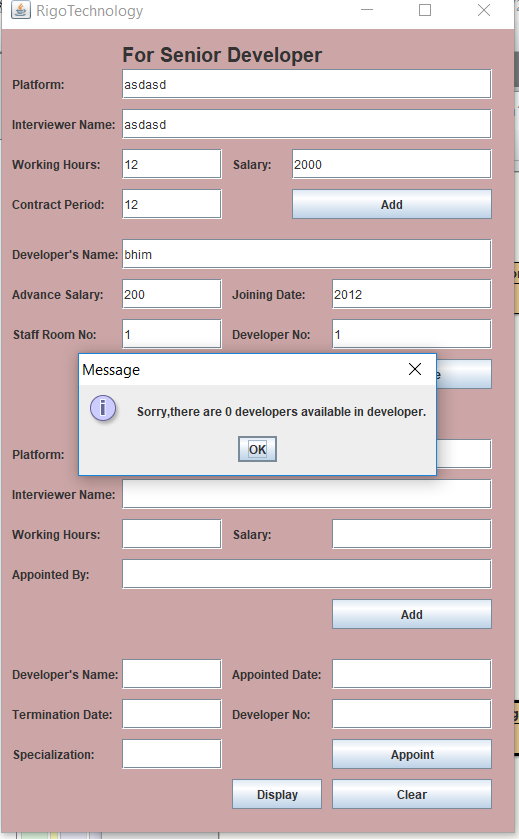


Figure 16 error 3

In the given above figure I was not able to appoint senior developer even though I used suitable data types which was debugged.

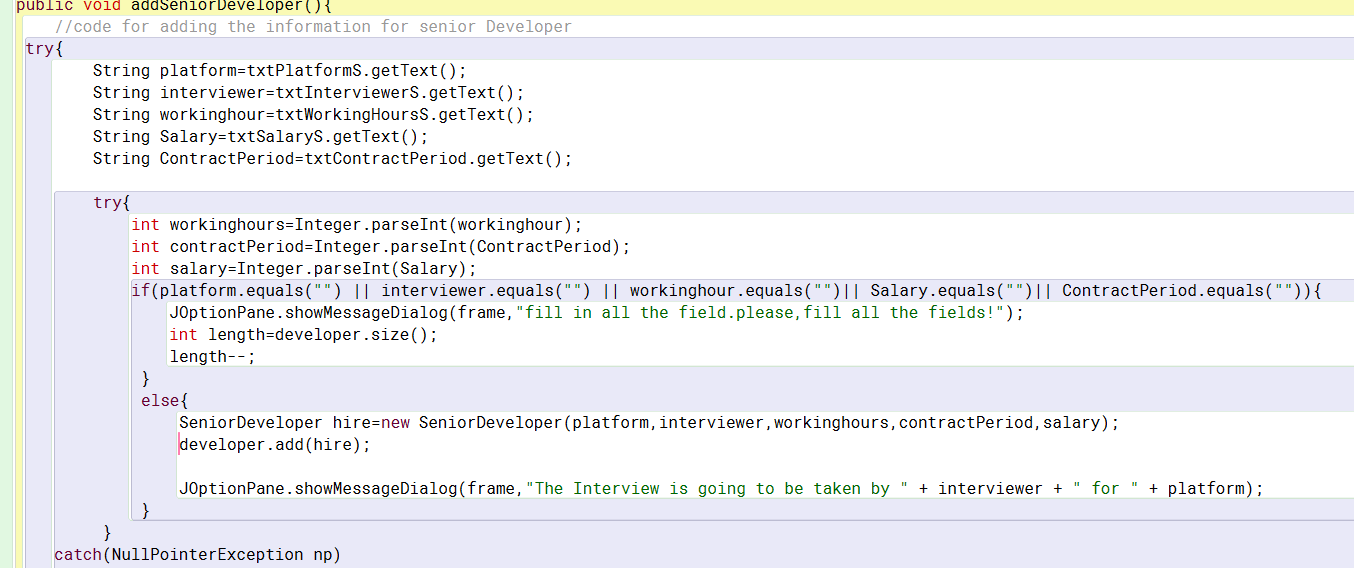


Figure 17 debugged 3

# Conclusion

This program was written based on BlueJ according to the scenario we were given and several tests were carried out.

