LONDON METROPOLITAN UNIVERSITY

ITAHARI INTERNATIONAL COLLEGE

Module Code & Module Title

CC4057NT Programming

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2023 Autumn

Student Name: Anmol Poudyal

London Met ID: 23049190

College ID: np05cp4a230006

Assignment Due Date: Jan 26, 2024

Assignment Submission Date: Jan 24, 2024

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 mark of zero will be awarded.

Table of Contents

Table of figures

Table of tables

1 Introduction

Java is one of the most used programming languages in the world it is specially used for developing web applications and other various programs. Java itself is a platform which has tons of various features like multi- platform, object-orientation as well as network centric programming language. It has been popular among developers for 20 years and still is. It is a fast, secure, reliable programming language for building varieties of things from mobile apps and enterprise software to big data applications and server-side technologies. (AWS, 2023)

Figure 1: Java logo

In this course work I have used blue-j as Integrated Development Environment (IDE). Blue-j gives you the development environment which helps us to write compile and run a program easily. Blue-j is one of the mostly used IDE among college and university students Here I had created 3 classes one of which is a parent class and others 2 are child classes. In this course work I have used concepts like method overriding, Concepts of inheritance and also have used encapsulations.
For the documentation purpose I have used Microsoft word which is one of the most used word processor.

2 Class diagram

A class diagram is a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects (visual-paradism, 2024)

