Module Code & Module Title

CS4001NA Programming

Assessment Weightage & Type

35% Individual Coursework

Year and Semester

2018-19 Autumn / 2018-19 Spring

Student Name: Shasank Shakya

London Met ID: SHS1345

College ID: NP01CP4A18303

Assignment Due Date: 12th July 2019

Assignment Submission Date: 12th July 2019

Word Count (Where Required):

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

Table of figure

Introduction

This semester we have learned java programming from the scratch. First we learned to write code in java through notepad. We had to install JDK and compile the written code in notepad manually. Finally the program would be executed through command prompt. Later we were introduced to BlueJ which is an IDE build for java only. By the time we this course work was given we had understood the fundamentals of java. We were able to write small calculation programs, some of simple loop programs, etc. Then we were taught object oriented programming for java and this was the most important concept for coursework one. We understood how the classes and objects work in java and that each class need an object to be accessed. We also learned about difference from calling object from a constructor. Using a constructor was a helpful to make this project as well.

What we have to do in this coursework is to make a main class named developer. This developer class has two sub classes junior developer and senior developer. Both of these classes over ride developer class. Developer class has four attributes whereas junior and senior developer class has 7 attributes each. There are specific accessor methods for assigning certain attributes. Some method help to display all the values after they are assigned.

Class Diagram

Developer	
-platform : String	
-interviewerName : String	
-developerName : String	
-workingHours : Integer	
+Developer()	(Constructor)
+getPlatform()	
+getInterviewerName()	
+getDeveloperName()	
+getWorkingHours()	
+setDeveloperName()	
+display()	

SeniorDeveloper	
-salary : Integer	
-joiningDate : String	
-staffRoomNumber : String	
-contractPeriod :Integer	
-advancedSalary : Integer	
-appointed : Boolean	
-terminated : Boolean	
+SeniorDeveloper()	(Constructor)
+getSalary()	
+getJoiningDate()	
+getStaffRoomNumber()	
+getContractPeriod()	
+getAdvancedSalary()	
+getAppointed()	
+getTerminated()	
+setSalary()	
+setContactPeriod()	

+AppointDeveloper()
+contractTermination()
+print()
+display()

JuniorDeveloper
-salary : Integer -appointedDate : String -evaluationDate : String -terminationDate : String -specialization : String -appointedBy : String -joined : Boolean

+JuniorDeveloper() (Constructor)
+getSalary()
+getAppointedDate()
+getEvaluationPeriod()
+getTerminationDate()
+getSpecialization()
+getAppointedBy()
+getJoined()
+setSalary()
+AppointedDeveloper()
+displays()

Pseudocode

```
Class Developer
platform =  string

interviewerName = string

developerName = string

workingHours = integer


Developer(platform =  string, interviewerName =  string, workingHours =  integer, )

    This.platform = platform

    This.interviewerName = interviewerName

    This.developerName = “”

    This.workingHours = workingHours


Function getPlatform()

Return this.platform


Function getInterviewerName()

Return this.interviewerName


Function getDeveloperName()

Return this.developerName


Function getWorkingHours()

Return this.workingHours
```

```
Function setDeveloperName(developerName)
  This.developerName = developerName


Function display()
  Print “the name of the platform is ” + getPlatform()
  Print “the number of working hours is” + getWorkingHours()
  Print “the name of the interviewer” + getInterviewerName()
  if  This.developerName != “” then
        print “the name of the developer is ” + getDeveloperName()
```

```
Class SeniorDeveloper extends Developer
salary =  double
joiningDate = string
staffRoomNumber = string
contractPeriod = integer
advancedSalary = interger
appointed = boolean
terminated = Boolean

SeniorDevelper(salary = double, contractPeriod = integer, platform = string, interviewerName = string, workingHours = integer)
Superclass Developer(platform, interviewerName, workingHours)
This.salary = salary
This.joiningDate = “”
This.staffRoomNumber = “”
This.contractPeriod = contractPeriod
This.advancedSalary = 0
This.appointed = false
This.terminated = false

Function getSalary()
Return this.salary

Function getJoiningDate()
Return this.joiningDate
```

```
Function getStaffRoomNumber()
Return this.staffRoomNumber


Function getContractPeriod()
Return this.contractPeriod


Function getAdvancedSalary()
Return this.advancedSalary


Function getTerminated()
Return this.terminated


Function setSalary(salary)
Return this.salary


Function setContractPeriod(contactPeriod)
Return this.contractPeriod


Function AppointDeveloper(newDeveloperName, joiningDate, advancedSalary, staffRoomNumber)
  If this.appointed
     Print getDeveloperName() + “is the name of the developer and their staff room number is ” + getStaffRoomNumber()
  Else
     Super.setDeveloperName(newDeveloperName)
  This.joiningDate = joiningDate

  This.staffRoomNumber = staffRoomNumber

  This.appointed = true

  This.terminated = false


Function contractTermination()

  If this.terminated

      Print “contract already terminated”

  Else

      Super.setDeveloperName(“”)

      This.joiningDate = “”

      This.advancedSalary = 0

      This.appointed = false

      This.terminated = true


Function printOut()

   print “the name of the platform is” + getPlatform()

   print “the name of the interviewer is ” getInterviewerName()

   print “this salary is ”+ getSalary()
```

```
Class JuniorDeveloper extends Developer
  Salary = double
  appointedDate = string
  evaluationPeriod = string
  terminationDate = string
  specialization = string
  appointedBy = string
  joined = Boolean

JuniorDeveloper (platform = string, interviewerName = string, workingHours = integer,
salary = double, appointedBy = string)
  Super class Developer(platform,interviewerName,wokrkingHours)
  This.salary= salary
  This.appointedDate = “”
  This.evaluationPeriod = “”
  This.terminationDate = “”
  This.specialization = “”
  This.appointedBy = appointedBy
  This.joined = false

Function getSalary()
Return this.salary

Function getAppointedDate()
Return this.appointedDate
```

```
Function getEvaluationPeriod()
Return this.evaluationPeriod


Function setSalary(salary)
   If this.joined = false
       This.salary = salary
  Else
      Print “Not possible to change the salary”


Function display()
   Super.display()
   If this.joined
        Print “the developer was appoineted on ” +getAppointedDate()
        Print “the name w of the developer was ” + getEvaluationPeriod()
        Print “the termination date of the developer was ” + getTerminationDate()
        Print “the developer’s salary Is ” + getSalary()
        Print “the specialization of the developer is” + getSpecialization()
        Print “the developer was appointed by ” + getAppointedBy()
```