While doing this coursework I faced many problems like no Internet, lack of time. The main problem that I faced was that I could not solve error on my own and it took a lot of time so, I consulted friends and sir for solution and eventually I was able to solve those problem. I also visited several sites for naming and code conventions to make my code flawless. And at the end I was able to develop a program with the help and cooperation from my friends, teachers and some useful sites.

# Bibliography

java. (2016) *java* [Online]. Available from: <http://docs.oracle.com/javaee/6/tutorial/doc/gjbbp.html> [Accessed 2 19 2019].

# Appendix

import java.awt.Container;

import java.awt.event.\*;

import java.util.\*;

import javax.swing.\*;

import java.io.IOException;

public class RigoTechnology implements ActionListener

{

JFrame frame;

JLabel lblHead1,lblPlatformS,lblInterviewerS,lblWorkingHoursS,lblSalaryS,lblContractPeriod,lblSeDeveloper,lblJoiningDate,lblAdvanceSalary,lblStaffRoomNo,lblSDeveloperNo;

JTextField txtPlatformS,txtInterviewerS,txtWorkingHoursS,txtSalaryS,txtContractPeriod,txtSeDeveloper,txtJoiningDate,txtAdvanceSalary,txtStaffRoomNo,txtSDeveloperNo;

JButton btnAddS,btnHireS,btnTerminateS;

JLabel lblHead2,lblPlatformJ,lblInterviewerJ,lblWorkingHoursJ,lblSalaryJ,lblAppointedBy,lblTerminationDate,lblDeveloperName,lblAppointedDate,lblSpecialization,lblJDeveloperNo;

JTextField txtPlatformJ,txtInterviewerJ,txtWorkingHoursJ,txtSalaryJ,txtAppointedBy,txtTerminationDate,txtDeveloperName,txtAppointedDate,txtSpecialization,txtJDeveloperNo;

JButton btnAddJ,btnAppointJ,btnDisplay,btnClear;

ArrayList<Developer>developer;

public RigoTechnology(){

developer=new ArrayList<Developer>();

gui();

}

public static void main(String[]args)

{

new RigoTechnology();

}

public void gui(){

frame=new JFrame("RigoTechnology");

frame.getContentPane().setBackground(new java.awt.Color(204, 166, 166));

lblHead1=new JLabel("For Senior Developer");

lblHead1.setBounds(120,10,300,30);

lblHead1.setFont(new java.awt.Font("TimesRoman", java.awt.Font.BOLD, 20));

frame.add(lblHead1);

lblPlatformS=new JLabel("Platform:");

lblPlatformS.setBounds(10,40,60,30);

frame.add(lblPlatformS);

txtPlatformS=new JTextField();

txtPlatformS.setBounds(120,40,370,30);

frame.add(txtPlatformS);

lblInterviewerS=new JLabel("Interviewer Name:");

lblInterviewerS.setBounds(10,80,115,30);

frame.add(lblInterviewerS);

txtInterviewerS=new JTextField();

txtInterviewerS.setBounds(120,80,370,30);

frame.add(txtInterviewerS);

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

lblWorkingHoursS.setBounds(10,120,100,30);

frame.add(lblWorkingHoursS);

txtWorkingHoursS=new JTextField();

txtWorkingHoursS.setBounds(120,120,100,30);

frame.add(txtWorkingHoursS);

lblSalaryS=new JLabel("Salary:");

lblSalaryS.setBounds(230,120,50,30);

frame.add(lblSalaryS);

txtSalaryS=new JTextField();

txtSalaryS.setBounds(290,120,200,30);

frame.add(txtSalaryS);

lblContractPeriod=new JLabel("Contract Period:");

lblContractPeriod.setBounds(10,160,110,30);

frame.add(lblContractPeriod);

txtContractPeriod=new JTextField();

txtContractPeriod.setBounds(120,160,100,30);

frame.add(txtContractPeriod);

btnAddS=new JButton("Add");

btnAddS.setBounds(290,160,200,30);

frame.add(btnAddS);

btnAddS.addActionListener(this);

lblSeDeveloper=new JLabel("Developer's Name:");

lblSeDeveloper.setBounds(10,210,110,30);

frame.add(lblSeDeveloper);

txtSeDeveloper=new JTextField();

txtSeDeveloper.setBounds(120,210,370,30);