2.1 Class diagram of teacher class

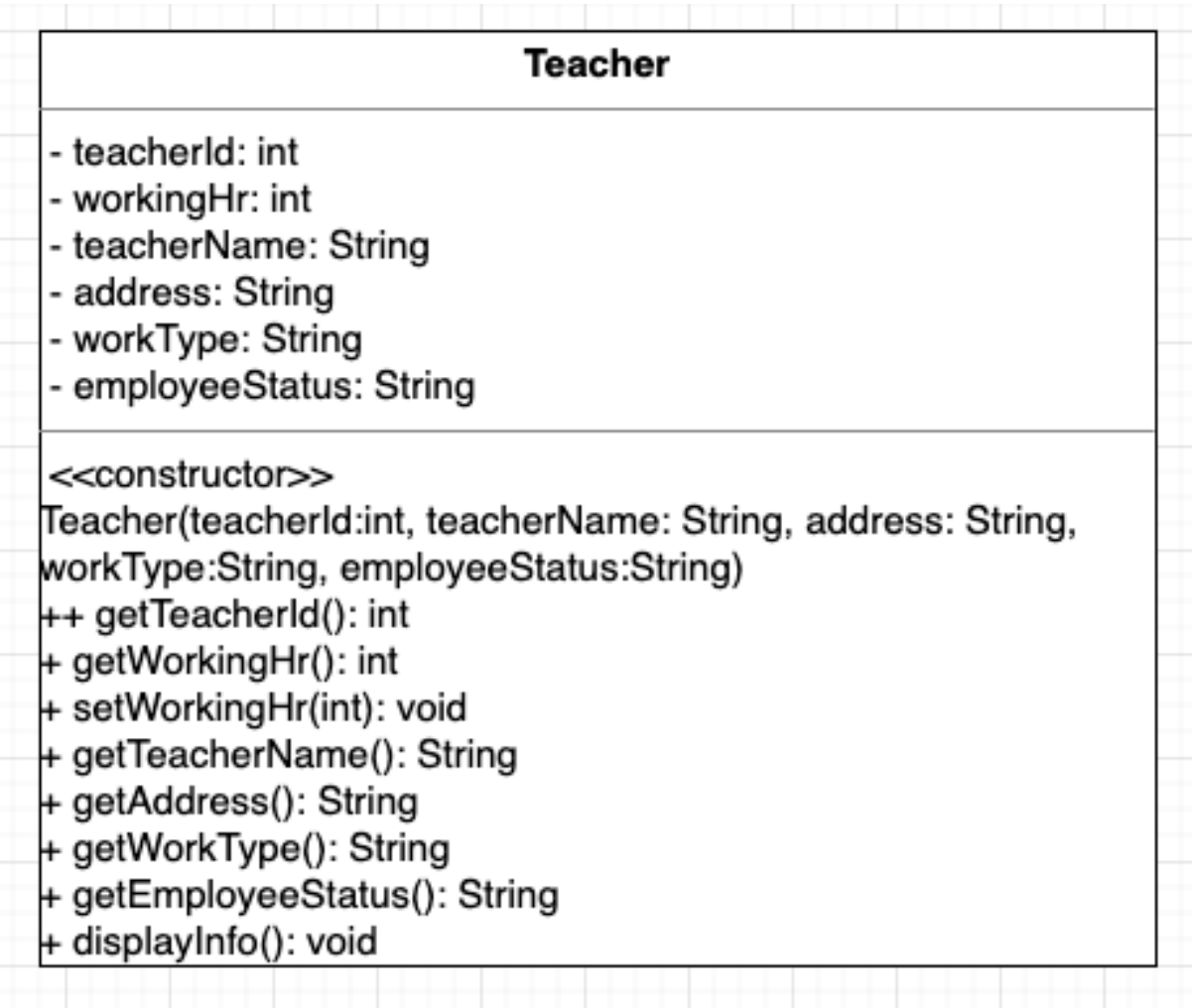

Figure 2: class diagram of teacher from draw.io

2.2 Class diagram of Lecturer class

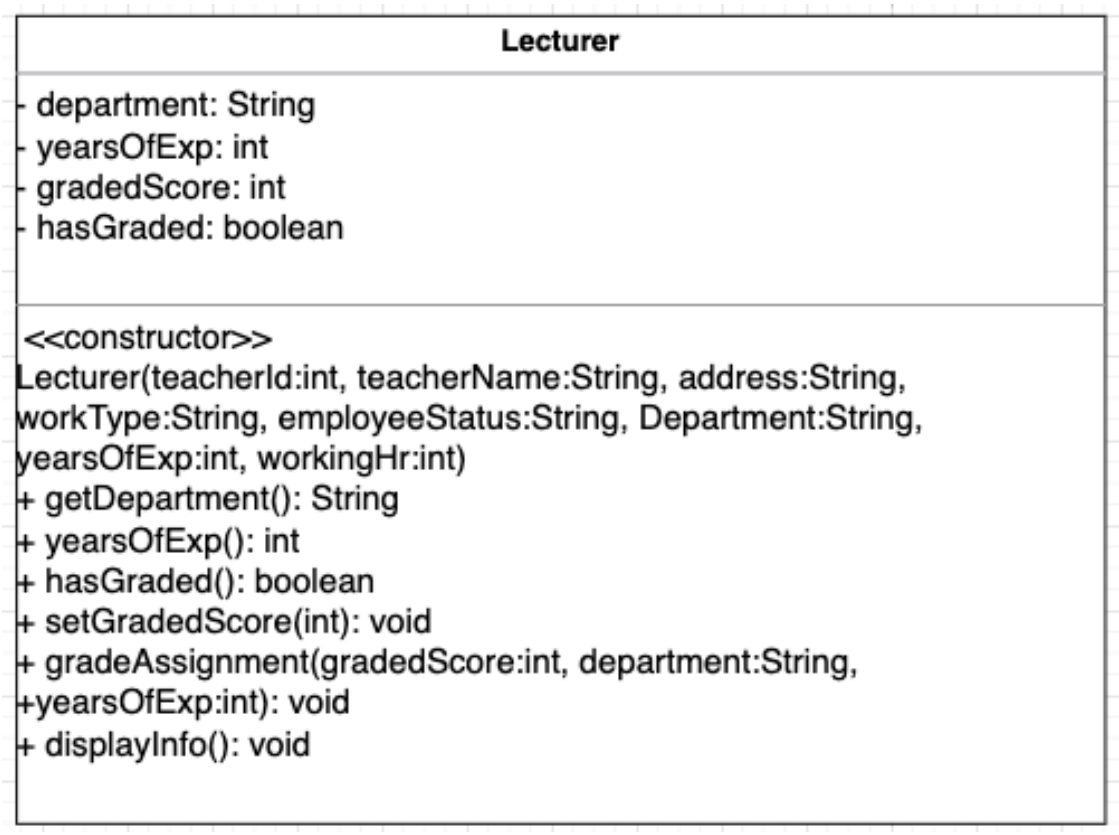

Figure 3: class diagram of Lecture class

2.3 Class diagram of Tutor class

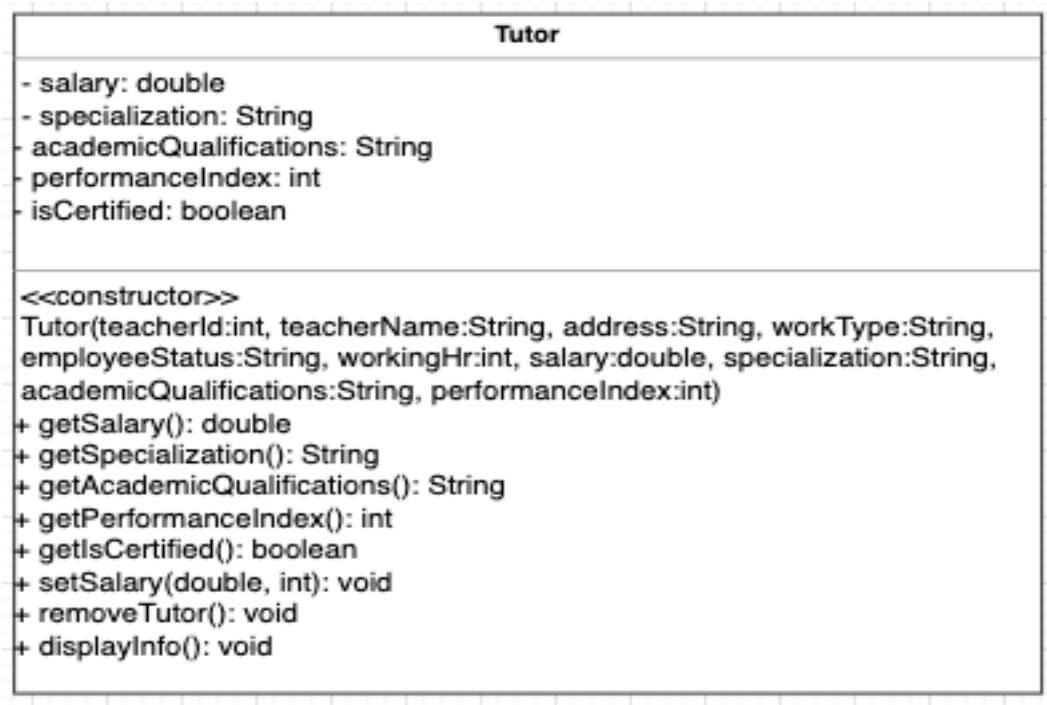

Figure 4: class diagram of Tutor class

2.4 Class diagram in blue-j

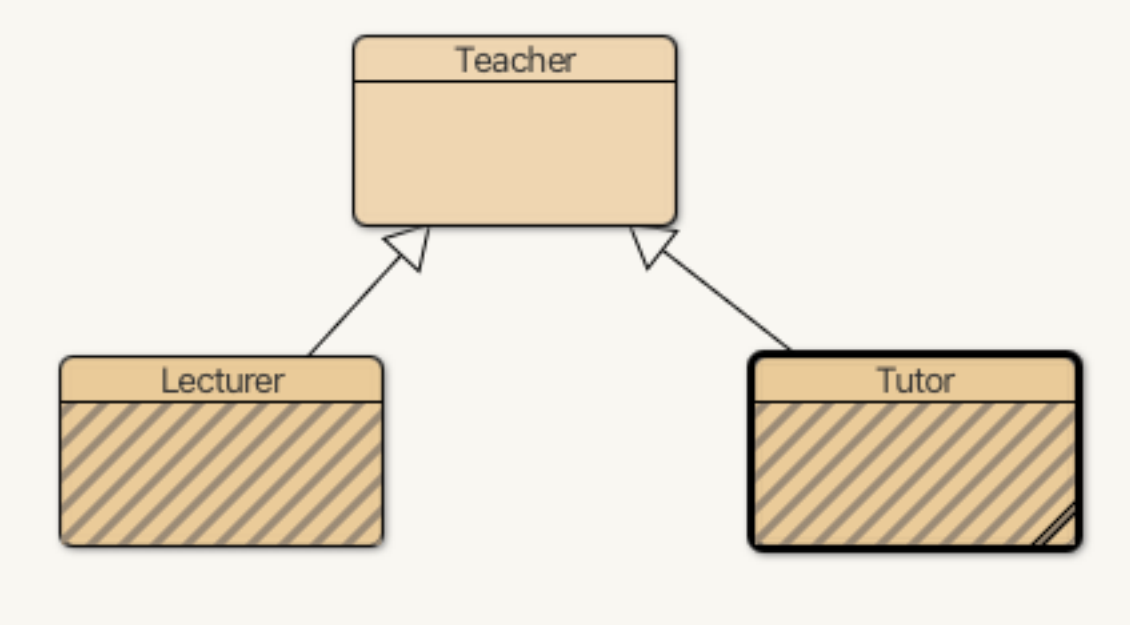

Figure 5: class diagram in blue-j

2.5 Class diagram in draw.io

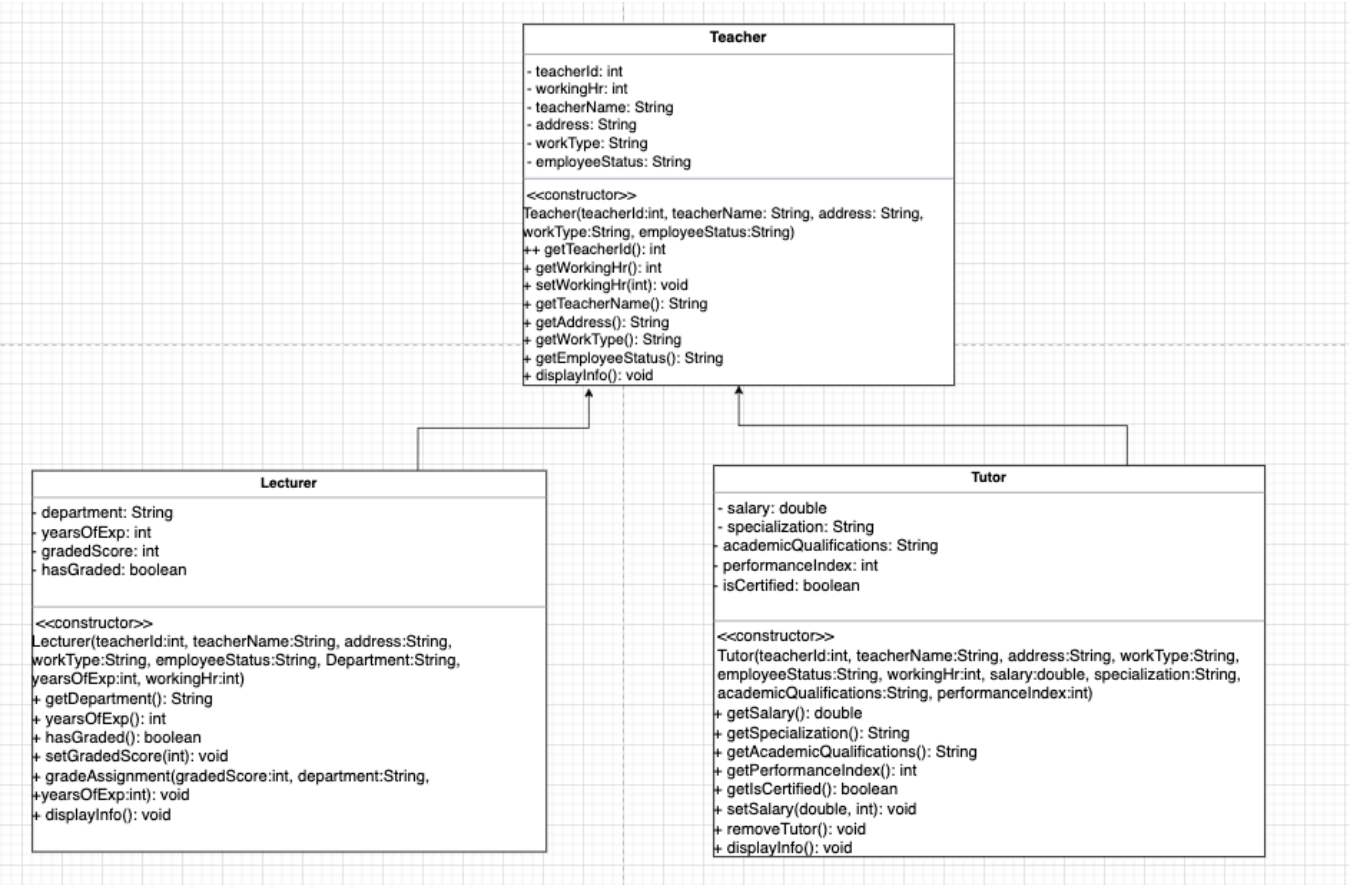

Figure 6: class diagram of from draw.io

3 Pseudocode

Pseudocode is a description of program in simple English language such that even a person from non-programming background could easily understand it.

3.1 Pseudocode of Teacher

CREATE a parent class Teacher

DO

DECLARE private instance variable teacherId as int

DECLARE private instance variable workingHr as int

DECLARE private instance variable teacherName as String

DECLARE private instance variable address as String

DECLARE private instance variable workType as String

DECLARE private instance variable employeeStatus as String

END DO

Create a constructor method Teacher with parameters int teacherId, String teacherName, String address, String workType, String employeeStatus

DO

INITIALIZE the instance teacherId with parameter of the constructor

INITIALIZE the instance teacherName with parameter of the constructor

INITIALIZE the instance address with parameter of the constructor

INITIALIZE the instance workType with parameter of the constructor

INITIALIZE the instance employeeStatus with parameter

END DO

CREATE an accessor method getTeacherId() that returns int value

DO

RETURN teacherId

END DO

CREATE an accessor method getTeacherName() that returns String value

DO

RETURN teacherName

END DO

CREATE an accessor method getAddress() that returns String value

DO

RETURN address

END DO

CREATE an accessor method getWorkType() that returns String value

DO

RETURN workType

END DO

CREATE an accessor method getEmployeeStatus() that returns String value

DO

RETURN employeeStatus

END DO

CREATE an accessor method getWorkingHr() that returns int value

DO

RETURN workingHr

END DO

CREATE a setter method setWorkingHr that has return type void and accepts parameter int workingHr

DECLARE displayInfo() method that has return type void

PRINT value of teacherId

PRINT value of teacherName

PRINT value of address

```
    PRINT value of workType
    PRINT value of employeeStatus
    IF workinghr is not set then
    PRINT value not assigned
    ELSE
    PRINT Value of workinghr
END IF
END DO
```

3.2 Pseudocode of Lecturer

```
CREATE child class Lecturer
DO
    DECLARE private instance variable department as String
    DECLARE private instance variable YearsOfExp as int
    DECLARE private instance variable gradedScore as int
    DECLARE private instance variable hasGraded as Boolean

    Create a constructor method Lecturer with parameters int teacherId, String
    teacherName, String address, String workType, String employeeStatus, String
    Department, int yearsOfExp, int workingHr
    DO
```