Method Description

Developer class

Developer() : this method is a constructor hence it instantiates the objects it has in it. Which are platform, interViewerName, developerName, workingHours.

getPlatform() : this method accesses field named platform

getInterviewerName() : this method accesses field named interviewerName

getDeveloperName() : this method accesses field named developerName

getWorkingHours() : this method accesses field named workingHours

display() : this method prints platform, working hours, interviewer name and developer name if assigned.

setDeveloperName(): this method is to assign a value to field named DeveloperName

SeniorDeveloper Class

SeniorDeveloper() : this method is a constructor used to instantiate objects in it. Platform interviewerName and workingHours are called from super class which is developer. Others like salary, joiningDate, staffRoomNumber, contractPeriod, advancedSalary, appointed and terminated are instantiated.

getSalary() : this method accesses field named salary

getJoiningDate(): this method accesses field named joiningDate

getStaffRoomNumber(): this method accesses field named staffRoomNumber

getContactPeriod(): this method accesses field named contractPeriod

getAdvancedSalary(): this method accesses field named advancedSalary

getAppointed(): this method accesses field named appointed

setSalary() : this method assigns a value to field salary

setContractPeriod() : this method assigns a value to field contractPeriod

AppointedDeveloper() : this method prints developerName and staffRoomNumber if appointed is true else it assigns values developerName, joiningDate, staffRoomNumber, appointed to true and terminated to false.

ContractTermination() : this method prints contract already terminated if value of terminated is true else it terminates whatever things have been assigned by assigning developerName and joiningDate as empty string, advancedSalary to 0, apoointed to false and terminated to true.

printOut() : this method prints out platform, interviewerName and salary.

Display() : this method prints contract is terminated if terminated is true else prints contract is not terminated. Also prints out joiningDate and advancedSalary.

JuniorDeveloper Class

JuniorDeveloper() : this is the constructor of this class to instantiate objects. It instantiates platform, interviewerName, workingHours from super class. Salary, appointedDate, evaluationPeriod, terminationDate, specialization, appointedBy, joined are instantiated with this constructor normally.

getSalary() : this method accesses field named salary.

getAppointedDate() : this method accesses field named appointedDate.

getEvaluationPeriod() : this method accesses field named evaluationPeriod.

getTerminationDate(): this method accesses field named terminationDate.

getSpecialization(): this method accesses field named specialization

getAppointedBy(): this method accesses field named appointedBy

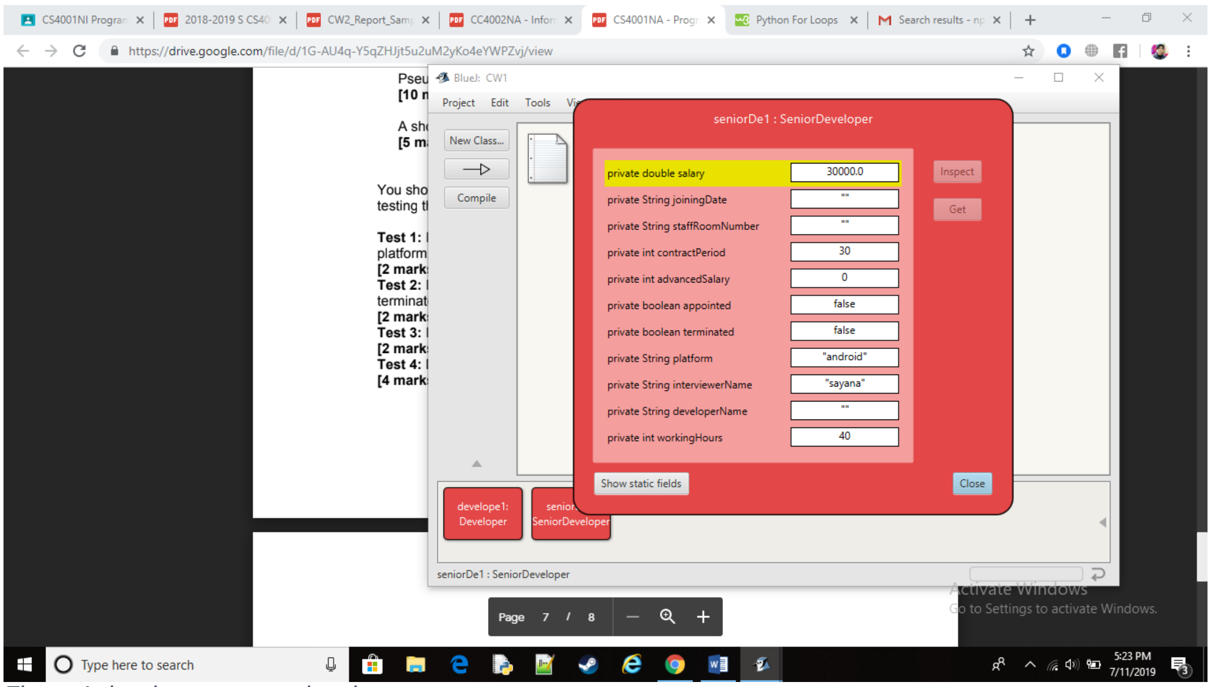
getJoined() : this method accesses field named joined

AppointedDeveloper() : this medthod assigns developerName, appointedDate, termimnationDate, specialization, evaluationPeriod and sets joined value as true. Else Prints Developer already appointed.

Displays() : this method prints out AppointedDate, evaluationPeriod, TerminationDate, salary, specialization, appointedBy if value of joined is true.