frame.add(txtSeDeveloper);

lblAdvanceSalary=new JLabel("Advance Salary:");

lblAdvanceSalary.setBounds(10,250,110,30);

frame.add(lblAdvanceSalary);

txtAdvanceSalary=new JTextField();

txtAdvanceSalary.setBounds(120,250,100,30);

frame.add(txtAdvanceSalary);

lblJoiningDate=new JLabel("Joining Date:");

lblJoiningDate.setBounds(230,250,200,30);

frame.add(lblJoiningDate);

txtJoiningDate=new JTextField();

txtJoiningDate.setBounds(330,250,160,30);

frame.add(txtJoiningDate);

lblStaffRoomNo=new JLabel("Staff Room No:");

lblStaffRoomNo.setBounds(10,290,100,30);

frame.add(lblStaffRoomNo);

txtStaffRoomNo=new JTextField();

txtStaffRoomNo.setBounds(120,290,100,30);

frame.add(txtStaffRoomNo);

lblSDeveloperNo=new JLabel("Developer No:");

lblSDeveloperNo.setBounds(230,290,100,30);

frame.add(lblSDeveloperNo);

txtSDeveloperNo=new JTextField();

txtSDeveloperNo.setBounds(330,290,160,30);

frame.add(txtSDeveloperNo);

btnHireS=new JButton("Hire");

btnHireS.setBounds(230,330,90,30);

frame.add(btnHireS);

btnHireS.addActionListener(this);

btnTerminateS=new JButton("Terminate");

btnTerminateS.setBounds(330,330,160,30);

frame.add(btnTerminateS);

btnTerminateS.addActionListener(this);

lblHead2=new JLabel("For Junior Developer");

lblHead2.setBounds(130,380,300,30);

lblHead2.setFont(new java.awt.Font("TimesRoman", java.awt.Font.BOLD, 20));

frame.add(lblHead2);

lblPlatformJ=new JLabel("Platform:");

lblPlatformJ.setBounds(10,410,60,30);

frame.add(lblPlatformJ);

txtPlatformJ=new JTextField();

txtPlatformJ.setBounds(120,410,370,30);

frame.add(txtPlatformJ);

lblInterviewerJ=new JLabel("Interviewer Name:");

lblInterviewerJ.setBounds(10,450,115,30);

frame.add(lblInterviewerJ);

txtInterviewerJ=new JTextField();

txtInterviewerJ.setBounds(120,450,370,30);

frame.add(txtInterviewerJ);

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

lblWorkingHoursJ.setBounds(10,490,100,30);

frame.add(lblWorkingHoursJ);

txtWorkingHoursJ=new JTextField();

txtWorkingHoursJ.setBounds(120,490,100,30);

frame.add(txtWorkingHoursJ);

lblSalaryJ=new JLabel("Salary:");

lblSalaryJ.setBounds(230,490,100,30);

frame.add(lblSalaryJ);

txtSalaryJ=new JTextField();

txtSalaryJ.setBounds(330,490,160,30);

frame.add(txtSalaryJ);

lblAppointedBy=new JLabel("Appointed By:");

lblAppointedBy.setBounds(10,530,90,30);

frame.add(lblAppointedBy);

txtAppointedBy=new JTextField();

txtAppointedBy.setBounds(120,530,370,30);

frame.add(txtAppointedBy);

btnAddJ=new JButton("Add");

btnAddJ.setBounds(330,570,160,30);

frame.add(btnAddJ);

btnAddJ.addActionListener(this);

lblDeveloperName=new JLabel("Developer's Name:");

lblDeveloperName.setBounds(10,630,115,30);

frame.add(lblDeveloperName);

txtDeveloperName=new JTextField();

txtDeveloperName.setBounds(120,630,100,30);

frame.add(txtDeveloperName);

lblAppointedDate=new JLabel("Appointed Date:");

lblAppointedDate.setBounds(230,630,100,30);

frame.add(lblAppointedDate);

txtAppointedDate=new JTextField();

txtAppointedDate.setBounds(330,630,160,30);

frame.add(txtAppointedDate);

lblTerminationDate=new JLabel("Termination Date:");

lblTerminationDate.setBounds(10,670,100,30);

frame.add(lblTerminationDate);

txtTerminationDate=new JTextField();