CALL the super class constructor

SET workinghr with parameter of Lecturer class

INITIALIZE the instance department with parameter of the constructor

INITIALIZE the instance YearsOfExp with parameter of the constructor

INITIALIZE the instance gradedScore to zero

INITIALIZE the instance hasGraded to false

END DO

CREATE an accessor method getDepartment () that returns String value

DO

RETURN department

END DO

CREATE an accessor method yearsOfExp() that returns int value

DO

RETURN yearsOfExp

END DO

CREATE an accessor method hasGraded () that returns int value

DO

RETURN hasGraded

END DO

CREATE a setter method that setGradedScore has return type void and accepts parameter int graded Score

CREATE gradeAssignment method that has return type void and accepts parameter int gradedScore, String department and int yearsOfExp

IF years of experience is greater than or equal to 5 and of same department

IF gradedscore is greater than or equal to 70 and graded score is smaller than or equal to 100

DO

PRINT “you have achieved A”

UPDATE the gradedscore with instance value

INITIALIZE the value of instance hasGraded to True

END DO

END IF

ELSE IF gradedscore is greater than or equal to 60

DO

PRINT “you have achieved B”

UPDATE the gradedscore with instance value

INITIALIZE the value of instance hasGraded to True

END DO

END ELSE

ELSE IF gradedscore is greater than or equal to 50

DO

PRINT “you have achieved C”

UPDATE the gradedscore with instance value

INITIALIZE the value of instance hasGraded to True

END DO

END ELSE

ELSE IF gradedscore is greater than or equal to 40

DO

PRINT “you have achieved D”

```
            UPDATE the gradedscore with instance value
            INITIALIZE the value of instance hasGraded to True
            END DO
        END ELSE
        ELSE IF gradedscore is less than 40
            DO
            PRINT "you have achieved E"
            UPDATE the gradedscore with instance value
            INITIALIZE the value of instance hasGraded to True
            END DO
       END ELSE



        ELSE
            DO
            PRINT wrong entry
            END DO
        END ELSE
    END IF
    ELSE
        PRINT grading failed only lecturer has access
    CREATE method displayInfo ()
    DO
```

```
        CALL displayInfo method from super class
        PRINT value of department
        PRINT value of yearsOfExp
        IF hasGraded is true
            DO
            PRINT value of gradedScore
            END DO
        ELSE
            DO
            PRINT assignment is not graded
            END DO
        END ELSE
    END method
```

3.3 Pseudocode of tutor class

```
CREATE child class Tutor
DO
    DECLARE private instance variable salary as double
    DECLARE private instance variable specialization as String
    DECLARE private instance variable acadamicQualifications as String
    DECLARE private instance variable performanceIndex as int
    DECLARE private instance variable isCertified as Boolean

    Create a constructor method Lecturer with parameters int teacherId, String
    teacherName, String address, String workType, String employeeStatus, int
```

workingHr, double salary, String specialization, String academicQualifications, int performanceIndex

DO

CALL superclass constructor

SET workingHr with the parameter of the constructor

INITIALIZE the instance salary with parameter of the constructor

INITIALIZE the instance specialization with parameter of the constructor

INITIALIZE the instance academicQualifications with parameter of the constructor

INITIALIZE the instance performanceIndex with parameter of the constructor

INITIALIZE the instance isCertified to false

END DO

CREATE an accessor method getSalary() that returns double value

DO

RETURN Salary

END DO

CREATE an accessor method getSpecialization () that returns String value

DO

RETURN specialization

END DO

CREATE an accessor method getAcademicQualifications () that returns String value

DO

RETURN academicQualifications

END DO

CREATE an accessor method getPerformanceIndex () that returns int value

DO

RETURN performanceIndex

END DO

CREATE an accessor method getIsCertified () that returns Boolean value

DO

RETURN isCertified

END DO

CREATE a public method setSalary that has return type void

DO

IF performance index is greater than or equal to 5 and workinghr greater than 20

DO

IF performance index is greater than or equal to 5 and performance index less than or equal to 7

DO

INNITIALIZE salary with (Salary + ((5/100f)*Salary)

UPDATE is certified to true

END DO

END IF

```
            ELSE IF performance is greater than or equal to 8 and
            performance index less than or equal to 9
                DO
                INNITIALIZE salary with (Salary + ((10/100f)*Salary)
                UPDATE iscertified to true
                END DO
            END ELSE
            ELSE IF performance is equal to 10
                DO
                INNITIALIZE salary with (Salary + ((20/100f)*Salary)
                UPDATE iscertified to true
                END DO
            END ELSE
            ELSE
                PRINT invalid result
            END ELSE
        END IF
    END DO
    ELSE
            PRINT salary cannot be approved
    END ELSE
END DO
```

```
CREATE method removeTutor() that has return type void
DO
    IF is certified is false
    DO
        INNITILIZE salary to null
        INNITILIZE specialization to null
        INNITILIZE academicQualificationsto null
        INNITILIZE performanceIndex to null
        PRINT removed tutor
        INNITIALIZE iscertified to false
    END DO
END DO

CREATE method displayInfo() has return type void
DO
    CALL superclass method displayInfo
```

```
            IF is certified is true
            DO
                PRINT salary
                PRINT specialization
                PRINT academicQualifications
                PRINT performanceIndex
            END DO
        END DO
END DO
```

4 Method Description

Here the method descriptions are given

4.1 Method description for Teacher class

Table 1: method description of Teacher class

Method	Description
Teacher (int, String, String, String, String)	It initializes instance variable of the class
getTeacherId ()	Returns the value of variable TeacherId that is in integer.
getTeacherName ()	Returns the value of variable teacherName that is in string.

getAddress ()	Returns the value of variable address that is in String
getWorkType ()	Returns the value of variable work type that is in String
getEmployeeStatus ()	Returns the value of variable employee status that is in integer.
getWorkingHr ()	Returns the value of variable employee status that is in integer.
setWorkingHr (int)	It takes the integer value as a parameter and initializes the variable workinhHr.
displayInfo ()	It displays values of teacher Id, teacherName, address, workType and employeeStatus also if workinghr is equal to zero than display suitable message else it displays values of workingHr

4.2 Method description for Lecturer class

Table 2: method description of lecture class

Lecturer (int, String, String, String, String, String, int, int)	This method is a constructor initializes different variable of the class with the respective parameters
getDepartment ()	This method returns the value of variable getdepartment that is in String
yearsOfExp ()	This method returns the value of variable yearsOfExp that is in integer

hasGraded ()	This method returns the value of variable hasGraded that is in Boolean.
setGradedScore (int)	This method takes integer value as a parameter and initializes value to gradedScore.
gradeAssignment (int, String, int)	This method accepts parameters gradedScore, department and yearsOfExp. If years of experience is greater or equal to 5 and department are same than other condition is executed that is if graded score is between 70 to 100 then it will print suitable message and set the graded score and also updates hasgraded to true else if between 60 to 70 then then it will print suitable message and set the graded score and also updates hasgraded to true else if between 50 to 60 then then it will print suitable message and set the graded score and also updates hasgraded to true and if between 40 to 50 then it will print suitable message and set the graded score and also updates hasgraded to true else if between 0 to 40 then it will print suitable message and set the graded score and also updates hasgraded to true else it will print wrong entry as a suitable message. If years of experience is not greater or equal to 5 and department is not equal than other condition is executed which will return message like
displayInfo ()	This method calls displayInfo method from super class and also displays value of department, years of experience if hasgraded is true it will also display graded score else it will display a suitable message

4.3 Method description for tutor class

Table 3: method description of tutor class

Method	Description
Tutor (int, String, String, String, String, int, double, String, String, int)	This method is a constructor initializes different variable of the class with the respective parameters also it initializes the super class constructor and sets the workinghr of the super class
getSalary ()	This method returns the value of variable salary that is in double
getSpecialization ()	This method returns the value of variable specialization that is in String
getAcademicQualifications ()	This method returns the value of variable academicQualifications that is in String
getPerformanceIndex ()	This method returns the value of variable performanceIndex that is in int
getIsCertified ()	This method returns the value of variable isCertified that is in Boolean
setSalary (double, int)	This method accepts two parameters salary and performance index. This method sets salary if performance index is greater than and working hr greater than 20 then the condition is executed that if performanceIndex is greater than or equal to 5 and performanceIndex is less than or equal to 7 then it increases salary by 5 % and updates iscertified to true else If performanceIndex is greater than or equal to and performanceIndex is less than or equal to 9 then it increases salary by

	10 % and updates iscertified to true. If performance index is equal to 10 then increase salary to 20% and updates iscertified to true else invalid message is displayed and if performance index is not greater than and workinghr is not greater than 20 then suitable message is displayed.
removeTutor ()	This method helps to remove tutor. If iscertified is false it initializes salary , specialization, academicQualifications, performanceIndex to null and display suitable message and also initialize iscertified to false
displayInfo ()	This method calls displayInfo () method of super class is if iscertified then it will display the values of salary, specialization, academicQualifications, and performanceIndex.

5 Testing

5.1 Test-1

Table 4: Inspect the Lecturer class, grade the assignment, and re-inspect the lecture class

Objective	To Inspect the Lecturer class, grade the assignment, and re-inspect the lecture class
Action	Created object of lecturer class and initializing the variables teacherID: 1 teacherName: “Bikram sir” address: “Salakpur” workType: “Teaching”

	employeeStatus: “active” department: “BIT” yearsOfExp: 12 workingHr: 10 Then, inspeced the Lecturer class object Called the gradeAssignment () method to grade the assignment gradedScore: 76 department:”BIT” yearsOfExp: 12 Reinspected the lecturer class after the graded score is set
Expected result	The object should be created and the parameters are displayed while inspecting and also graded score must be set and the hasGrade must be updated to true and also suitable message must be displayed after grading
Actual result	The object was created and the parameters were passed successfully and displayed. Graded score was also set and the Boolean hasGraded was updated to true
Conclusion	The test was successful….