Testing

Test 1:

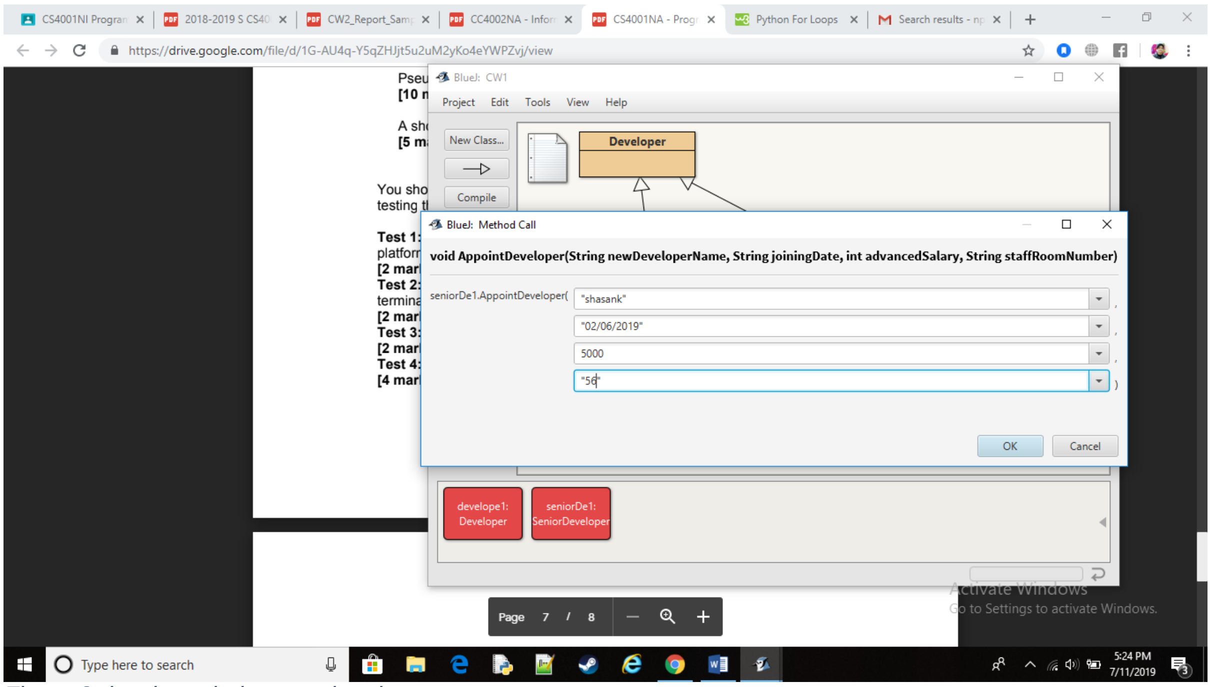


seniorDe1 : SeniorDeveloper

private double salary	30000.0
private String joiningDate	""
private String staffRoomNumber	""
private int contractPeriod	30
private int advancedSalary	0
private boolean appointed	false
private boolean terminated	false
private String platform	"android"
private String interviewerName	"sayana"
private String developerName	""
private int workingHours	40

Figure 1 developer not appointed

SeniorClass developer without appointing a developer.



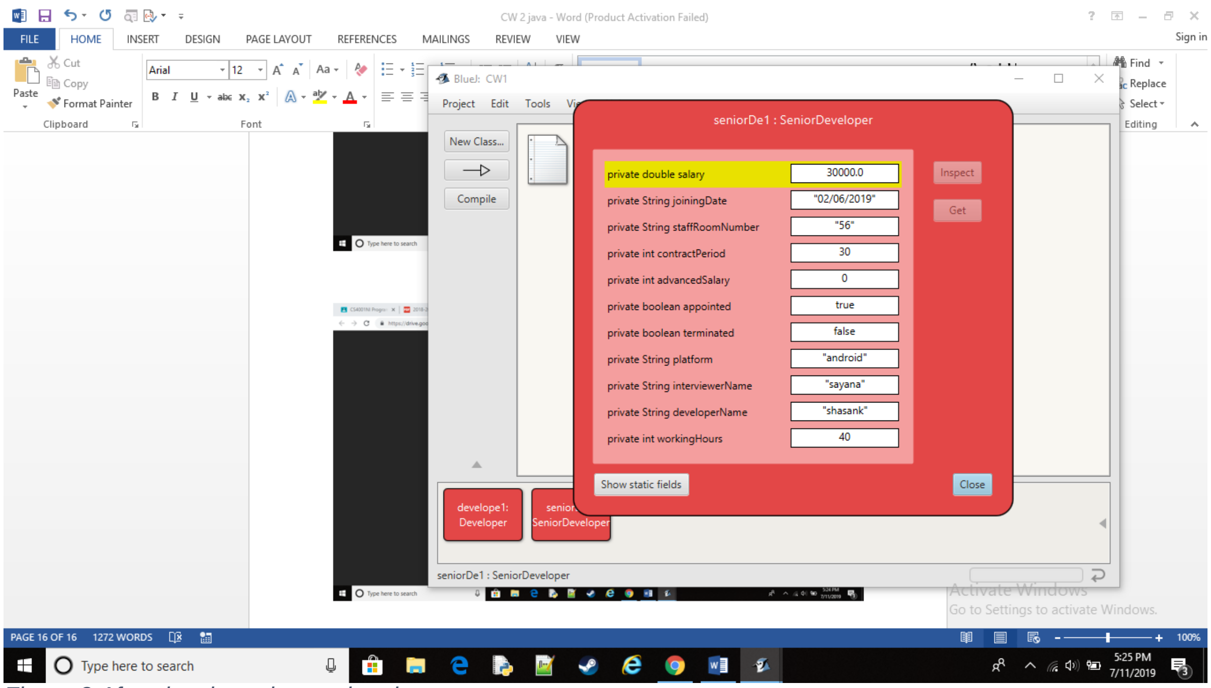
BlueJ: Method Call

void AppointDeveloper(String newDeveloperName, String joiningDate, int advancedSalary, String staffRoomNumber)

```
seniorDe1.AppointDeveloper( "shasank"
                            "02/06/2019"
                            5000
                            "56" )
```

Figure 2 developer being appointed

Appointing a developer.


CW 2 java - Word (Product Activation Failed)

