LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2019-20 Autumn

Student Name: Karsang Gurung

Group:n6

London Met ID:

College ID: np01nt4a190138

Assignment Due Date:

Assignment Submission Date:

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

List of figure

List of table

1. Introduction

This project was assigned to us on the 8th week for our module of java programing. This assignment solely shows the use of java programming language. It shows how the staffs are hired in an organisation be it full time staff or part time.

2. Class Diagram

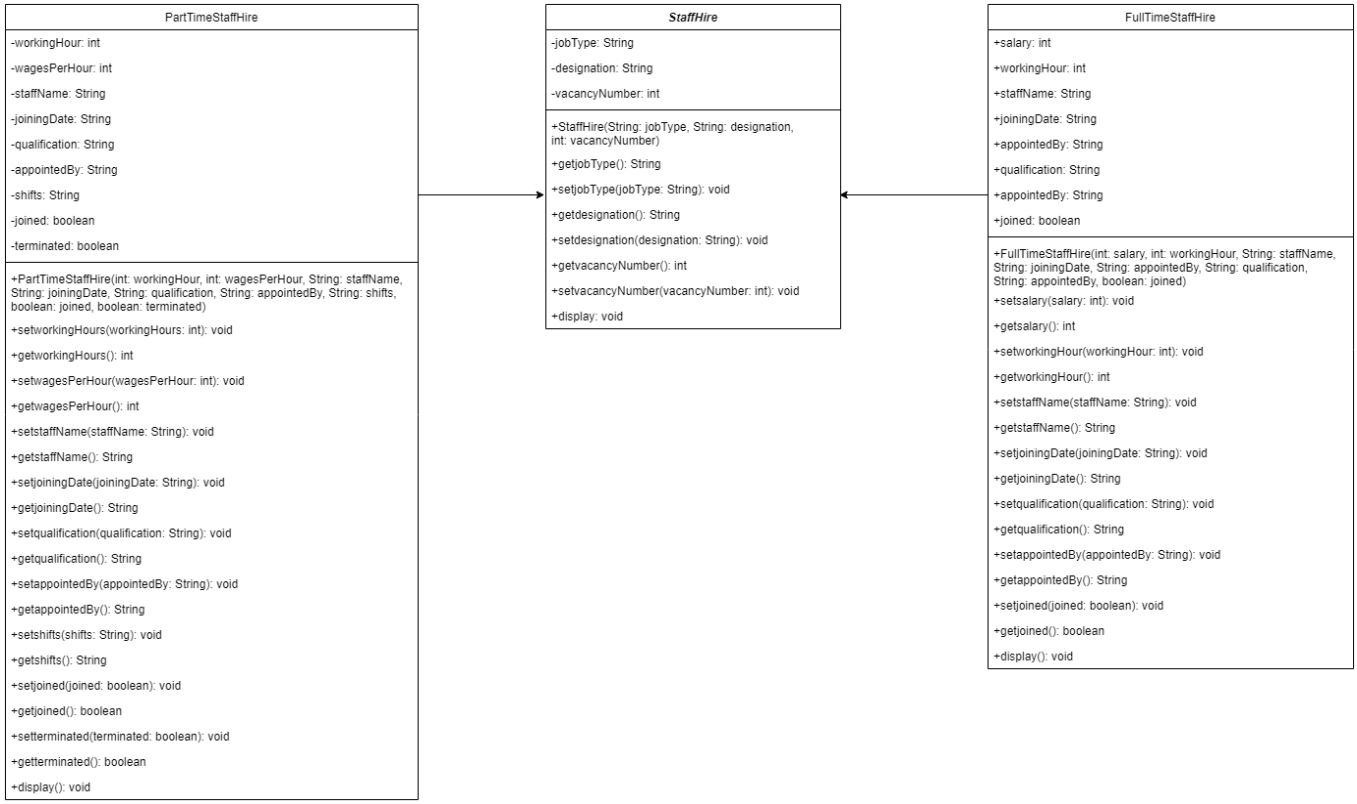


Figure 1: Class Diagram

3. Pseudocodes

3.1 Pseudocode of StaffHire (parent class)

```
CALL StaffHire(String jobType, String designation, int vacancyNumber)

DO
        This.jobType IS EQUALS TO jobType;
        This.designation IS EQUALS TO designation;
        This.vaccancyNumber IS EQUALS TO vacancyNumber;
        END DO

CALL getjobType(): String
        DO
                return jobType;
        END DO

CALL getdesignation(): String
        DO
                return designation;
        END DO

CALL  vacancyNumber()
        DO
                return vacancyNumber()
        END DO

CALL setjobType(String jobType)
        DO
                This.jobType IS EQUALS TO Type
        END DO

CALL Display()
        DO
                OUTPUT("vacancyNumber"+getvacancyNumber())
                OUTPUT("designation"+getdesignation())
                OUTPUT("jobType"+getjobType())
        END DO
```

3.2 Pseudocode of FullTimeStaffHire

```
CALL FullTimeStaffHire(int vacancyNmber, String designation, String jobType,
int Salary, int workingHour)

        DO

                SUPER(vacancyNumber, Designation, jobType)
                This.salary IS EQUALS TO salary
                This.workingHour IS EQUALS TO workingHour
                This.staffName IS EQUALS TO empty String
                This.joiningDate IS EQUALS TO empty String
                This.qualification IS EQUALS TO empty String
                This.Joined IS EQUALS TO false

        END DO

CALL getsalary(): int
        DO
                return salary;
        END DO

CALL getworkingHour(): int
        DO
                return workingHour;
        END DO

CALL getjoiningDate(): String
        DO
                return joiningDate;
        END DO

CALL getqualification(): String
        DO
                return qualification;
        END DO

CALL getappointedBy(): String
        DO
                return appointedBy;
        END DO

CALL getjoined(): boolean
        DO
                return getjoined;
        END DO
```

```
CALL setWorkingHour(int workingHour)
        DO
                This.workingHour IS EQUALS TO workingHour
        END DO

CALL CheckSalarY (int Salary)
        DO

                IF (joined) (compared true)
                DO
                        DISPLAY (“It is not possible to change the salary”);
                END DO

                ELSE

                DO
                        This.salary IS EQUALS TO salary
                END DO

        END DO

CALL  Checkworking (int workingHour)
        DO
                This.workingHour IS EQUAL TO workinghour;
        END DO

CALL FullTimeStaffHire(String staffName, String joiningDate, String
qualification, String appointedBy)

DO

        IF(Joined)

        DO

        OUTPUT(“staffName” +getStaffName())
        OUTPUT("joinedDate"+getJoiningDate())

        END DO


        ELSE

        DO
```

```
        This.staffName IS EQUALS TO staffName;
        This.joiningDate IS EQUALS TO joiningDate;
        This.qualification IS EQUALS TO qualification;
        This.appointed_By IS EQUALS TO appointedBy;
        This.joined IS EQUALS TO true;

        END DO

        CALL display()

        DO

        SUPER.display()

        IF(joined)

        OUTPUT("staffName="+getstaffName())
        OUTPUT("salary="+getSalary())
        OUTPUT("workingHour="+getworkingHour())
        OUTPUT("dateJoined="+getjoiningDate())
        OUTPUT("appointedBy="+getappointedBy())

        END DO
    END DO
```