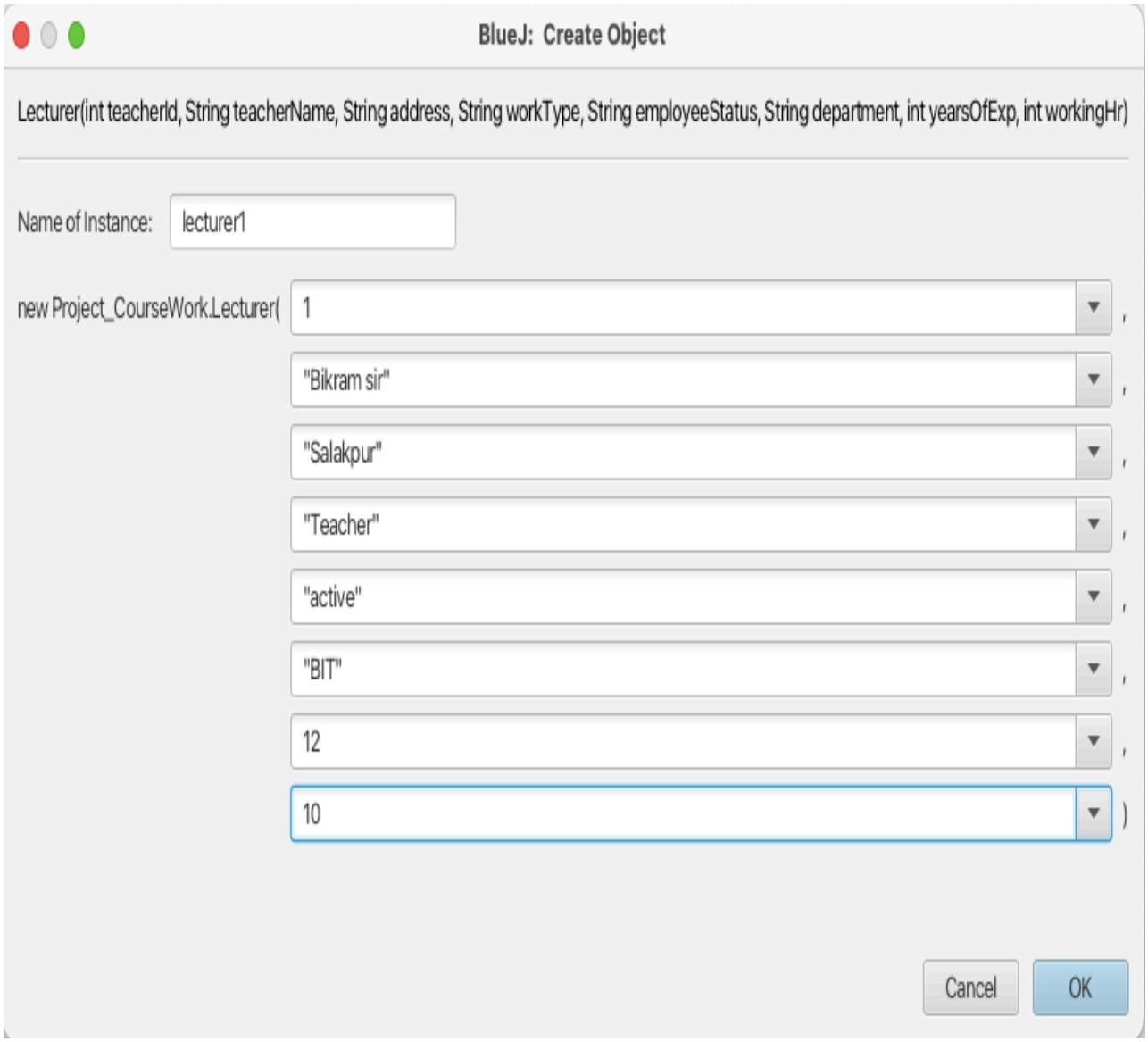

Figure 7: object creation of lecturer class

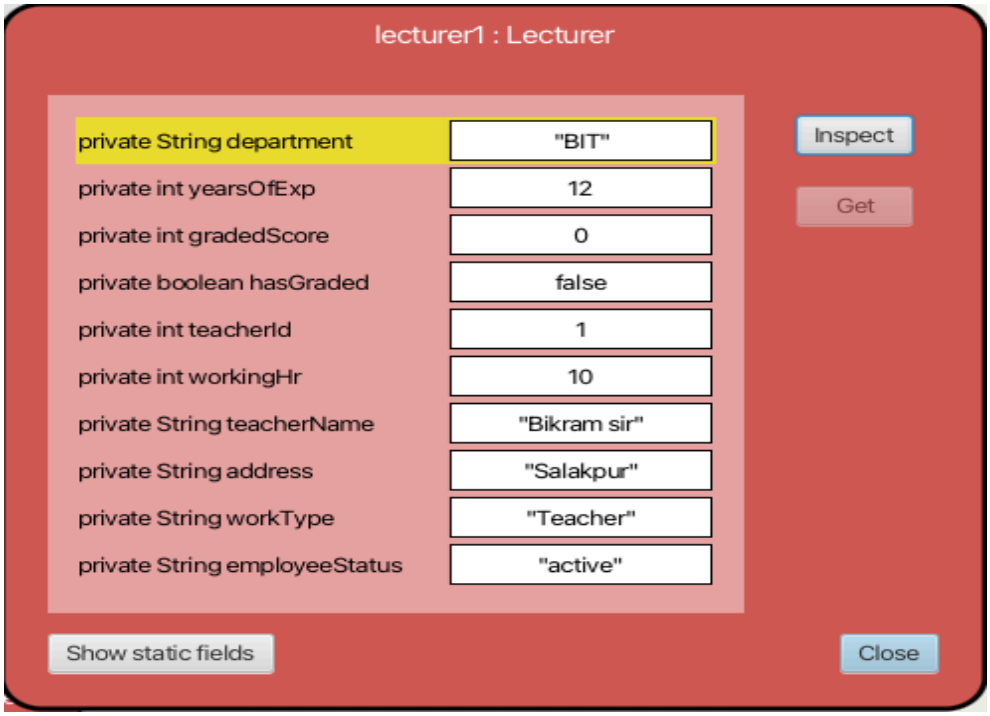

Figure 8: Inspecting lecturer class

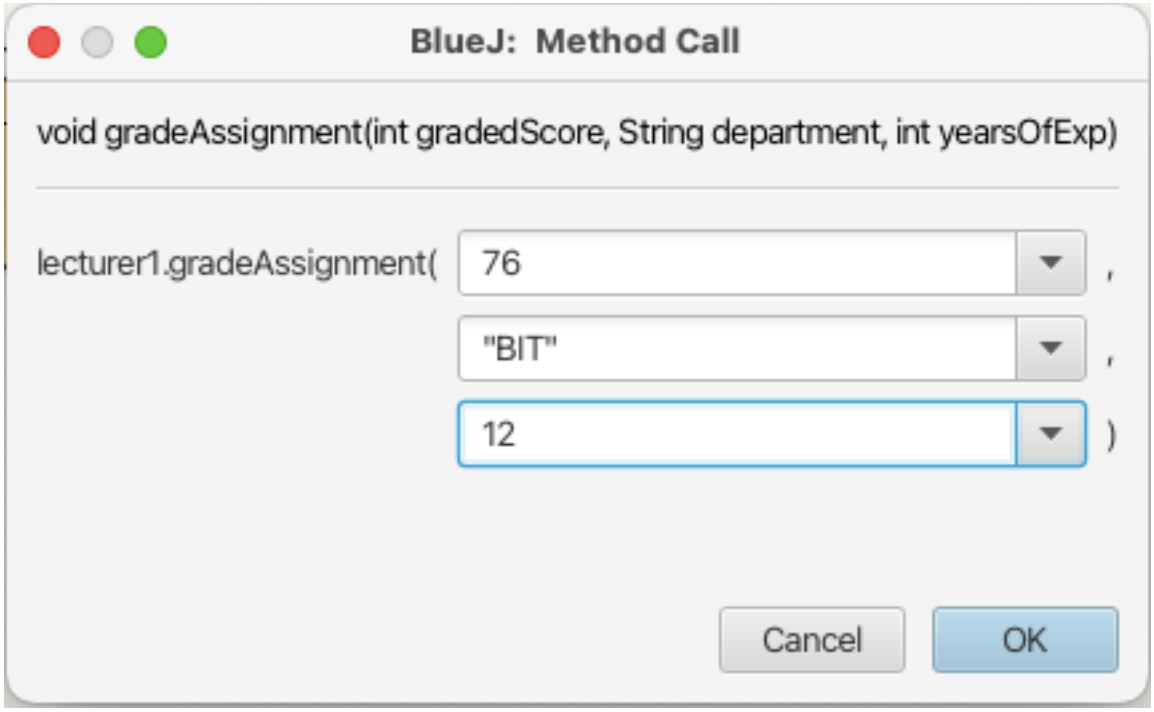

Figure 9: inserting the value of parameter in grade assignment method

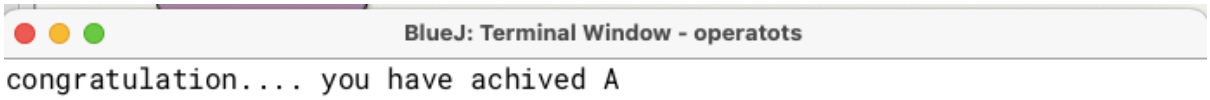

Figure 10: result after grading Score

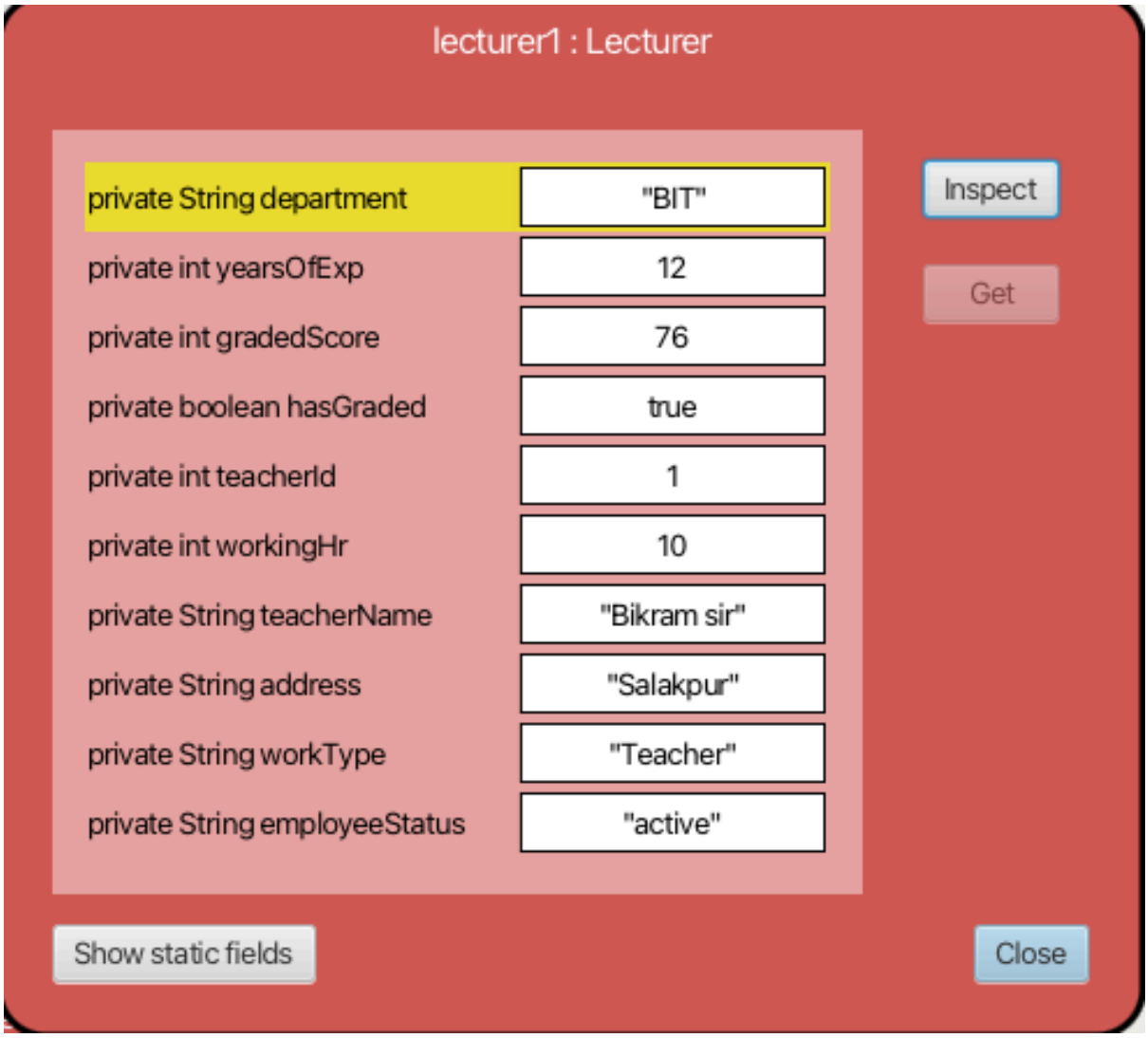

Figure 11: Re-inspecting the lecturer class

5.2 Test 2

Table 5: Inspect Tutor class, set salary and reinspect the Tutor class

Objective	Inspect Tutor class, set salary and reinspect the Tutor class
Action	Created object of tutor class and initialized the variables teacherID: 2 teacherName: “Nikesh sir” address: “Biratnagar” workType: “Teaching” employeeStatus: “active” workinghr: 12 salary: 25000 specialization: “Java” academicQualification: ”Bachelor’s in computing” performanceIndex: 8 called the set salary method to set the salary and passed the parameter salary: 25000 performanceIndex: 8 then re-inspected the tutor class after salary is set