BlueJ: CW1

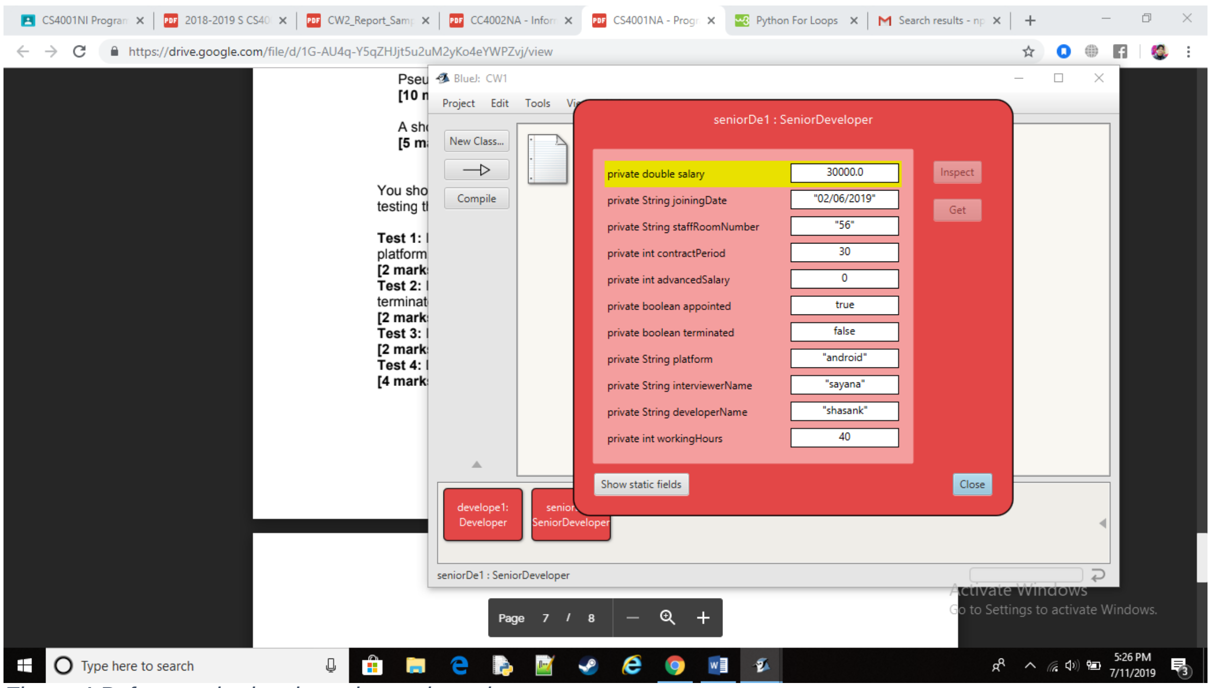
seniorDe1 : SeniorDeveloper

private double salary	30000.0
private String joiningDate	"02/06/2019"
private String staffRoomNumber	"56"
private int contractPeriod	30
private int advancedSalary	0
private boolean appointed	true
private boolean terminated	false
private String platform	"android"
private String interviewerName	"sayana"
private String developerName	"shasank"
private int workingHours	40

Figure 3 After developer is appointed

Inspection after developer being appointed.

Test 2:


https://drive.google.com/file/d/1G-AU4q-Y5qZHJjt5u2uM2yKo4eYWPZvj/view

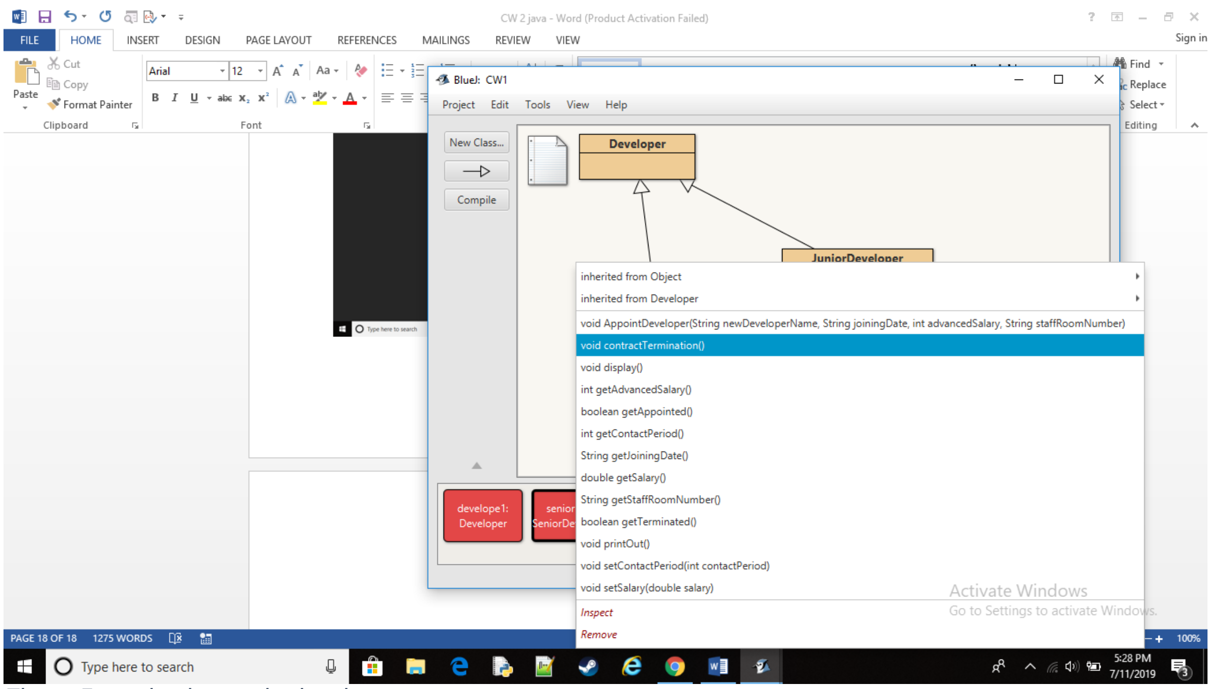
BlueJ: CW1

seniorDe1 : SeniorDeveloper

private double salary	30000.0
private String joiningDate	"02/06/2019"
private String staffRoomNumber	"56"
private int contractPeriod	30
private int advancedSalary	0
private boolean appointed	true
private boolean terminated	false
private String platform	"android"
private String interviewerName	"sayana"
private String developerName	"shasank"
private int workingHours	40