3.3 Pseudocode of PartTimeStaffire

```
CALL PartTimeStaffHire (int vacancyNumber, String designation, String jobType,
int workingHour, int wagesPerHour, String shifts)

DO

        SUPER(vacancy_Number, Designation, job_Type)
        This.workingHour IS EQUALS TO workinghour;
        This.wages_per_hour IS EQUALS TO wagesPerHour;
        This.shifts IS EQUALS TO shifts;
        This.staff_Name IS EQUALS TO empty String;
        This.joining_Date IS EQUALS TO empty String;
        This.qualification IS EQUALS TO empty String;
        This.appointed_By IS EQUALS TO empty String;
        This.joined IS EQUALS TO false;
        This.terminated IS EQUALS TO false;

END DO

CALL getvacancyNumber(): int
        DO
                return vacancyNumber;
        END DO

CALL getworkingHour()
        DO
                return workingHour;
        END DO

CALL getWagesPerHour()
        DO
                return wagesPerHour;
        END DO

CALL staffName()
        DO
                return staffName;
        END DO

CALL String getJoining_Date()
        DO
                return joiningDate;
        END DO

CALL getqualification(): String
```

```
        DO
            return qualification;
        END DO

CALL getappointedBy()
        DO
            return appointedBy;
        END DO

CALL getShifts()
        DO
            return shifts;
        END DO

CALL getJoined(): boolean
        DO
            return joined;
        END DO

CALL getTerminated(): boolean
        DO
            return terminated;
        END DO

CALL setWorkingShifts(String shifts)
DO

        IF(joined)
        DO
        OUTPUT("staffName="+getstaffName())
        OUTPUT("joinedDate"+getjoinedDate())
        END DO

        ELSE
        This.shifts IS EQUALS TO shifts

END DO

CALL PartTimeStaffHire(String staffName, String joinedDate, String qualification
, String appointedBy)

DO
        IF(joined)

        DO
        OUTPUT("staffName="+getstaffName())
```

```
            OUTPUT("joinedDate"+getJoinedDate())
            END DO

            ELSE

            DO

            This.staffName IS EQUALS TO staffName
            This.joining_Date IS EQUALS TO joinedDate
            This.qualification IS EQUALS TO qualification
            This.appointed_By IS EQUALS TO appointedBy
            This.joined IS EQUALS TO joined
            This.terminated IS EQUALS TO false

            END DO

        END DO

        CALL TerminatedStaff(boolean Terminated )

        DO
            OUTPUT("The staff is already terminated")
        END DO
            ELSE

            DO

            This.staffName IS EQUALS TO empty String
            This.joinedDate IS EQUALS TO empty String
            This.qualification IS EQUALS TO empty String
            This.appointedBy IS EQUALS TO empty String
            This.joined IS EQUALS TO empty String
            This.terminated IS EQUALS TO true

        END DO
        END DO

        CALL display()
        SUPER display()
            IF joined EQUALS TRUE
        DO
            PRINT("staffName="+getstaffName())
            PRINT("salary="+getsalary())
            PRINT("workingHour="+getworkingHour())
            PRINT("joinedDate="+getjoinedDate())
            PRINT("qualification="+getqualification())
```

```
        PRINT("appointedBy="=getappointedBy())
    END DO
```

4. Method description

4.1 Method description for StaffHire

Method	Description
StaffHire	Initializes designation and jobType with empty String and vacancyNumber with int.
setjobType	Assigns the parameter value to jobType
getjobType	Return the value of jobType
setdesignation	Assigns the parameter value to designation
getdesignation	Returns the value of designation
setvacancyNumber	Assigns the parameter value to vacancyNumber with int datatype
getvacancyNumber	Returns the value of vacancyNumber

Table 1: Method description for StaffHire

4.2 Method description for FullTimeStaffHire

Method	Description
FullTimeStaffHire	Initializes description of parent class with parameters staffName, qualification, joinedDate and appointedBy
Setsalary	Assigns the parameter value to salary
getsalary	Returns the value of salary
setworkingHour	Assigns the parameter value workingHour
getworkingHour	Returns the value of workingHour
setstaffName	Assigns the parameter value of staffName
getstaffName	Returns the value of staffName
setjoinedDate	Assigns the parameter value of joinedDate
getjoinedDate	Returns the value of joinedDate
setqualification	Assigns the paramerter value of qualification
getqualification	Returns the value of qualification
setappointedBy	Assigns the parameter value of appoinedBy
getappointedBy	Returns the value of appointedBy
setjoined	Assigns the parameter value of joined
Getjoined	Return the value of joined in Boolean datatype

Table 2: Method description for FullTimeStaffHire

4.3 Method description for PartTimeStaffHire

Method	Description
PartTimeSaffHire	Initializes the description of the parent class via parameters designation, jobType, vacancyNumber,workingHour, wagesPerHour and shifts
setworkingHour	Assigns the parameter value to workingHour
getworkingHour	Returns the value of workingHour
setwagesPerHour	Assigns parameter value to wagesPerHour
getwagesPerHour	Returns value of wagesPerHour
setstaffName	Assigns parameter value to staffName
getstaffName	Returns value of staffName
setjoinedDate	Assigns parameter value to joinedDate
getjoinedDate	Returns value of joinedDate
setqualification	Assigns parameter value to qualification
getquaification	Returns value of qualification
setappointedBy	Assigns parameter value to appointedBy
getappointedBy	Returns value of appointedBy
setshifts	Assigns parameter value to shifts
getshifts	Returns value of shifts
setjoined	Assigns parameter value to joined
getjoined	Returns value of joined in Boolean datatype
setterminated	Assigns parameter value to terminated
getterminated	Returns value of terminated in Boolean datatype

Table 3: Method description for PartTimeStaffHire

5. Testing

5.1 Test 1

Objective	To inspect FullTimeStaffHire, hire full time staff and re-inspect the FullTimeStaffHire class
Action	The FullTimeStaffHire class is called with following arguments: vacancyNumber: 13 designation: “Managing Director” jobType: “Full time” salary: 75000 workingHour: 9 Inspection of FullTimeStaffHire class. Void FullTimeStaffHire is called with the following arguments: staffName: “Karsang Gurung” joiningDate: “22 Feb, 2019” qualification: “MBA” appointedBy: “CEO” re-inspection of the FullStaffHire class.
Expected Result	Full time staff should be displayed after getting appointed
Actual Result	The staff was hired
Conclusion	The test is successful