Expected result	The object should be created and the parameters are displayed while inspecting and also salary must be initialized
Actual result	The object was created and the parameters were passed successfully. Salary was also updated
Conclusiomn	The test was successful…

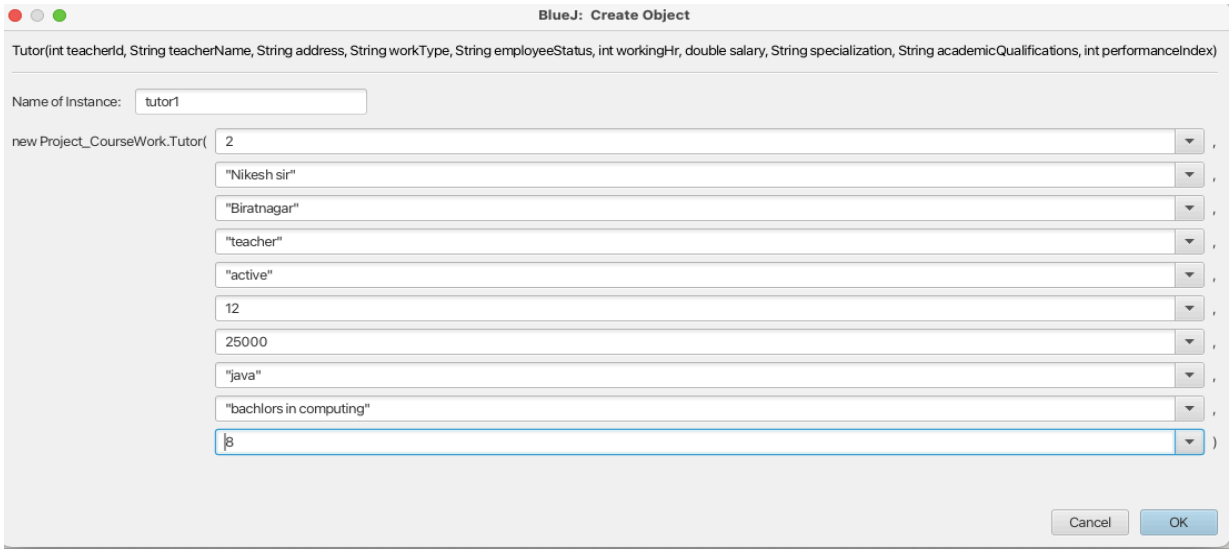

Figure 12: creating an object of tutor class

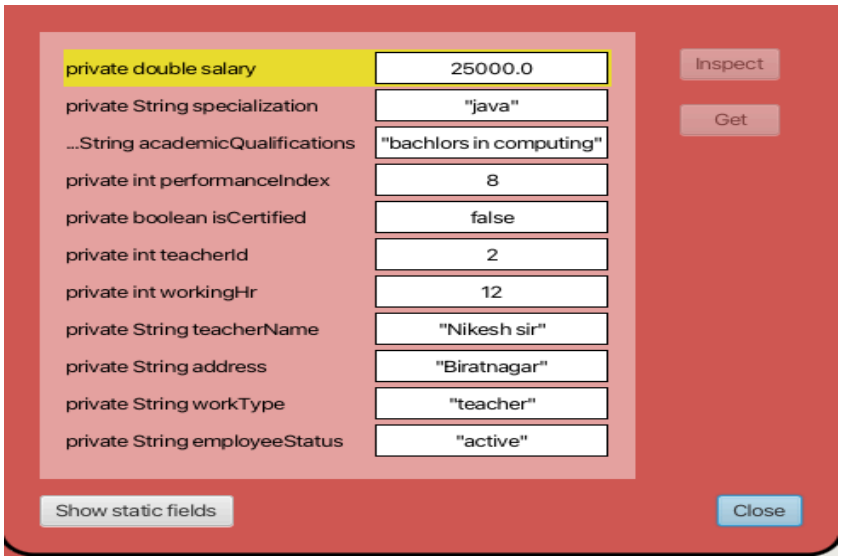

Figure 13: inspecting tutor class

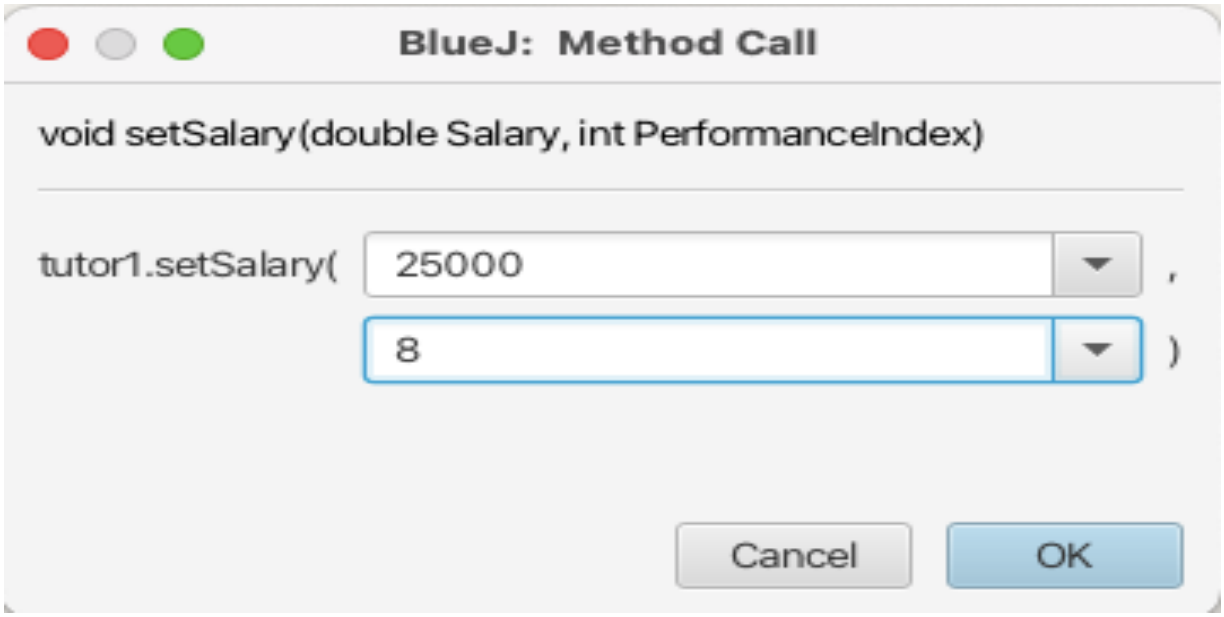

Figure 14: initializing the value of parameters in set salary method

Objective	Inspect Tutor class again after removing the tutor
Action	Used the previous test tutor object and inspected the tutor object after the remove tutor method was executed
Expected result	All the values must be set to null and suitable message must be displayed
Actual result	All the values were initialized to null and suitable message was displayed after tutor was removed
Conclusion	The test was successful…