Figure 4 Before seniordeveloper is terminated

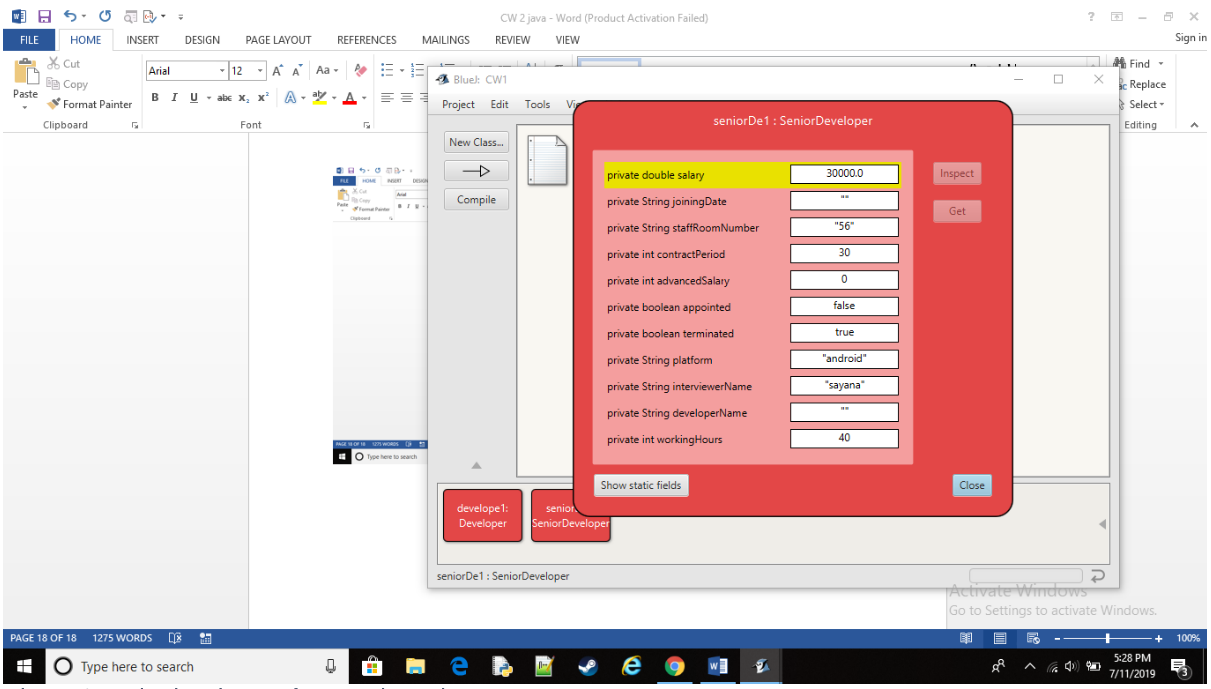
Inspection SeniorDeveloper class before terminated



inherited from Object
inherited from Developer
void AppointDeveloper(String newDeveloperName, String joiningDate, int advancedSalary, String staffRoomNumber)
void contractTermination()
void display()
int getAdvancedSalary()
boolean getAppointed()
int getContactPeriod()
String getJoiningDate()
double getSalary()
String getStaffRoomNumber()
boolean getTerminated()
void printOut()
void setContactPeriod(int contactPeriod)
void setSalary(double salary)

Figure 5 terminating seniordeveloper

Calling method contractTermination.



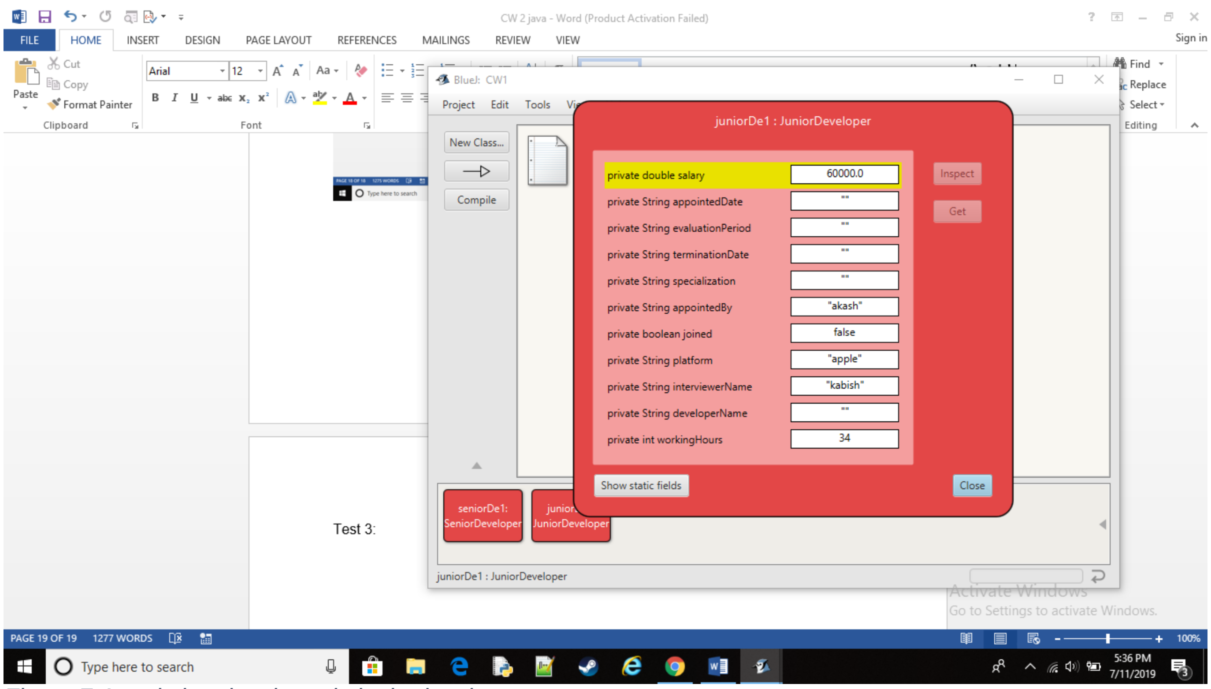
seniorDe1 : SeniorDeveloper

private double salary	30000.0
private String joiningDate	""
private String staffRoomNumber	"56"
private int contractPeriod	30
private int advancedSalary	0
private boolean appointed	false
private boolean terminated	true
private String platform	"android"
private String interviewerName	"sayana"
private String developerName	""
private int workingHours	40

Figure 6 seniordeveloper after terminated

Inspection of SeniorDeveloper Class after being terminated.