Table 4: Test 1

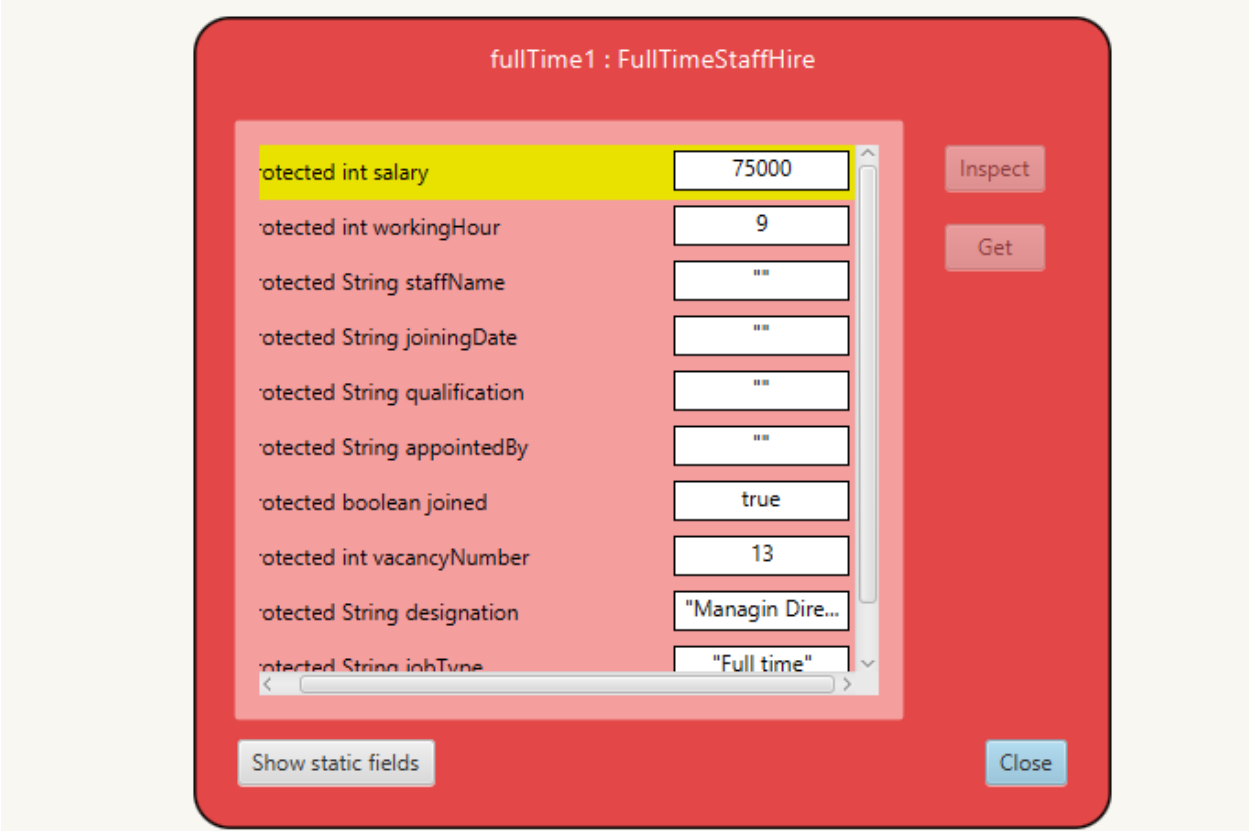


Figure 2: Inspection of FullTimeStaffHire

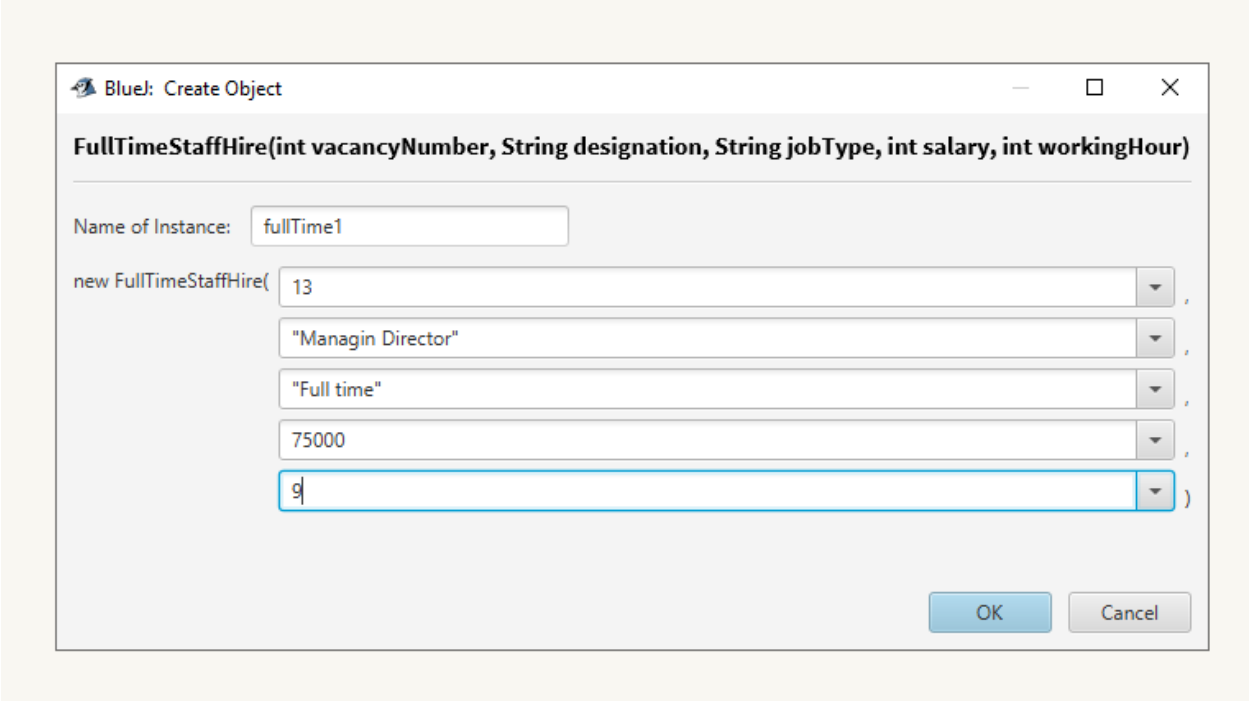


Figure 3: hiring full time staff

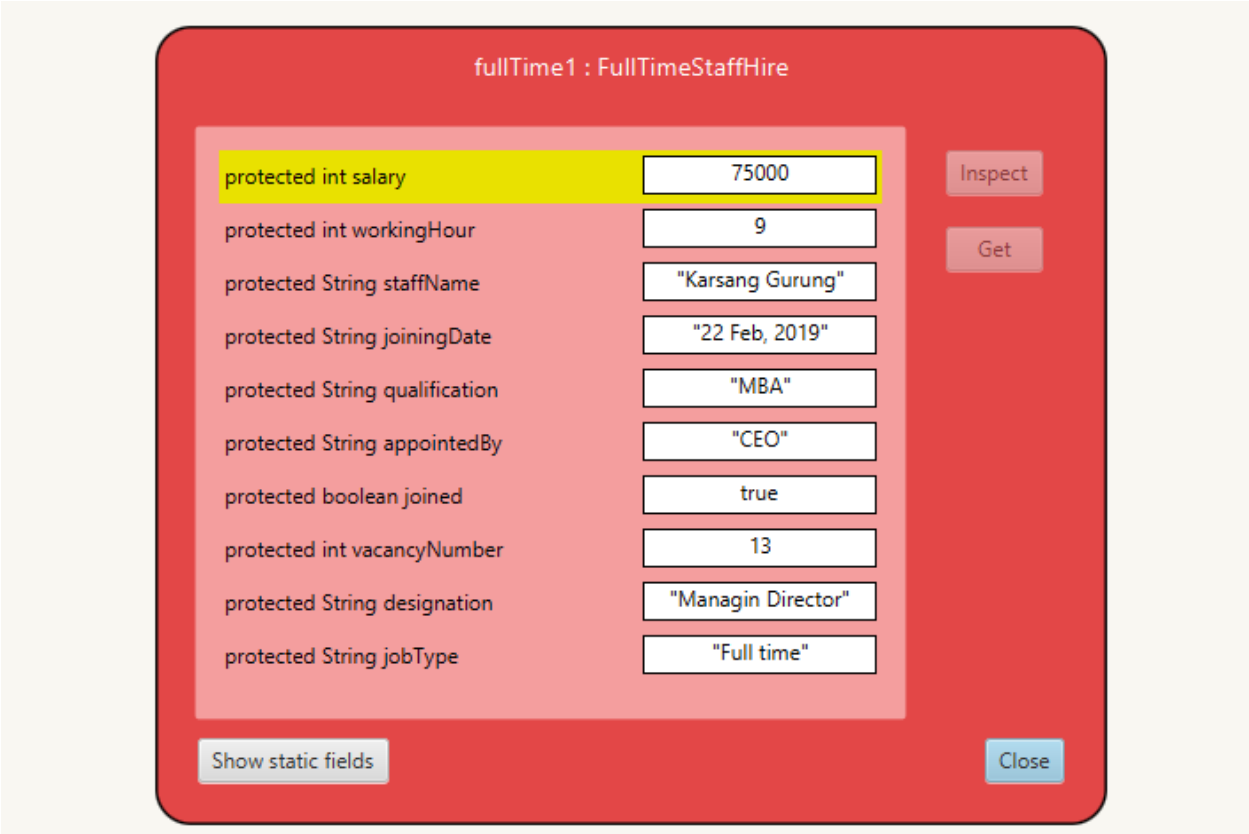


Figure 4: re-inspection

5.2 Test 2

Objective	To inspect PartTimeStaffHire, hire part time staff and re-inspect PartTimeStaffHire
Action	The PartTimeStaffHire is called with the following arrangements: Designation:"CEO" jobType: "Part time" vacancyNumber: 13 workingHour: 9 wagesPerHour: 100 shifts: "Day" Inspection of PartTimeStaffHire class Void PartTimeStaffHire is called with the following arguments: staffName: "Karsang Gurung" joiningDate: "22 Feb, 2019"

	qualification: “MBA” appointedBy: “Founder” re-inspection of PartTimeStaffHire class
Expected result	Part Time staff should be displayed after getting appointed
Actual result	The staff was hired