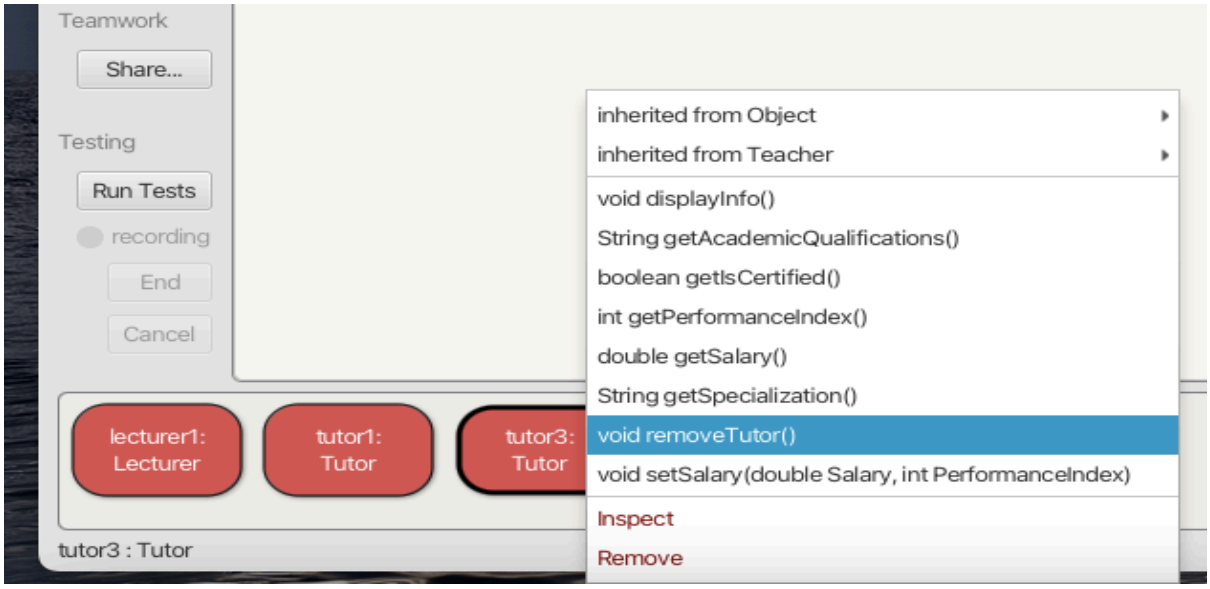

Figure 15: calling remove tutor method

Figure 16: result after the remove tutor method was called

Figure 17: re-inspecting the tutor

Objective	Display the details of lecturer and tutor classes
Actions	Called the display method on the object of the lecturer and tutorial class and displayed the info
Expected result	The objects detailed information must be displayed with the suitable message

	after the display info method will be called.
Actual result	All the information of both lecturer and tutor class were displayed with the suitable message after the display info method was called.
Conclusion	The result was successful…