txtTerminationDate.setBounds(120,670,100,30);

frame.add(txtTerminationDate);

lblJDeveloperNo=new JLabel("Developer No:");

lblJDeveloperNo.setBounds(230,670,100,30);

frame.add(lblJDeveloperNo);

txtJDeveloperNo=new JTextField();

txtJDeveloperNo.setBounds(330,670,160,30);

frame.add(txtJDeveloperNo);

lblSpecialization=new JLabel("Specialization:");

lblSpecialization.setBounds(10,710,100,30);

frame.add(lblSpecialization);

txtSpecialization=new JTextField();

txtSpecialization.setBounds(120,710,100,30);

frame.add(txtSpecialization);

btnAppointJ=new JButton("Appoint");

btnAppointJ.setBounds(330,710,160,30);

frame.add(btnAppointJ);

btnAppointJ.addActionListener(this);

btnDisplay=new JButton("Display");

btnDisplay.setBounds(230,750,90,30);

frame.add(btnDisplay);

btnDisplay.addActionListener(this);

btnClear=new JButton("Clear");

btnClear.setBounds(330,750,160,30);

frame.add(btnClear);

btnClear.addActionListener(this);

//adding in senior developer

frame.setLayout(null);

frame.setVisible(true);

frame.setSize(530,850);

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

}

public void actionPerformed(ActionEvent ae)

{if(ae.getSource().equals(btnAddS)){

addSeniorDeveloper();

}

if(ae.getSource().equals(btnHireS)){

HireDeveloper();

}

if(ae.getSource().equals(btnTerminateS)){

terminated();

}

if(ae.getSource().equals(btnAddJ)){

addJuniorDeveloper();

}

if(ae.getSource().equals(btnAppointJ)){

AppointedJunior();

}

if(ae.getSource().equals(btnDisplay)){

Display();

}

if(ae.getSource().equals(btnClear)){

Clear();

}

}

public void addSeniorDeveloper(){

//code for adding the information for senior Developer

try{

String platform=txtPlatformS.getText();

String interviewer=txtInterviewerS.getText();

String workinghour=txtWorkingHoursS.getText();

String Salary=txtSalaryS.getText();

String ContractPeriod=txtContractPeriod.getText();

try{

int workinghours=Integer.parseInt(workinghour);

int contractPeriod=Integer.parseInt(ContractPeriod);

int salary=Integer.parseInt(Salary);

if(platform.equals("") || interviewer.equals("") || workinghour.equals("")|| Salary.equals("")|| ContractPeriod.equals("")){

JOptionPane.showMessageDialog(frame,"fill in all the field.please,fill all the fields!");

int length=developer.size();

length--;

}

else{

SeniorDeveloper hire=new SeniorDeveloper(platform,interviewer,workinghours,contractPeriod,salary);

developer.add(hire);

JOptionPane.showMessageDialog(frame,"The Interview is going to be taken by " + interviewer + " for " + platform);

}

}

catch(NullPointerException np)

{

JOptionPane.showMessageDialog(frame,np,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

catch(NumberFormatException nf)

{

JOptionPane.showMessageDialog(frame,nf,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

}

catch(NumberFormatException nfe)

{

JOptionPane.showMessageDialog(frame,"Error in parsing string at addJuniorDeveloper()!!"+"\n"+nfe,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

}

//hireing the Senior Developer

public void HireDeveloper()

{

try

{

String name = txtSeDeveloper.getText();

String joiningDate = txtJoiningDate.getText();

String staffRoomNo = txtStaffRoomNo.getText();

int advSalary = Integer.parseInt(txtAdvanceSalary.getText());

int devNo = Integer.parseInt(txtSDeveloperNo.getText());

if(developer.size() < devNo)

{

JOptionPane.showMessageDialog(frame,"Sorry,there are "+developer.size()+" developers available in developer.");

}

else

{

if(name.equals("")||joiningDate.equals("")||staffRoomNo.equals(""))

{

JOptionPane.showMessageDialog(frame,"Please fill all fields.","Error",JOptionPane.ERROR\_MESSAGE);

}

else

{

if(developer.get(devNo-1) instanceof SeniorDeveloper)

{

//converting type to senior dev

SeniorDeveloper objS = (SeniorDeveloper)developer.get(devNo-1);

//checking if developer is already appointed

if(objS.isAppointed() == false)