Table 5: test 2

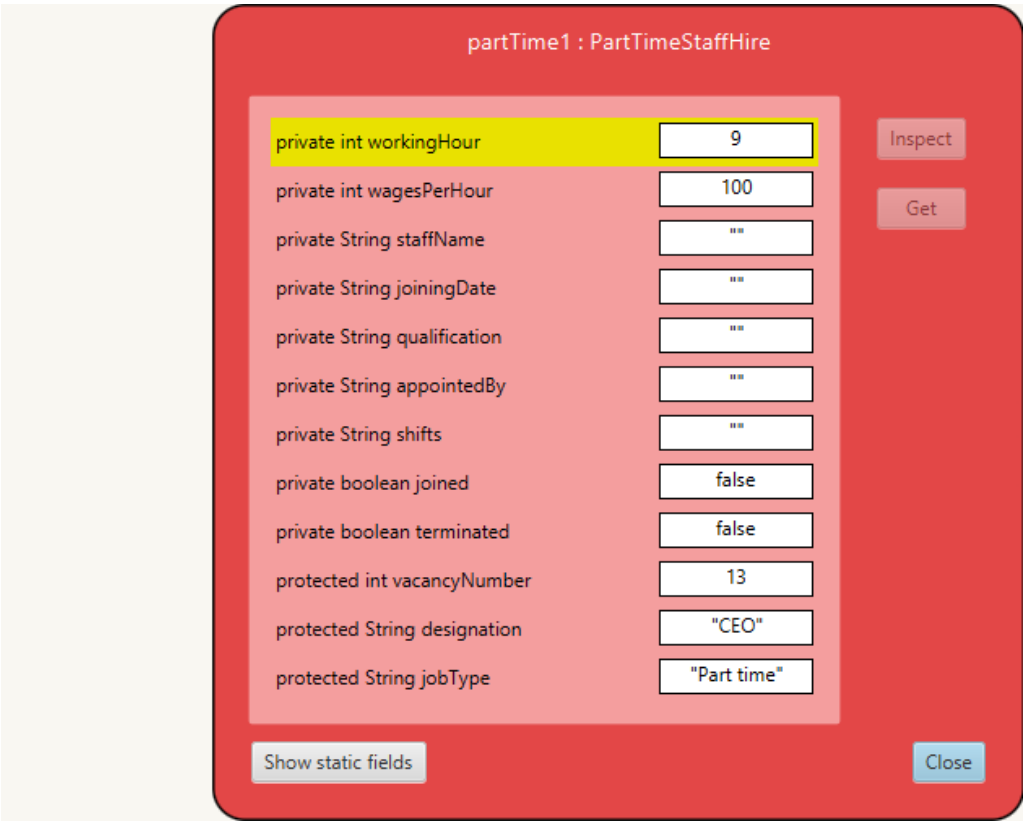


Figure 5: inspection of PartTimeStaffHire