Test 3:

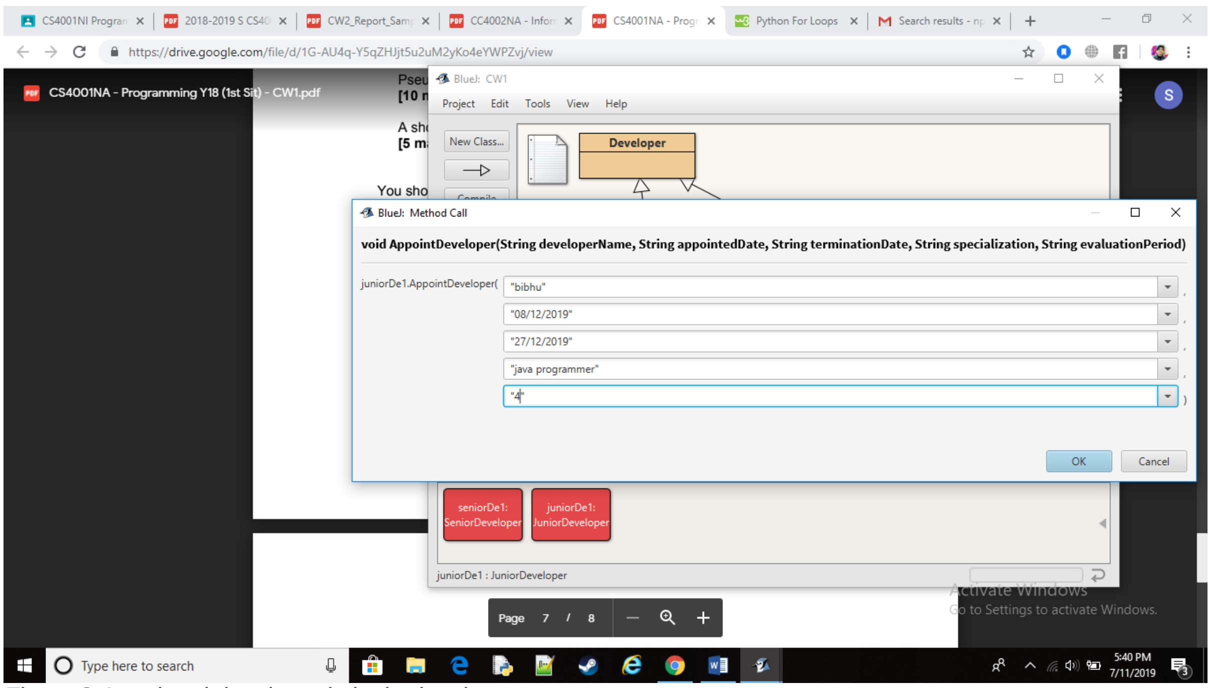

juniorDe1 : JuniorDeveloper

private double salary	60000.0
private String appointedDate	""
private String evaluationPeriod	""
private String terminationDate	""
private String specialization	""
private String appointedBy	"akash"
private boolean joined	false
private String platform	"apple"
private String interviewerName	"kabish"
private String developerName	""
private int workingHours	34

Inspect
Get
Show static fields
Close
seniorDe1: SeniorDeveloper
juniorDe1: JuniorDeveloper
juniorDe1 : JuniorDeveloper

Figure 7 Appointing developer in juniordeveloper

Inspection of JuniorDeveloper before appointing developer


BlueJ: Method Call

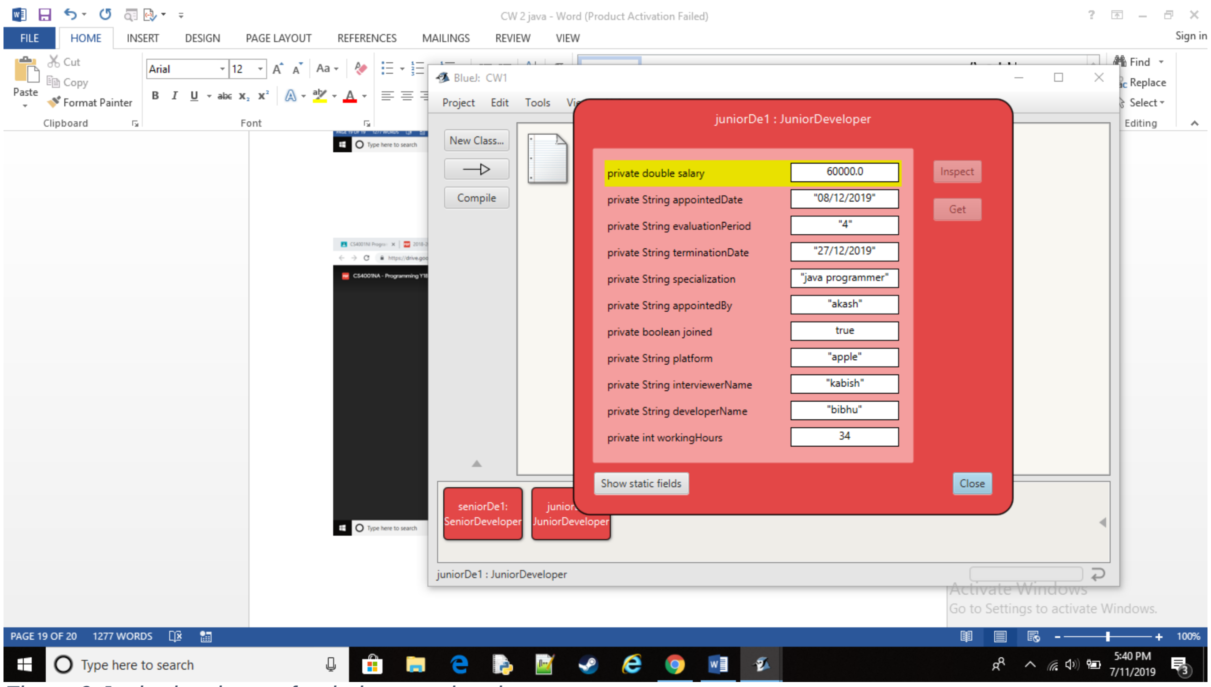
void AppointDeveloper(String developerName, String appointedDate, String terminationDate, String specialization, String evaluationPeriod)

```
juniorDe1.AppointDeveloper( "bibhu"
                            "08/12/2019"
                            "27/12/2019"
                            "java programmer"
                            "4"                )
```

Developer
OK
Cancel
seniorDe1: SeniorDeveloper
juniorDe1: JuniorDeveloper
juniorDe1 : JuniorDeveloper