{

// calling hire senior dev method

objS.hire(name, joiningDate, advSalary,staffRoomNo);

JOptionPane.showMessageDialog(frame,"Successfully hired "+name+" as Senior Developer.");

//clearing fields

txtSeDeveloper.setText("");

txtJoiningDate.setText("");

txtStaffRoomNo.setText("");

txtAdvanceSalary.setText("");

txtJDeveloperNo.setText("");

}

else

{

JOptionPane.showMessageDialog(frame,"A developer has already been hired.");

}

}

else

{

JOptionPane.showMessageDialog(frame,"Sorry,Developer isn't Senior Developer.");

}

}

}

}

catch(Exception e)

{

JOptionPane.showMessageDialog(frame,"Please fill all field with relevent data type.\n [advSalary and DevNo in whole number]","Error",JOptionPane.ERROR\_MESSAGE);

}

}

public void terminated()

{

try{

int developerNo = Integer.parseInt(txtSDeveloperNo.getText());

if(developer.size()<developerNo){

JOptionPane.showMessageDialog(frame,developer.size()+" developers available in developer.","Error",JOptionPane.ERROR\_MESSAGE);

}else{

if(developer.get(developerNo-1) instanceof SeniorDeveloper){

SeniorDeveloper objS = (SeniorDeveloper)developer.get(developerNo-1);

if(objS.isAppointed() == true){

if(objS.isTerminated() == false){

objS.contractTerminated();

JOptionPane.showMessageDialog(frame,"Developer is successfully terminated.");

txtSDeveloperNo.setText("");

}else{

JOptionPane.showMessageDialog(frame,"Developer has already been terminated.","Error",JOptionPane.ERROR\_MESSAGE);

}

}else{

JOptionPane.showMessageDialog(frame,"Developer hasn't been appointed.","Error",JOptionPane.ERROR\_MESSAGE);

}

}else{

JOptionPane.showMessageDialog(frame,"Developer isn't Senior developer.","Error",JOptionPane.ERROR\_MESSAGE);

}

}

}

catch(Exception Event){

JOptionPane.showMessageDialog(frame,"Please fill Developer No.(in whole number)","Error",JOptionPane.ERROR\_MESSAGE);

}

}

public void addJuniorDeveloper()

{

try

{

//code for adding the information for junior Developer

String PlatformJ= txtPlatformJ.getText();

String InterviewerJ =txtInterviewerJ.getText();

String WorkingHoursJ =txtWorkingHoursJ.getText();

String SalaryJ =txtSalaryJ.getText();

String AppointedBy =txtAppointedBy.getText();

String terminationDate=txtTerminationDate.getText();

double SalaryJe=0.0;

try

{

int WorkingHours=Integer.parseInt(txtWorkingHoursJ.getText());

SalaryJe=Double.parseDouble(txtSalaryJ.getText());

if (PlatformJ.equals("") || InterviewerJ.equals("") || WorkingHoursJ.equals(" ") || SalaryJ.equals("") ||AppointedBy.equals("") )

{

JOptionPane.showMessageDialog(frame,"You should fill in all the fields. Please, fill all the fields!");

int length =developer.size();

}

else

{

if ( SalaryJe<=0)

{

JOptionPane.showMessageDialog(frame,"Please! Enter the valid number","MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

else{

JuniorDeveloper hireJ= new JuniorDeveloper(PlatformJ,InterviewerJ,WorkingHours,SalaryJe,AppointedBy,terminationDate);

developer.add(hireJ);

JOptionPane.showMessageDialog(frame,"For appointing the junior developer"+"\n"+"Platform:"+" "+PlatformJ+"\n"+"Interviewer Name:"+" "+InterviewerJ+"\n"+"Working Hour:"+" "+WorkingHoursJ +"\n"+"Salary:"+" "+SalaryJ +"\n"+"Appointed By::"+" "+AppointedBy,"MESSAGE",JOptionPane.INFORMATION\_MESSAGE);

}

}

}

catch(NullPointerException np)

{

JOptionPane.showMessageDialog(frame,np,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

catch(NumberFormatException nf)

{

JOptionPane.showMessageDialog(frame,nf,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

}

catch(NumberFormatException nfe)