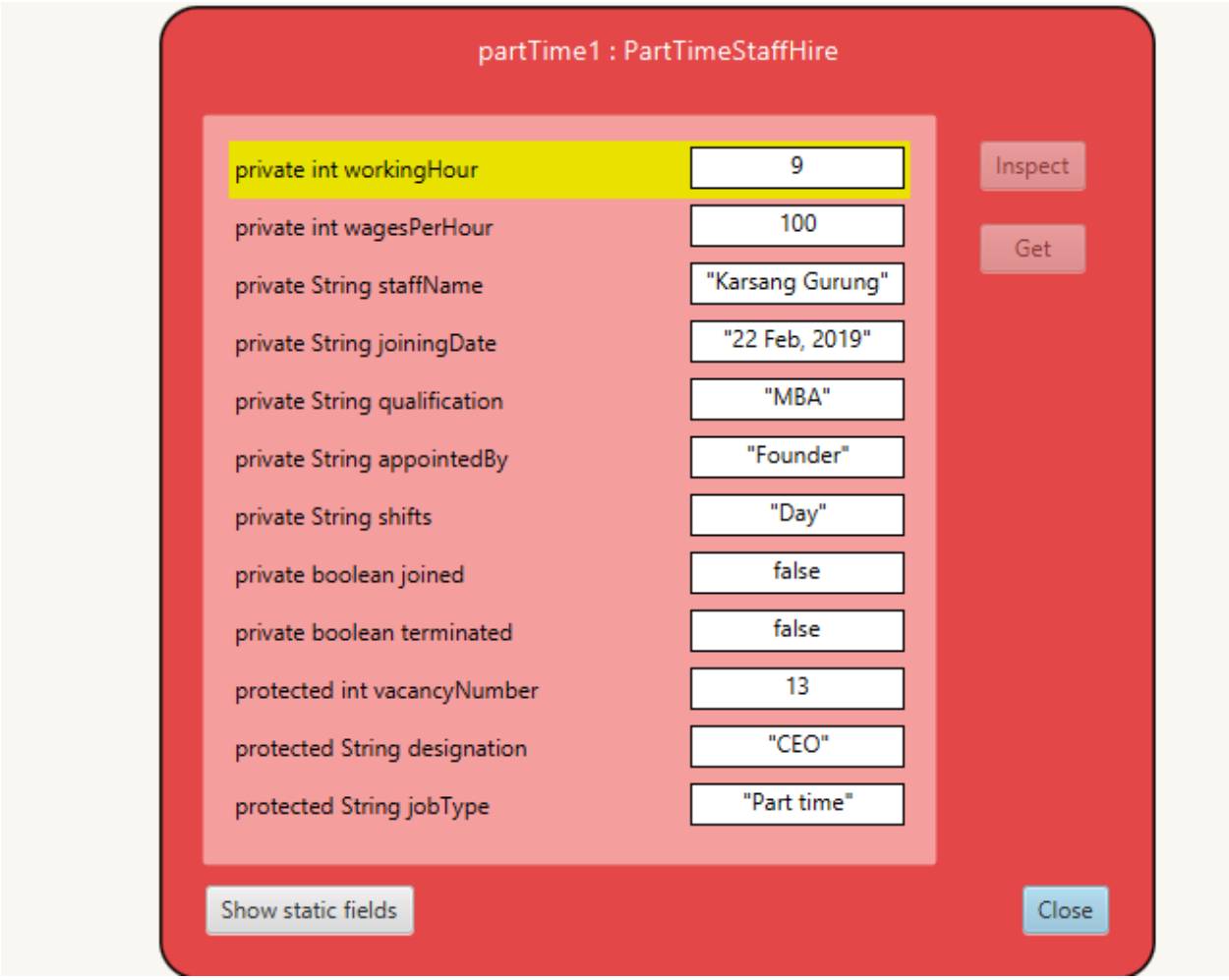
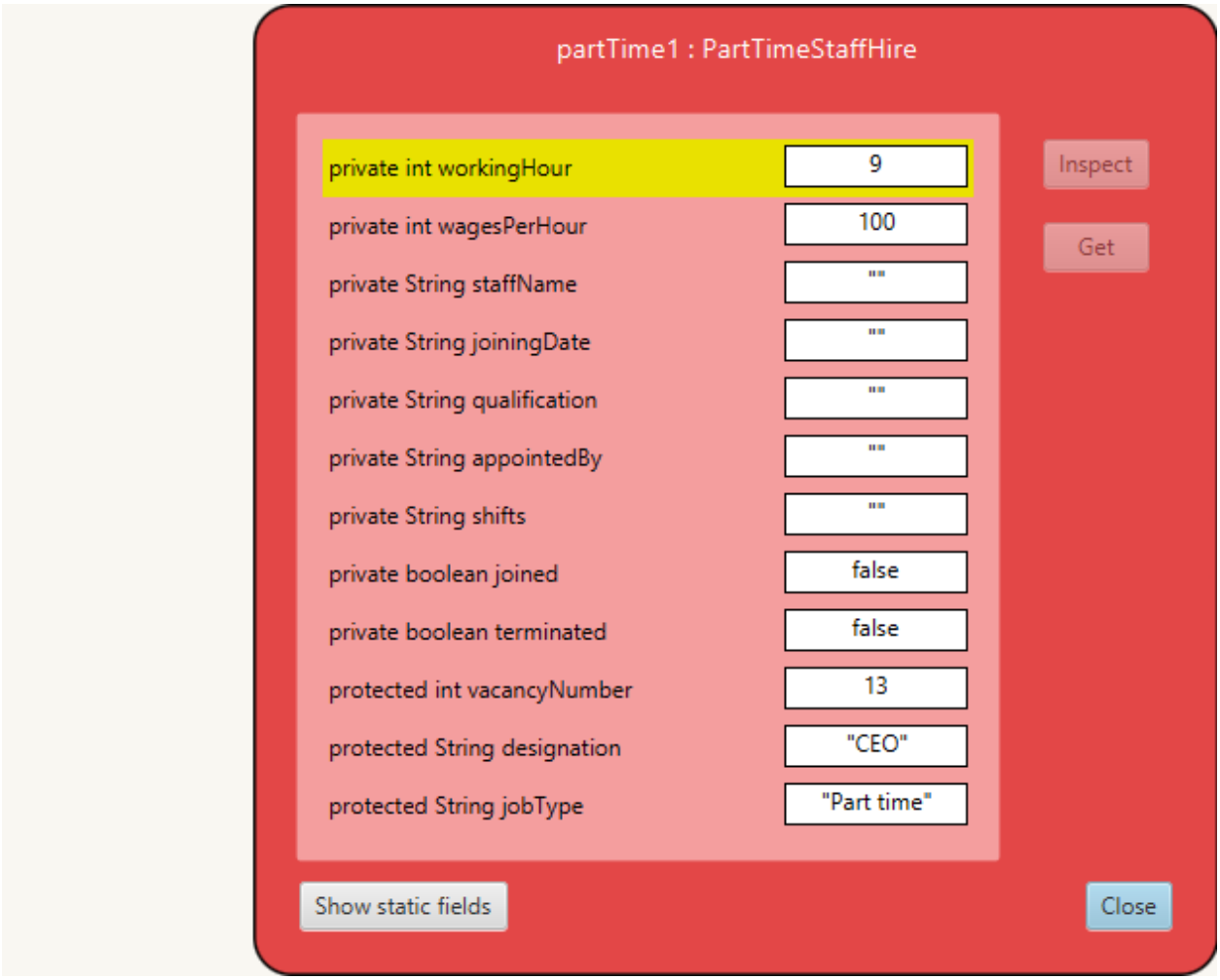