Figure 8 Appointed developer in juniordeveloper

Appointing developer to JuniorDeveloper class



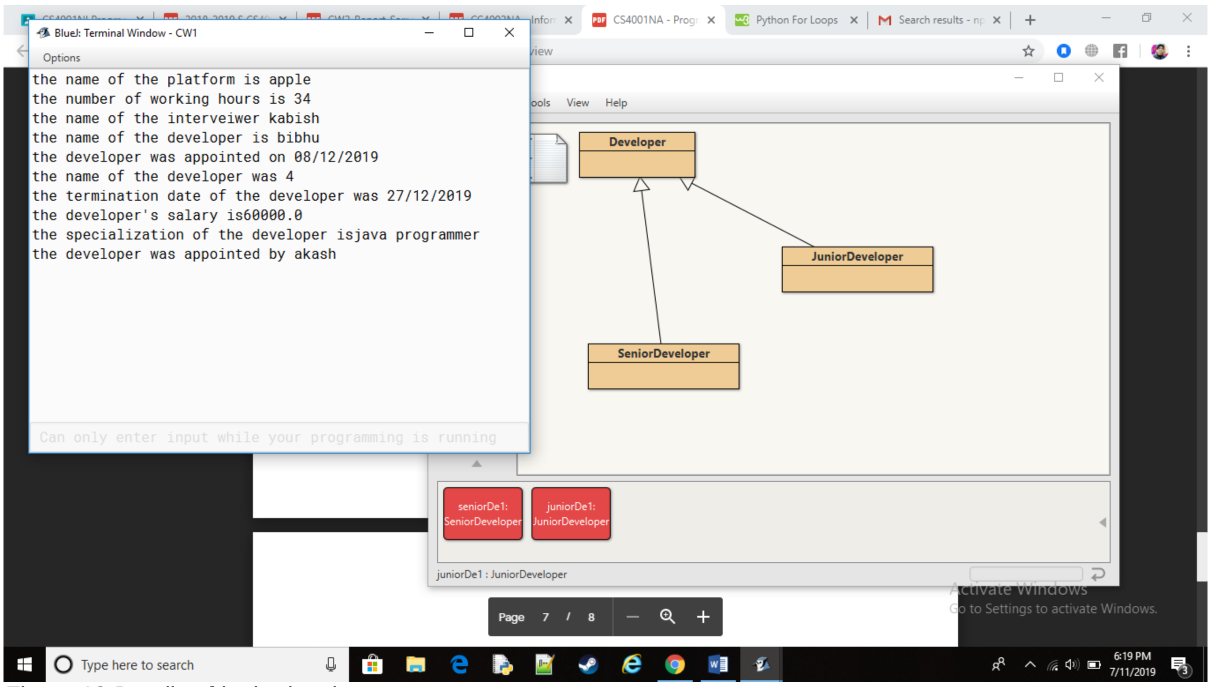
juniorDe1 : JuniorDeveloper

private double salary	60000.0
private String appointedDate	"08/12/2019"
private String evaluationPeriod	"4"
private String terminationDate	"27/12/2019"
private String specialization	"java programmer"
private String appointedBy	"akash"
private boolean joined	true
private String platform	"apple"
private String interviewerName	"kabish"
private String developerName	"bibhu"
private int workingHours	34

Figure 9 Juniordeveloper after being appointed

Inspection after appointing developer in Junior Developer class

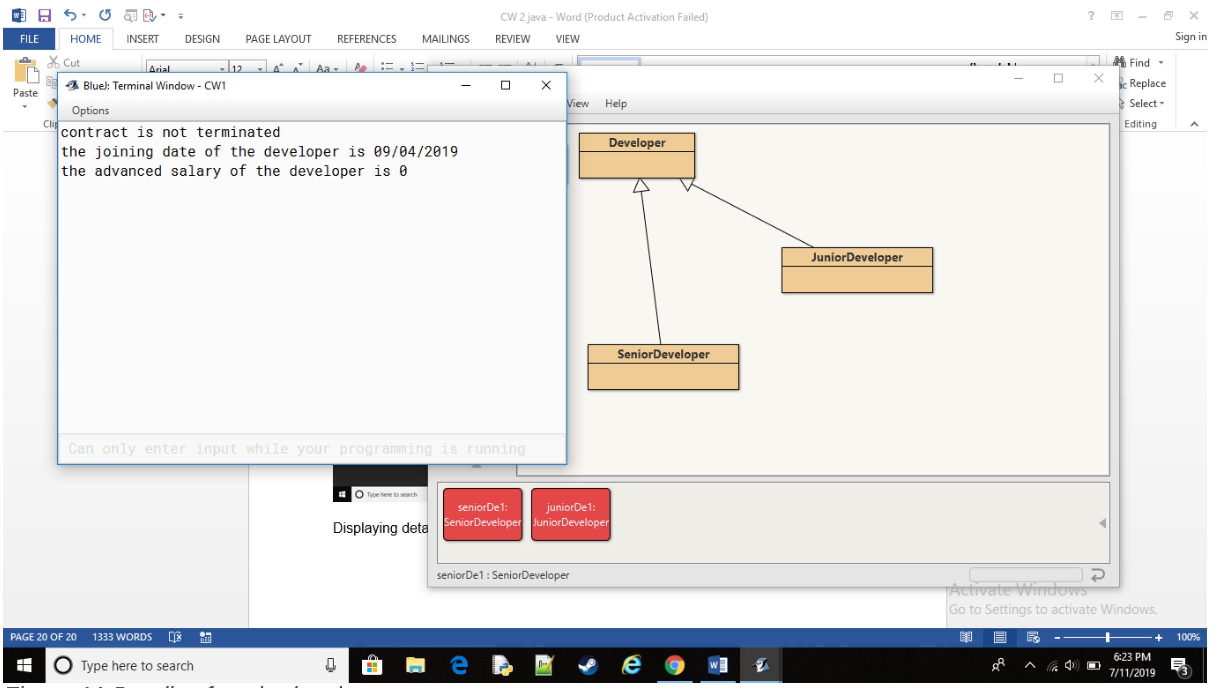
Test 4:



```
the name of the platform is apple
the number of working hours is 34
the name of the interveiwer kabish
the name of the developer is bibhu
the developer was appointed on 08/12/2019
the name of the developer was 4
the termination date of the developer was 27/12/2019
the developer's salary is60000.0
the specialization of the developer isjava programmer
the developer was appointed by akash
```

Figure 10 Details of juniordeveloper

Displaying details of Junior Developer class



```
contract is not terminated
the joining date of the developer is 09/04/2019
the advanced salary of the developer is 0
```

BlueJ: Terminal Window - CW1
Options
Can only enter input while your programming is running
Developer
JuniorDeveloper
SeniorDeveloper
seniorDe1: SeniorDeveloper
juniorDe1: JuniorDeveloper
seniorDe1 : SeniorDeveloper
Displaying deta
CW 2 java - Word (Product Activation Failed)
PAGE 20 OF 20 1333 WORDS
Type here to search
6:23 PM 7/11/2019