{

JOptionPane.showMessageDialog(frame,"Error in parsing string at addGenToRent()!!"+"\n"+nfe,"MESSAGE",JOptionPane.ERROR\_MESSAGE);

}

txtPlatformJ.setText("");

txtInterviewerJ.setText("");

txtWorkingHoursJ.setText("");

txtSalaryJ.setText("");

txtAppointedBy.setText("");

}

public void AppointedJunior()

{

try

{

String name = txtDeveloperName.getText();

String appointedDate = txtAppointedDate.getText();

String terminationDate = txtTerminationDate.getText();

String specialization = txtSpecialization.getText();

int devNo = Integer.parseInt(txtJDeveloperNo.getText());

if(developer.size() < devNo)

{

JOptionPane.showMessageDialog(frame,"Sorry,there are "+developer.size()+" developers available in developer.");

}

else

{

if(name.equals("")||appointedDate.equals("")||terminationDate.equals("")||specialization.equals(""))

{

JOptionPane.showMessageDialog(frame,"Please fill all fields.","Error",JOptionPane.ERROR\_MESSAGE);

}

else

{

if(developer.get(devNo-1) instanceof JuniorDeveloper)

{

//changing type to junior developer

JuniorDeveloper objJ = (JuniorDeveloper)developer.get(devNo-1);

// checking if developer is already got joined

if(objJ.isJoined() == false)

{

//calling appoint junior developer

objJ.appointDeveloper(name,appointedDate, terminationDate,specialization);

JOptionPane.showMessageDialog(frame,name+"is successfully appointed as Junior Developer.");

//clearing fields

txtDeveloperName.setText("");

txtAppointedDate.setText("");

txtTerminationDate.setText("");

txtSpecialization.setText("");

txtJDeveloperNo.setText("");

}

else

{

JOptionPane.showMessageDialog(frame,"A developer has already Joined.");

}

}

else

{

JOptionPane.showMessageDialog(frame,"Sorry,Developer isn't Junior Developer.");

}

}

}

}

catch(Exception e)

{

JOptionPane.showMessageDialog(frame,"Please fill all field with relevent data type.\n [DevNo in whole number]","Error",JOptionPane.ERROR\_MESSAGE);

}

}

public void Display() {

//method for displaying all the information for senior and junior developer

for (Developer d: developer) {

if(d instanceof SeniorDeveloper) {

System.out.println(" Senior Developer ");

SeniorDeveloper dis = (SeniorDeveloper) d;

dis.display();

System.out.println(" ");

}

if(d instanceof JuniorDeveloper) {

System.out.println(" Junior Developer ");

JuniorDeveloper dis1 = (JuniorDeveloper) d;

dis1.display();

System.out.println(" ");

}

}

}

public void Clear(){

//code for clearing all the text fields

txtPlatformJ.setText("");

txtInterviewerJ.setText("");

txtWorkingHoursJ.setText("");

txtSalaryJ.setText("");

txtAppointedBy.setText("");

txtTerminationDate.setText("");

txtDeveloperName.setText("");

txtAppointedDate.setText("");

txtSpecialization.setText("");

txtJDeveloperNo.setText("");

txtPlatformS.setText("");

txtInterviewerS.setText("");

txtWorkingHoursS.setText("");

txtSalaryS.setText("");

txtContractPeriod.setText("");

txtSeDeveloper.setText("");

txtJoiningDate.setText("");

txtAdvanceSalary.setText("");

txtStaffRoomNo.setText("");

txtSDeveloperNo.setText("");

}

}

Appendix 2

public class Developer{

String platform;

String interviewername;

int workinghours;

String developername;

//initializing the attributes

public Developer(String platform,String interviewername,int workinghours){

this.platform=platform;

this.interviewername=interviewername;

this.workinghours=workinghours;

this.developername="";

}

//using each corresponding accessor method for accessing each method

public String getDeveloperName(){

return developername;

}

public String getplatform(){

return platform;

}

public String getInterviewername(){

return interviewername;

}

public int getworkinghours(){

return workinghours;

}

public void setdevelopername(String developername){

this.developername=developername;

}

//display method

public void display(){

System.out.println("Platform =" +getplatform());

System.out.println("Interviewername =" +getInterviewername());

System.out.println("Workinghours =" +getworkinghours());

if(!developername.equals("")){

System.out.println("Developername =" +developername);