Figure 6: re-inspection

5.3 Test 3

Objective	To terminated Part Time Staff
Action	Inspection of PartTimeStaffHire class, Termination status of the staff and Re-inspection of PartTimeStaffHire class
Expected result	The status should be terminated
Actual result	The status is terminated.

Table 6: test 3



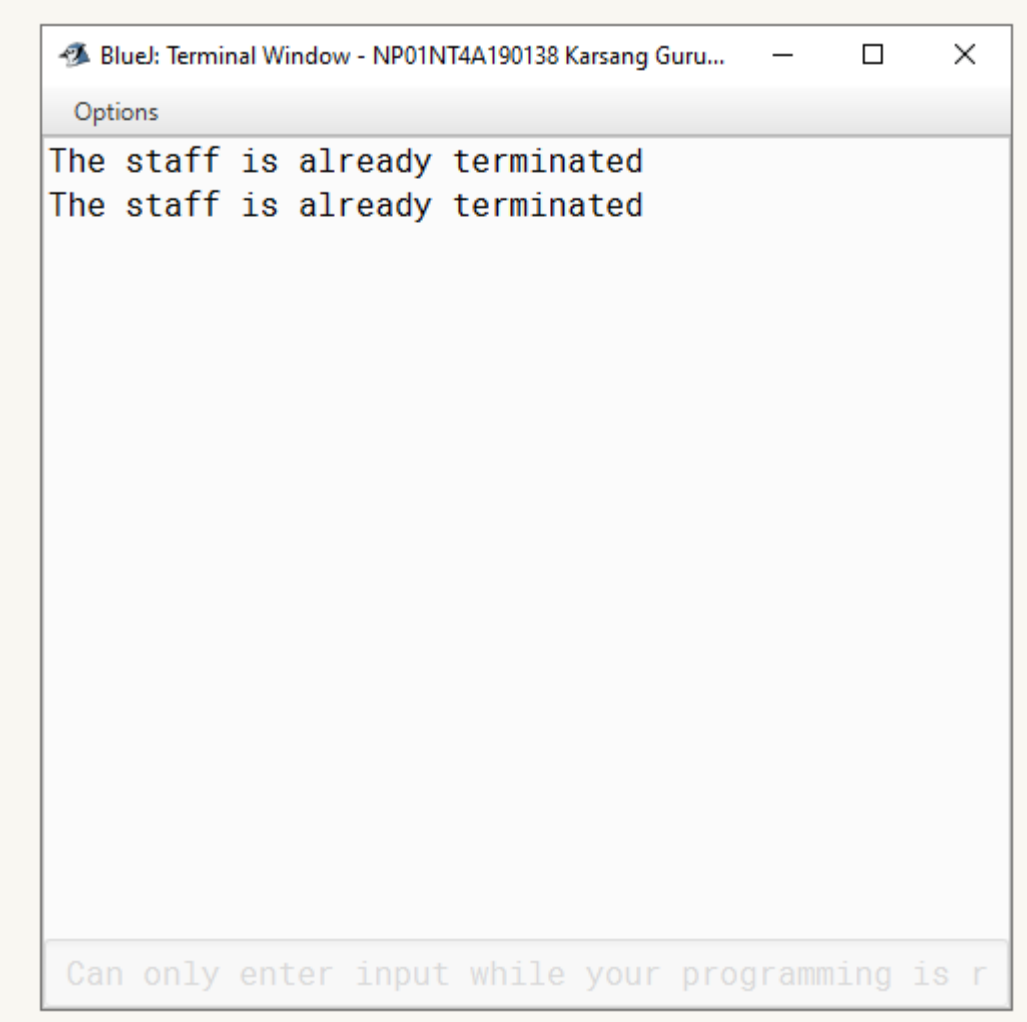


Figure 7: termination of fullstaff

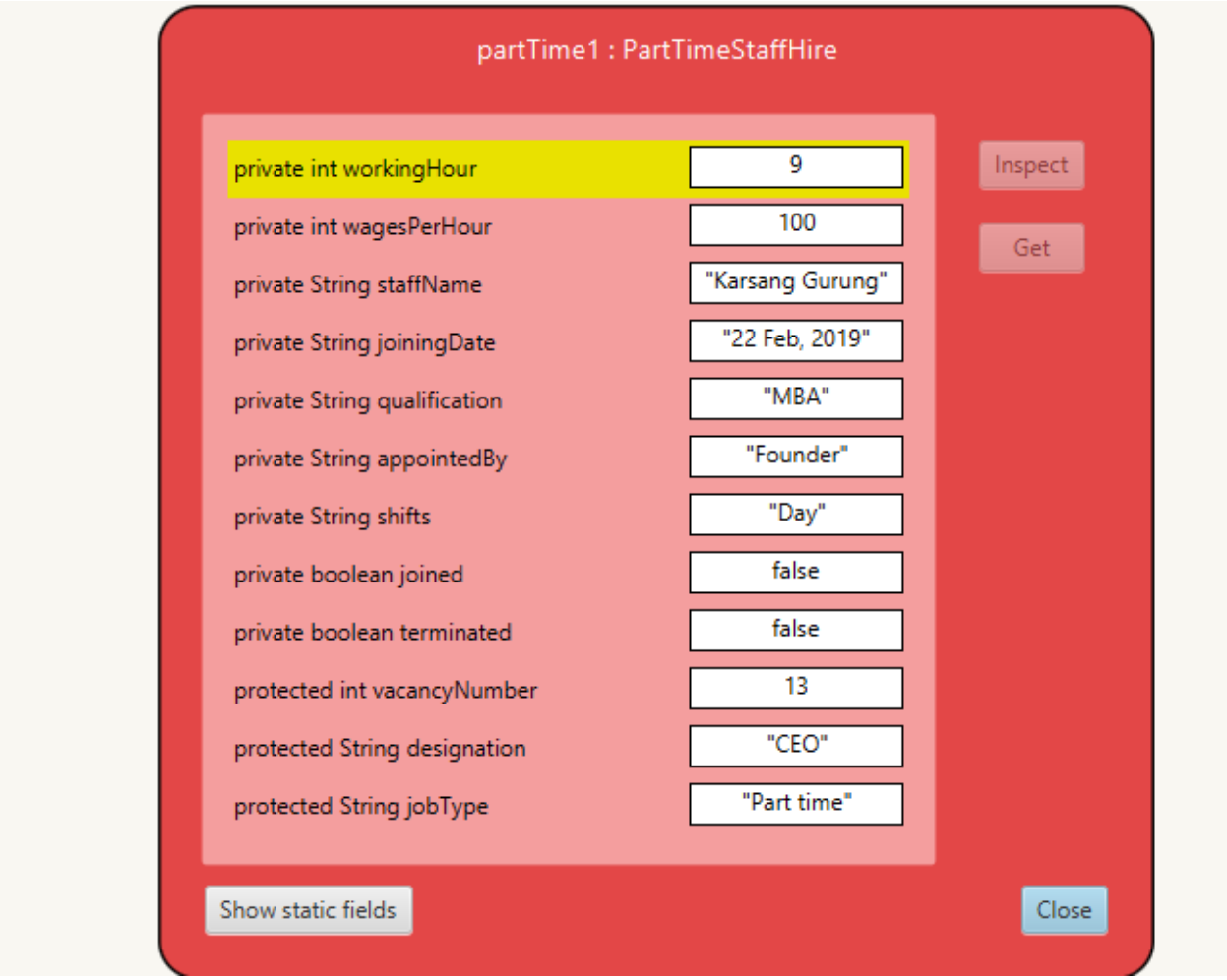
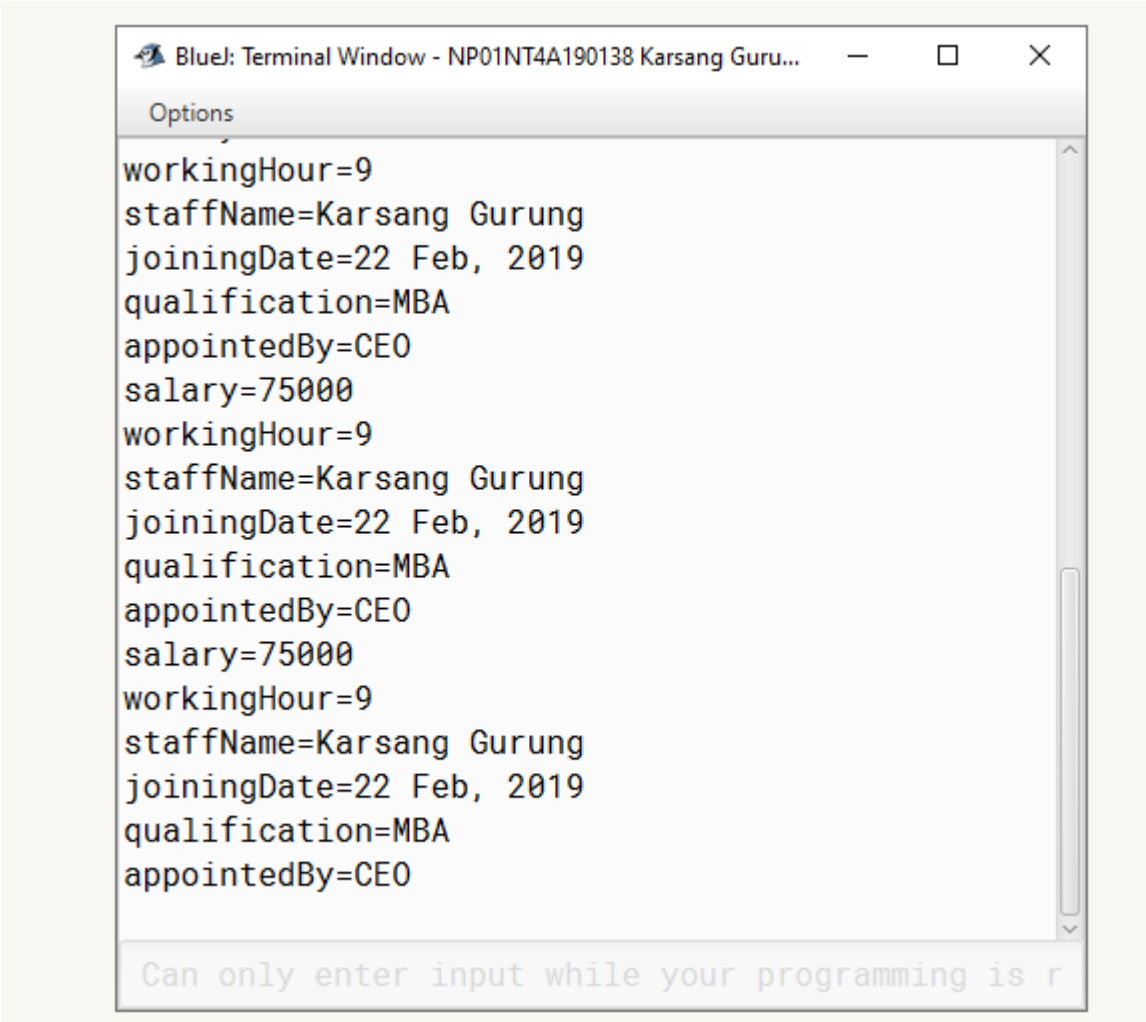
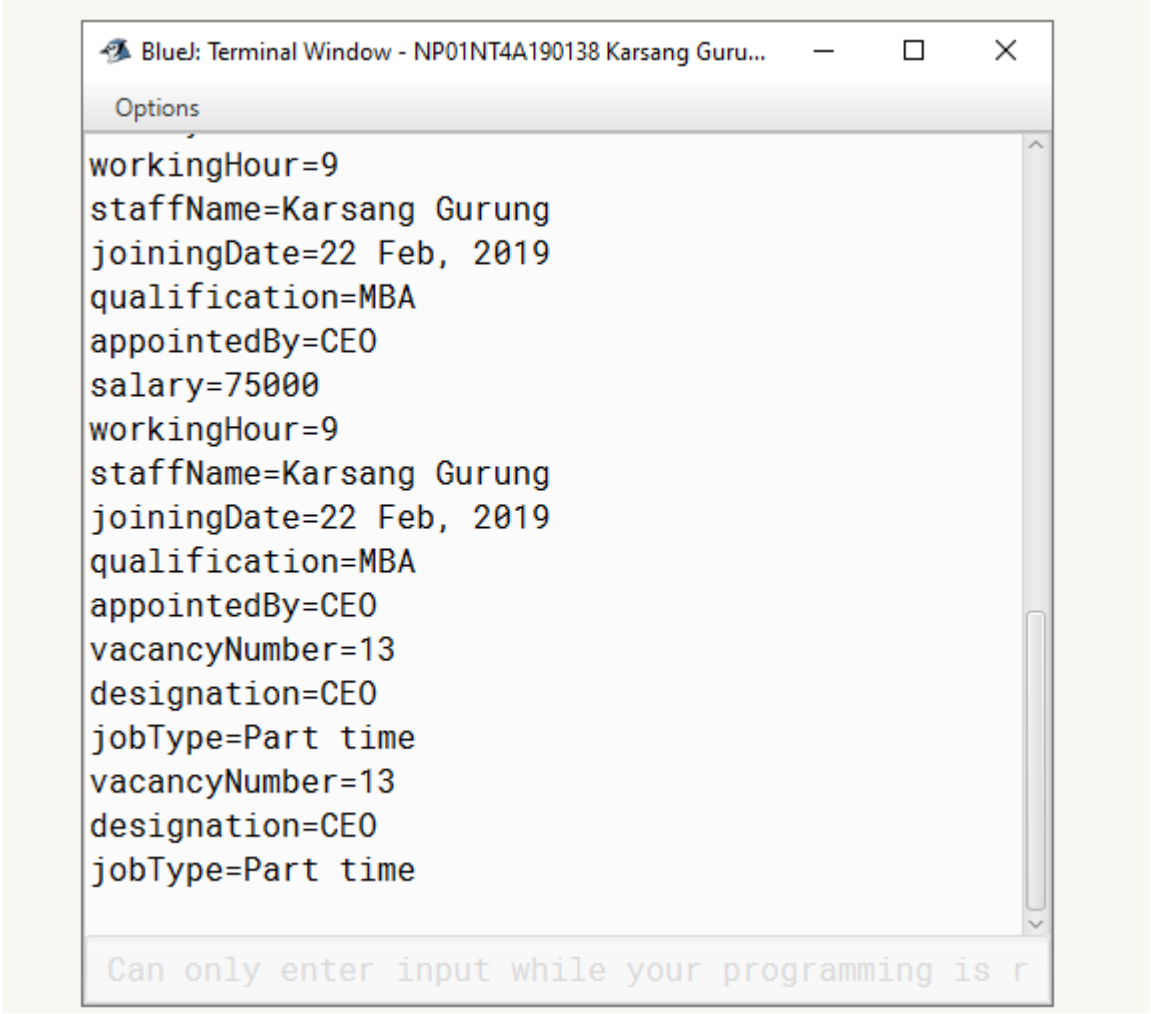