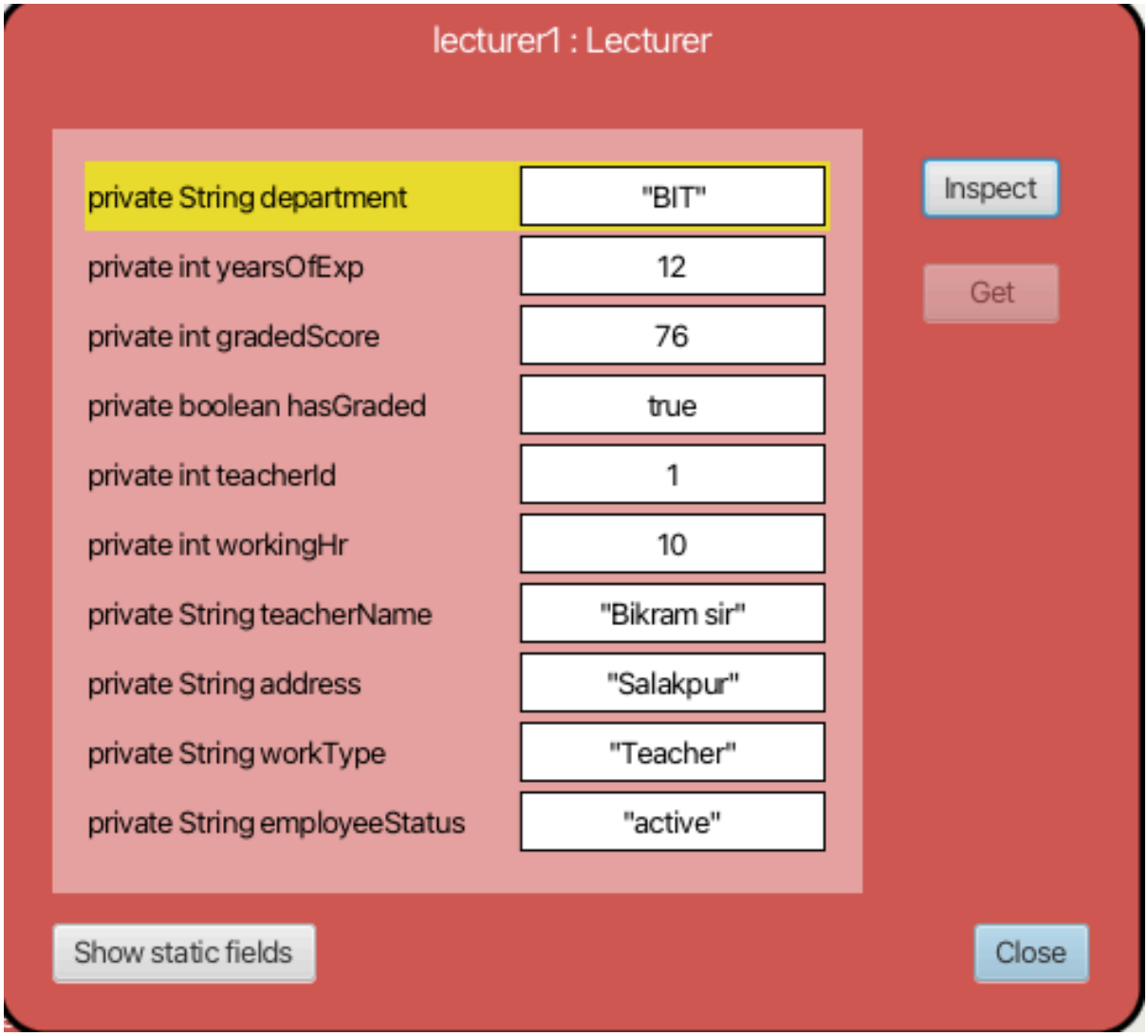

Figure 18: parameterized class lecturer

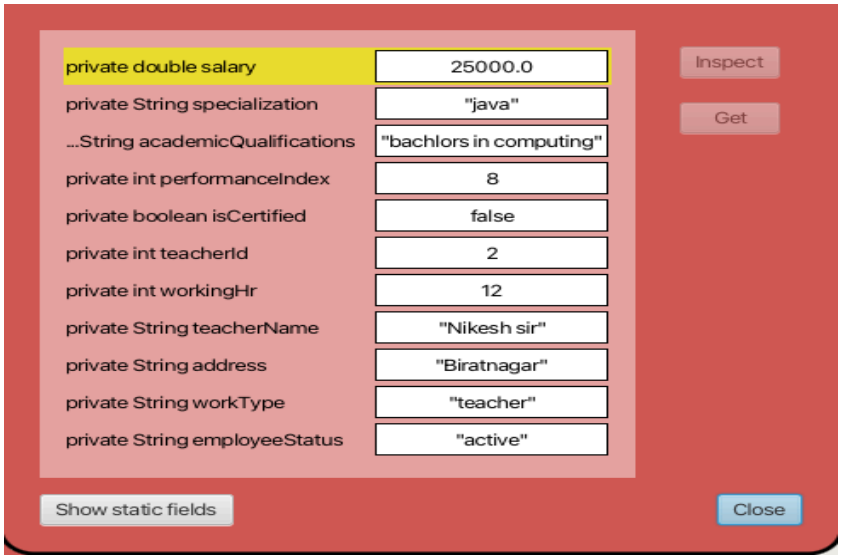

Figure 19: parameterized class tutor

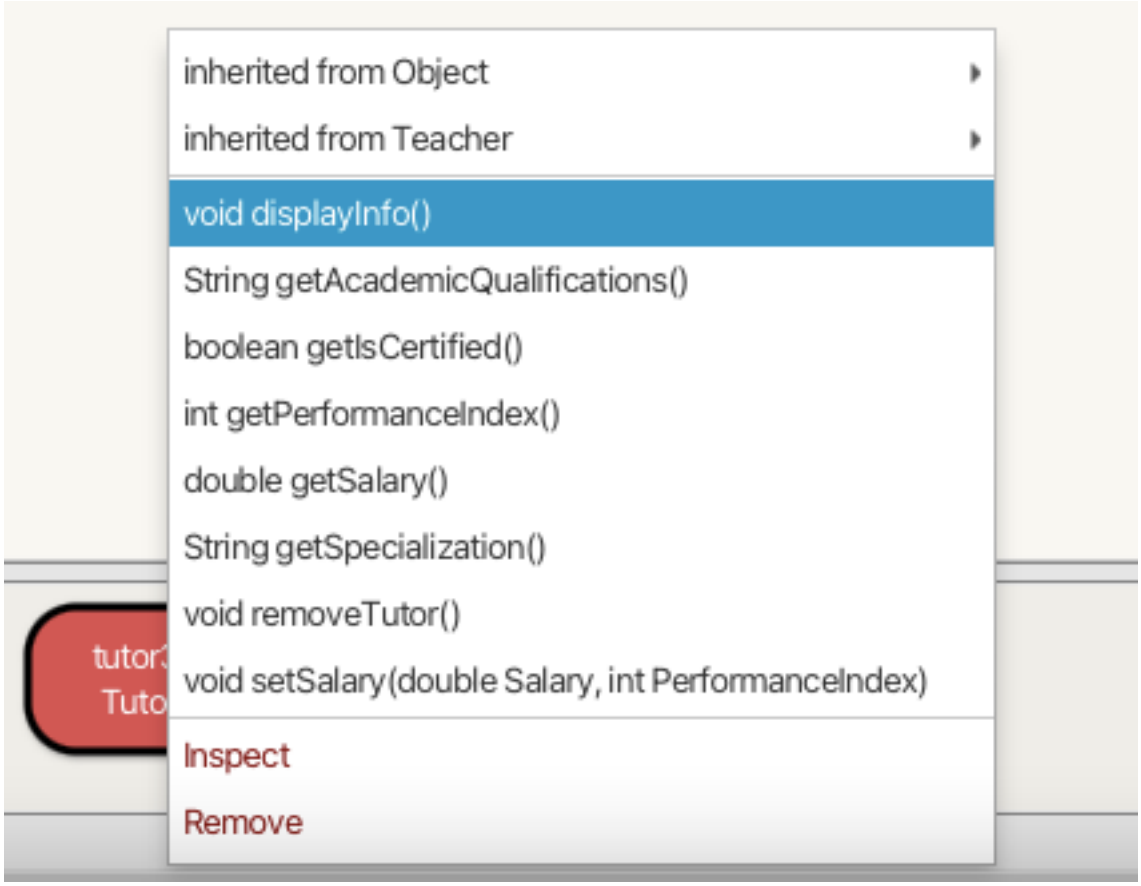

Figure 20: Display method calling for lecture class

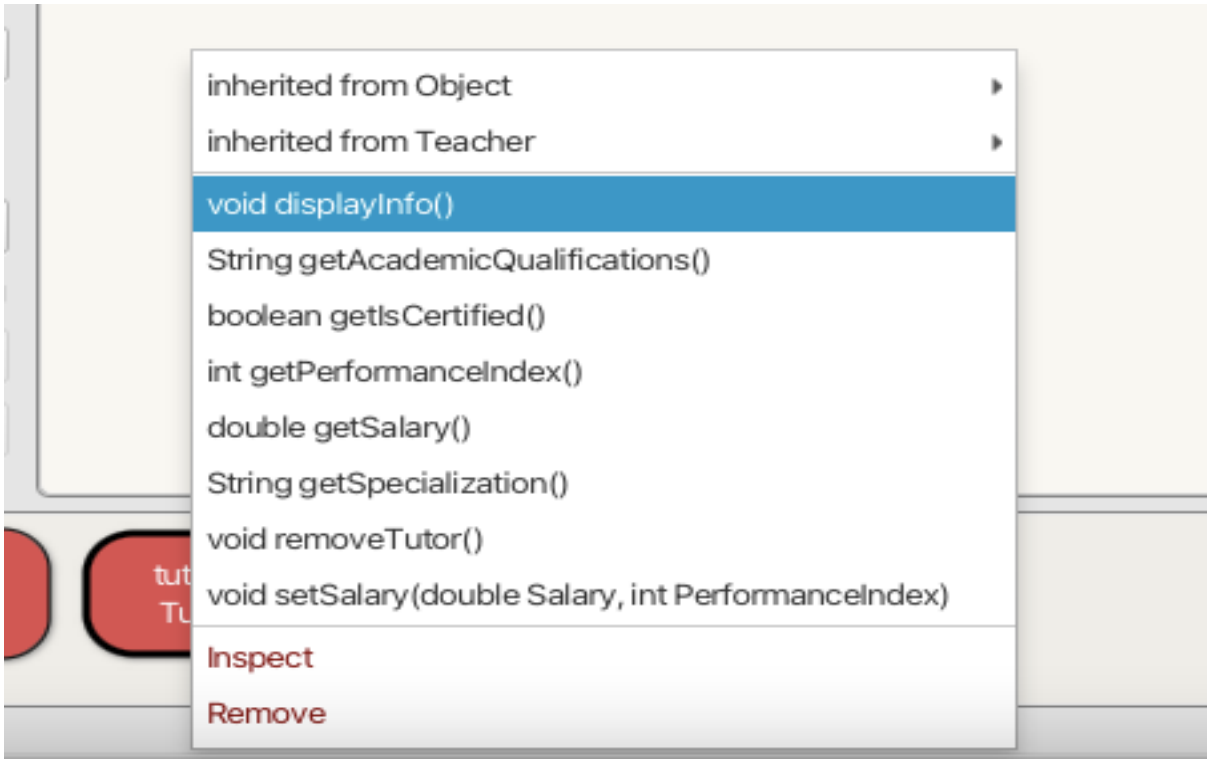

Figure 21: display method calling for tutor class

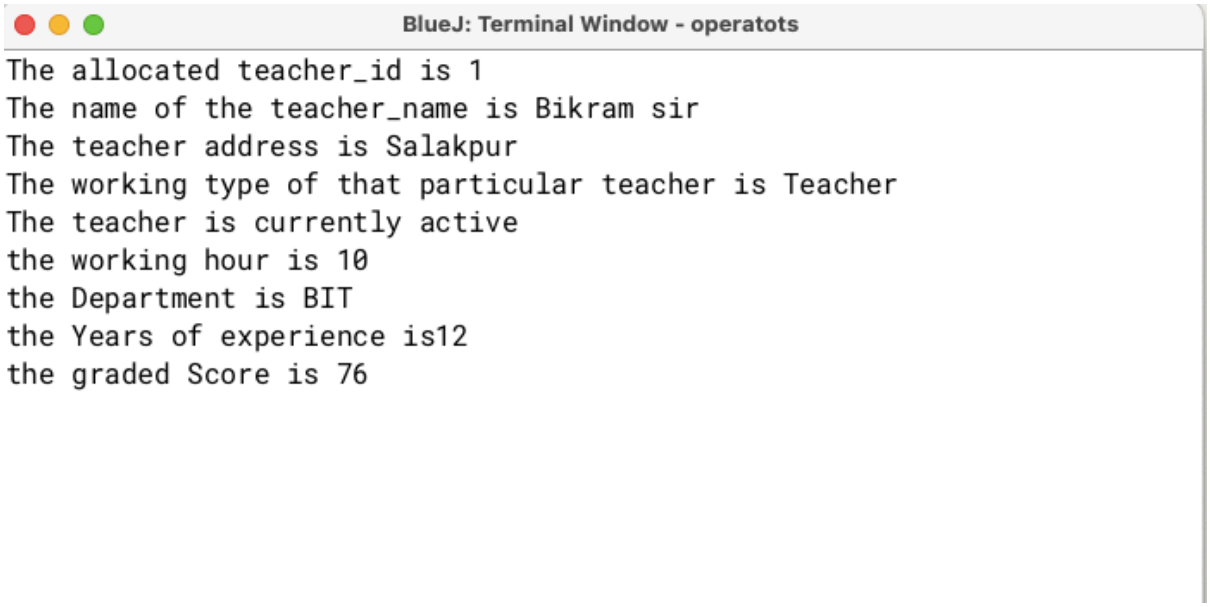

Figure 22: result of displaying details of lecturer class

```
BlueJ: Terminal Window - operatots
The allocated teacher_id is 2
The name of the teacher_name is Nikesh sir
The teacher address is Biratnagar
The working type of that particular teacher is teacher
The teacher is currently active
the working hour is 12
```

Figure 23: result of displaying details of tutor class

6 Error detection and correction

6.1 Syntax error

In the shown error below a semicolon is missing due to which an error is popping up which shows (';'expected) this means the java compiler cannot end the statement this type of error's is classified as syntax errors.

```
public void setGradedScore(int gradedScore)
{
    this.gradedScore = gradedScore
}
public void  gradeAssignment(int g                    ';' expected                    ent,int yearsOfExp)
{
    if (yearsOfExp >= 5 && department.equals(getDepartment()))
```

Figure 24: syntax error

6.2 Syntax error Correction

To correct the above syntax error, we use the line ending statement that is a semicolon.