}

}

}

public class SeniorDeveloper extends Developer{

int salary;

String joiningDate;

String staffRoomNumber;

int contractPeriod;

float advanceSalary;

boolean appointed;

boolean terminated;

//initializing the attributes

public SeniorDeveloper(String platform,String interviewername,int workinghours,int salary,int contractPeriod){

super(platform,interviewername,workinghours);

this.salary=salary;

this.workinghours=workinghours;

this.contractPeriod = contractPeriod;

joiningDate="";

staffRoomNumber="";

advanceSalary=0f;

appointed=false;

terminated=false;

}

//using each corresponding accessor method for accessing each method

public float getSalary(){

return salary;

}

public String getJoiningDate(){

return joiningDate;

}

public String getStaffRoomNumber(){

return staffRoomNumber;

}

public int getContractPeriod(){

return contractPeriod;

}

public float getAdvanceSalary(){

return advanceSalary;

}

public boolean isAppointed(){

return appointed;

}

public boolean isTerminated(){

return terminated;

}

public void setSalary(int salary){

this.salary=salary;

}

public void setContractPeriod(int contractPeriod){

this.contractPeriod=contractPeriod;

}

// method to hire developer

public void hire(String developername,String joiningDate,float advanceSalary,String staffRoomNumber){

if (appointed==true){

System.out.println("Developer is already appointed.");

System.out.println("developer name= "+developername);

System.out.println("room no= "+getStaffRoomNumber());

}

else if (appointed==false){

setdevelopername(developername);

this.joiningDate=joiningDate;

this.staffRoomNumber=staffRoomNumber;

this.advanceSalary=advanceSalary;

appointed=true;

terminated=false;

System.out.println("Developer is now hired.");

}

}

public void contractTerminated(){

if (terminated==true){

System.out.println("you are already terminated");

}

else if (terminated==false){

this.terminated=true;

setdevelopername("");

joiningDate="";

advanceSalary=0.0f;

appointed=false;

terminated=true;

System.out.println("Developer is now terminated.");

}

}

public void print(){

System.out.println("platform= "+getplatform());

System.out.println("interviewersName= "+getInterviewername());

System.out.println("salary= "+getSalary());

}

//method to display

public void display(){

super.display();

if (appointed==true){

System.out.println("termination= "+isTerminated());

System.out.println("joindate= "+getJoiningDate());

System.out.println("advancesalary= "+getAdvanceSalary());

}

}

}

public class JuniorDeveloper extends Developer{

double salary;

String appointedDate;

String evaluationDate;

String terminationDate;

String specialization;

String appointedBy;

boolean joined;

//initializing the attributes

public JuniorDeveloper(String platform,String interviewername,int workinghours,double salary,String appointedBy,String terminationDate){

super(platform,interviewername,workinghours);

this.appointedBy=appointedBy;

this.terminationDate=terminationDate;

this.salary = salary;

appointedDate="";

evaluationDate="";

specialization="";

joined=false;

}

//using each corresponding accessor method for accessing each method

public double getSalary(){

return salary;

}

public String getAppointedDate(){

return appointedDate;

}

public String getEvaluationDate(){

return evaluationDate;

}

public String getTerminationDate(){

return terminationDate;

}

public String getSpecialization(){

return specialization;

}

public String getAppointedBy(){

return appointedBy;

}

public boolean isJoined(){

return joined;

}

public void setSalary(double salary){

if (joined==false){

this.salary=salary;

System.out.println("you salary is changedd");

}

else if (joined==true){

System.out.println("it is not possible to change the salary");

}

}

public void appointDeveloper(String developername,String appointedDate,String terminationDate,String specialization){

if (joined==false){

setdevelopername(developername);

this.appointedDate = appointedDate;

this.terminationDate = terminationDate;

this.specialization = specialization;

joined = true;

System.out.println("Developer is appointed.");

}

else if(joined==true){

System.out.println("you are already joined");

}

}

//method to display

public void display(){

super.display();

if (joined==true){

System.out.println("appointedDate="+appointedDate);

System.out.println("evaluationPeriod="+evaluationDate);

System.out.println("terminationDate="+terminationDate);

System.out.println("developerSalary="+salary);

System.out.println("developerSpecialization="+specialization);

System.out.println("appointedBy="+appointedBy);

}

}

}