Figure 11 Details of seniordeveloper

Displaying details of Senior Developer class

Appendix

```
public class Developer
{
    private String platform;
    private String interviewerName;
    private String developerName;
    private int workingHours;

    //Constructor method
    public Developer(String platform,String interviewerName,int workingHours)
    {
        this.platform = platform;
        this.interviewerName = interviewerName;
        this.developerName = "";
        this.workingHours = workingHours;
    }

    //accessor methods
    public String getPlatform()
    {
        return this.platform;
    }

    public String getInterviewerName()
```

```
{
    return this.interviewerName;
}

public String getDeveloperName()
{
    return this.developerName;
}

public int getWorkingHours()
{
    return this.workingHours;
}

//setting the developer name
public void setDeveloperName(String developerName)
{
    this.developerName = developerName;
}

//displaying the destails of the developer class
public void display()
{
    System.out.println("the name of the platform is " + getPlatform());
```

```
        System.out.println("the number of working hours is " + getWorkingHours());
        System.out.println("the name of the interveiwer " + getInterviewerName());
        if (this.developerName != ""){
            System.out.println("the name of the developer is " + getDeveloperName());
        }
    }

}



public class SeniorDeveloper extends Developer
{
    private double salary;
    private String joiningDate;
    private String staffRoomNumber;
    private int contractPeriod;
    private int advancedSalary;
    private boolean appointed;
    private boolean terminated;

    public SeniorDeveloper(int salary,int contractPeriod,String platform,String interviewerName,int workingHours)
    {
        super(platform,interviewerName,workingHours);
        this.salary = salary;
        this.joiningDate = "";

        this.staffRoomNumber = "";

        this.contractPeriod = contractPeriod;

        this.advancedSalary = 0;

        this.appointed = false;

        this.terminated = false;

    }


    public double getSalary()

    {

        return this.salary;

    }


    public String getJoiningDate()

    {

        return this.joiningDate;

    }


    public String getStaffRoomNumber()

    {

        return this.staffRoomNumber;

    }


    public int getContactPeriod()
```

```
{
    return this.contractPeriod;
}

public int getAdvancedSalary()
{
    return this.advancedSalary;
}

public boolean getAppointed()
{
    return this.appointed;
}

public boolean getTerminated()
{
    return this.terminated;
}

public void setSalary(double salary)
{
    this.salary = salary;
}
```

```
    public void setContactPeriod(int contactPeriod)
    {
        this.contractPeriod = contractPeriod;
    }

    public void AppointDeveloper(String newDeveloperName,String joiningDate,int
advancedSalary,String staffRoomNumber)
    {
        if(this.appointed){
            System.out.println(getDeveloperName() + "is the name of the developer and
their staff room number is " + getStaffRoomNumber());

        }
        else{
            super.setDeveloperName(newDeveloperName);
            this.joiningDate=joiningDate;
            this.staffRoomNumber=staffRoomNumber;
            this.appointed=true;
            this.terminated=false;
        }

    }

    public void contractTermination()
    {
        if(this.terminated){
            System.out.println("contract already terminated");
        }
        else{
            super.setDeveloperName("");
            this.joiningDate="";
            this.advancedSalary=0;
            this.appointed=false;
            this.terminated=true;
        }
    }

    public void printOut()
    {
        System.out.println("the name of the platform is "+getPlatform());
        System.out.println("the name of the interviewer is "+getInterviewerName());
        System.out.println("the salary is "+getSalary());
    }

    public void display()
    {
        //display();
        if(this.appointed){
            if(this.terminated){
```

```
            System.out.println("contract is terminated");
         }
         else{
            System.out.println("contract is not terminated");
         }
         System.out.println("the joining date of the developer is " + getJoiningDate());
         System.out.println("the advanced salary of the developer is " + 
getAdvancedSalary());
      }
   }
}
public class JuniorDeveloper extends Developer
{
   private double salary;
   private String appointedDate;
   private String evaluationPeriod;
   private String terminationDate;
   private String specialization;
   private String appointedBy;
   private boolean joined;

   public JuniorDeveloper(String platform,String interviewerName,int workingHours,int 
salary,String appointedBy)
   {
      super(platform,interviewerName,workingHours);
      this.salary=salary;
      this.appointedDate="";
      this.evaluationPeriod="";
      this.terminationDate="";
      this.specialization="";
      this.appointedBy=appointedBy;
      this.joined=false;
   }

   public double getSalary()
   {
    return this.salary;
   }

   public String getAppointedDate()
   {
    return this.appointedDate;
   }

   public String getEvalutionPeriod()
   {
    return this.evaluationPeriod;
   }
```

```
public String getTerminationDate()
{
 return this.terminationDate;
}

public String getSpecialization()
{
 return this.specialization;
}

public String getAppointedBy()
{
 return this.appointedBy;
}

public boolean getJoined()
{
 return this.joined;
}

public void setSalary(double salary)
{
    if(this.joined==false){
            this.salary=salary;
        }
        else{
            System.out.println("Not possible to change the salary");
        }
    }

    public void AppointDeveloper(String developerName,String appointedDate,String
terminationDate,String specialization, String evaluationPeriod)
    {
        if(this.joined==false){
            super.setDeveloperName(developerName);
            this.appointedDate=appointedDate;
            this.terminationDate=terminationDate;
            this.specialization=specialization;
            this.evaluationPeriod = evaluationPeriod;
            this.joined=true;
        }
        else{
            System.out.println("Developer already appointed");
        }

    }

    public void displays()
    {
        super.display();
        if(this.joined){
            System.out.println("the developer was appointed on " + getAppointedDate());
            System.out.println("the name of the developer was " + getEvalutionPeriod() );
            System.out.println("the termination date of the developer was " + getTerminationDate());
            System.out.println("the developer's salary is" + getSalary());
            System.out.println("the specialization of the developer is" + getSpecialization());
            System.out.println("the developer was appointed by " + getAppointedBy());
        }
    }
```

Conclusion

I would like to thank all the teachers of this module for helping me understand the concepts and guiding me though out the semester. Comparatively I have found course work of java easier because we did not have to put too much logic behind it. We just had to assign all the variable with their given data types. What I found confusing was the concept of constructor and parent child class. I am glad that I did get help from my teachers on this.

Form this course work I learned the concept of parent and child class. The child class can access the attributes of parent class. What was so different about java was when we make objects it used to make something similar to GUI feature. Where we could assign values to different attributes and access different method from it. This course work was similar to making forms too hence I got the idea of making a form which can be made more interactive later on with advanced programming skills.