After the semicolon is added the error is gone.

```
}
public void setGradedScore(int gradedScore)
{
    this.gradedScore = gradedScore;
}
public void  gradeAssignment(int gradedScore,String department,int yearsOfExp)
{
    if (yearsOfExp >= 5 && department.equals(getDepartment()))
```

6.3 Semantic error

In the shown error below most an error is popping up which shows (incompatible types: possible lossy conversion from int to float) this is because we have entered wrong data. Here we have declared variable as an integer which only accepts integer value but we are assigning it with float value which cannot be converted by java compiler however the float variable can hold integers value.

```
public Lecturer(int teacherId,String teacherName,String address,String workType,String employeeStatus,String department,int yearsOfExp,int wor
{
    super(teacherId,teacherName,address,workType,employeeStatus);
    super.setWorkingHr(workingHr);
    this.department =department;
    this.yearsOfExp= yearsOfExp;
    this.gradedScore= 12.5;
    this.hasGraded= fals
}
```

incompatible types: possible lossy conversion from double to int

Figure 25: semantic error

6.4 Semantic error correction

The above error was corrected after the actual integer variable was introduced to the respective variable.

```
public Lecturer(int teacherId,String teacherName,String address,String workType,String employeeStatus,String department,int yearsOfExp,int wor
{
    super(teacherId,teacherName,address,workType,employeeStatus);
    super.setWorkingHr(workingHr);
    this.department =department;
    this.yearsOfExp= yearsOfExp;
    this.gradedScore= 12;
    this.hasGraded= false;
}
```

Figure 26: correction of semantic error

6.5 Logical errors

In the error shown below there is an error int the line “if (hasGraded = true)” as the = operator assigns the value whereas == operator is a comparison operator which could be used during the comparison. This is an example of logical error although the program is programmatically right but is logically wrong.

```
if (hasGraded = true)
{
    System.out.println("the graded Score is "+gradedScore);
}
else
{
    System.out.println("The assignment is not graded yet");
}
```

Figure 27: logical error

6.6 Logical errors correction

The above error is corrected by using comparison operator for Boolean value. Alternatively, we could use directly has graded as Boolean because it itself is a Boolean but still we are comparing the values.

```
public void displayInfo()
{
    super.displayInfo();
    System.out.println("the Department is "+department);
    System.out.println("the Years of experience is"+yearsOfExp);
    if (hasGraded == true)
    {
        System.out.println("the graded Score is "+gradedScore);
    }
    else
```

Figure 28: logical error correction

7 Conclusion

For the conclusion, I have done this coursework using java as a programming language. For the work I have created 3 classes that are Teacher, lecturer and tutor here I have used the concept of inheritance where I have used teacher as a parent class where as I have used lecturer and tutor as a child class. In these classes I have declared and used various data typed variables as well as different methods. Regardless of other method I have used display method using the concept of method overriding. During the process of coursework completion, I have gained wide range of knowledge like error identification, error handling, code formatting but mainly I have learned a lot about the different types methods and its use. During the completion of my work I have had encounter with different types of errors in which forgetting the semicolon was one of the main. Despite the problems I overcame all of the problems through my research and guidance from my teachers and seniors.

8 References

AWS, 2023. *AWS.amazone.* [Online] Available at: https://aws.amazon.com/what-is/java/ [Accessed 01 2024].

Rouse, M., 2020. *Techopedia.* [Online] Available at: https://www.techopedia.com/definition/3927/java [Accessed 01 2024].

visual-paradism, 2024. *visual-paradism.com.* [Online] Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/
[Accessed January 2024].

9 Appendix

9.1 Teacher class

```
package Project_CourseWork;


/**
 * This is a parent class of the program.
 *
 * @author (Anmol Poudyal)
 * @version (2023/12/29)
 */
public class Teacher
{
   private int teacherId;
   private int workingHr;
   private String teacherName;
   private String address;
   private String workType;
   private String employeeStatus;


   public Teacher (int teacherId,String teacherName,String address,String workType,String employeeStatus)
   {
       this.teacherId = teacherId;
       this.teacherName = teacherName;
       this.address = address;
```

```
        this.workType = workType;
        this.employeeStatus = employeeStatus;
    }

    public int getTeacherId()
    {
        return teacherId;
    }

    public String getTeacherName ()
    {
        return teacherName;
    }

    public String getAddress()
    {
        return address;
    }

    public String getWorkType()
    {
        return workType;
    }
```

```
  public String getEmployeeStatus()
  {
      return employeeStatus;
  }

  public int getWorkingHr()
  {
      return workingHr;
  }
  public void setWorkingHr(int workingHr)
  {
      this.workingHr = workingHr;
  }

  public void displayInfo()
  {
      System.out.println("The allocated teacher_id is "+teacherId);
      System.out.println("The name of the teacher_name is "+teacherName);
      System.out.println("The teacher address is "+address);
      System.out.println("The working type of that particular teacher is "+ workType);
      System.out.println("The teacher is currently "+employeeStatus);
      if (workingHr == 0)
      {
          System.out.println(" oppps!! The working hour is not assigned yet ");
```

```
        }
        else
        {
            System.out.println("the working hour is "+workingHr);
        }
    }
}
```

9.2 Lecturer class

```
package Project_CourseWork;

/**
 * This is a child class of the program.
 *
 * @author (Anmol Poudyal)
 * @version (2023/12/29)
 */

public class Lecturer extends Teacher
{
    private String department;
    private int yearsOfExp;
    private int gradedScore;
    private boolean hasGraded;
```

```
 public Lecturer(int teacherId,String teacherName,String address,String workType,String employeeStatus,String department,int yearsOfExp,int workingHr)
  {
    super(teacherId,teacherName,address,workType,employeeStatus);
    super.setWorkingHr(workingHr);
    this.department =department;
    this.yearsOfExp= yearsOfExp;
    this.gradedScore= 12;
    this.hasGraded= false;
  }
  public String getDepartment()
  {
    return department;
  }
  public int yearsOfExp()
  {
    return yearsOfExp;
  }
  public boolean hasGraded()
  {
    return hasGraded;
  }
  public void setGradedScore(int gradedScore)
  {
```

```
        this.gradedScore = gradedScore;
    }
    public void  gradeAssignment(int gradedScore,String department,int yearsOfExp)
    {
        if (yearsOfExp >= 5 && department.equals(getDepartment()))
        {
            if(gradedScore>=70 || gradedScore<=100)
            {
                System.out.println("congratulation.... you have achived A");
                this.setGradedScore(gradedScore);
                this.hasGraded=true;
            }
            else if(gradedScore>=60)
            {
                System.out.println("congratulation.... you have achived B");
                this.setGradedScore(gradedScore);
                this.hasGraded=true;
            }
            else if(gradedScore>=50)
            {
                System.out.println("congratulation.... you have achived C");
                this.setGradedScore(gradedScore);
                this.hasGraded=true;
            }
```

```
        else if(gradedScore>=40)
        {
            System.out.println("congratulation.... you have achived D");
            this.setGradedScore(gradedScore);
            this.hasGraded=true;
         }
        else if(gradedScore<40)
        {
             System.out.println("congratulation.... you have achived E");
             this.setGradedScore(gradedScore);
             this.hasGraded=true;
        }
        else
        {
            System.out.println("wrong entry");
        }


    }
    else
    {
        System.out.println("Grading failed only lecturer has the power to grade assignment");
    }
  }
```

```
  public void displayInfo()
   {
       super.displayInfo();
       System.out.println("the Department is "+department);
       System.out.println("the Years of experience is"+yearsOfExp);
       if (hasGraded == true)
       {
            System.out.println("the graded Score is "+gradedScore);
       }
       else
       {
           System.out.println("The assignment is not graded yet");
       }
   }
}
```

9.3 Tutor class

```
package Project_CourseWork;


/**
 * This is a child class of the program.
 *
 * @author (Anmol Poudyal)
 * @version (2023/12/29)
 */
```

```
public class Tutor extends Teacher
{
    private double salary;
    private String specialization;
    private String academicQualifications;
    private int performanceIndex;
    private boolean isCertified;

    public Tutor(int teacherId,String teacherName,String address,String workType,String employeeStatus,int workingHr,double salary,
    String specialization,String academicQualifications,int performanceIndex)
    {
        super(teacherId,teacherName,address,workType,employeeStatus);
        super.setWorkingHr(workingHr);
        this.salary=salary;
        this.specialization=specialization;
        this.academicQualifications=academicQualifications;
        this.performanceIndex=performanceIndex;
        this.isCertified=false;

    }
    public double getSalary()
    {
        return salary;
    }
```

```
public String getSpecialization()
{
    return specialization;
}

public String getAcademicQualifications()
{
    return academicQualifications;
}

public int  getPerformanceIndex()
{
    return  performanceIndex;
}

public boolean getIsCertified()
{
    return isCertified;
}

public void setSalary(double Salary,int PerformanceIndex)
{
```

```
if ( PerformanceIndex>=5 && getWorkingHr()>20 )
{
    if (PerformanceIndex>=5 && PerformanceIndex<=7)
    {
        Salary = (Salary + ((5/100f)*Salary));
          this.salary=Salary;
          this.isCertified=true;
    }
    else if(PerformanceIndex>=8 && PerformanceIndex<=9)
    {
        Salary = (Salary + ((10/100f)*Salary));
          this.salary=Salary;
          this.isCertified=true;
    }
    else if(PerformanceIndex==10)
    {
        Salary = (Salary + ((20/100f)*Salary));
          this.salary=Salary;
          this.isCertified=true;
    }
    else
    {
        System.out.println("invalid result");
    }
```

```
        }
        else
        {
            System.out.println("your salary cannot be approved");
        }


    }
    public void removeTutor()
    {


        if (isCertified == false)
        {
            this.salary= 0;
            this.specialization="";
            this.academicQualifications="";
            this.performanceIndex=0;
            System.out.println("removed successfully......Thank u");
            this.isCertified=false;
        }
    }
    public void displayInfo()
    {
        super.displayInfo();
```

```
        if (isCertified)
        {
            System.out.println("The salary is " + salary);
            System.out.println("Specialized in " +specialization);
            System.out.println("Academic qualification " +academicQualifications);
            System.out.println("teachers performance index is " +performanceIndex);
        }
    }
}
```

10 Plagiarism report

1/24/24, 7:26 PM Anmol poudyal (23049190).docx

Originality report

COURSE NAME
Programming 2023 Autumn

STUDENT NAME
ANMOL POUDYAL

FILE NAME
Anmol poudyal (23049190).docx

REPORT CREATED
Jan 24, 2024

Summary

Flagged passages	1	1%
Cited/quoted passages	2	1%

Web matches

coursehero.com	1	1%
wikipedia.org	1	0.8%
amazon.com	1	0.5%