Figure 8: termination of parttimestaff

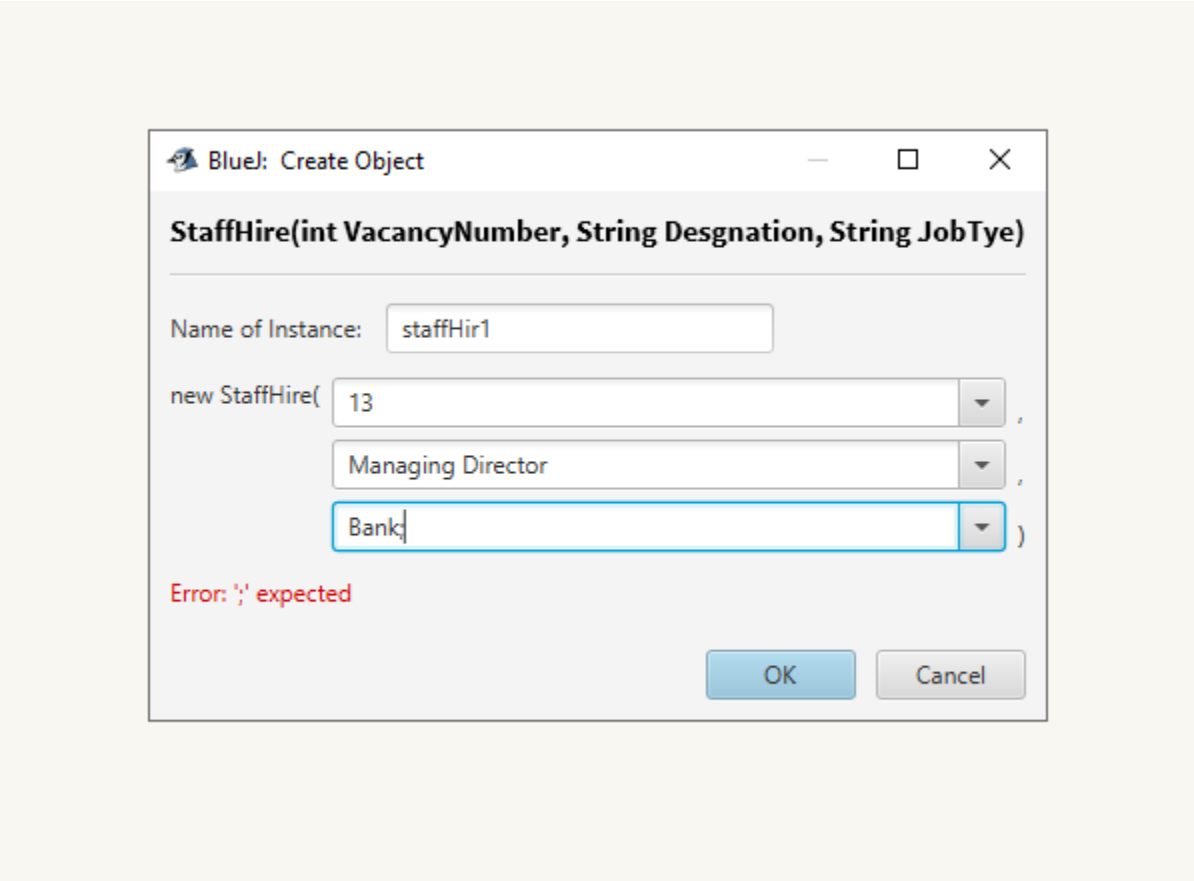
5.4 Test 4

Objective	To display the final result of FullTimeStaffHire and PartTimeStaffHire class after inspection
Action	Display the attributes of both FullTimeStaffHire class and PartTimeStaffHire class
Expected result	All the attributes should be displayed
Actual result	All the attributes were displayed

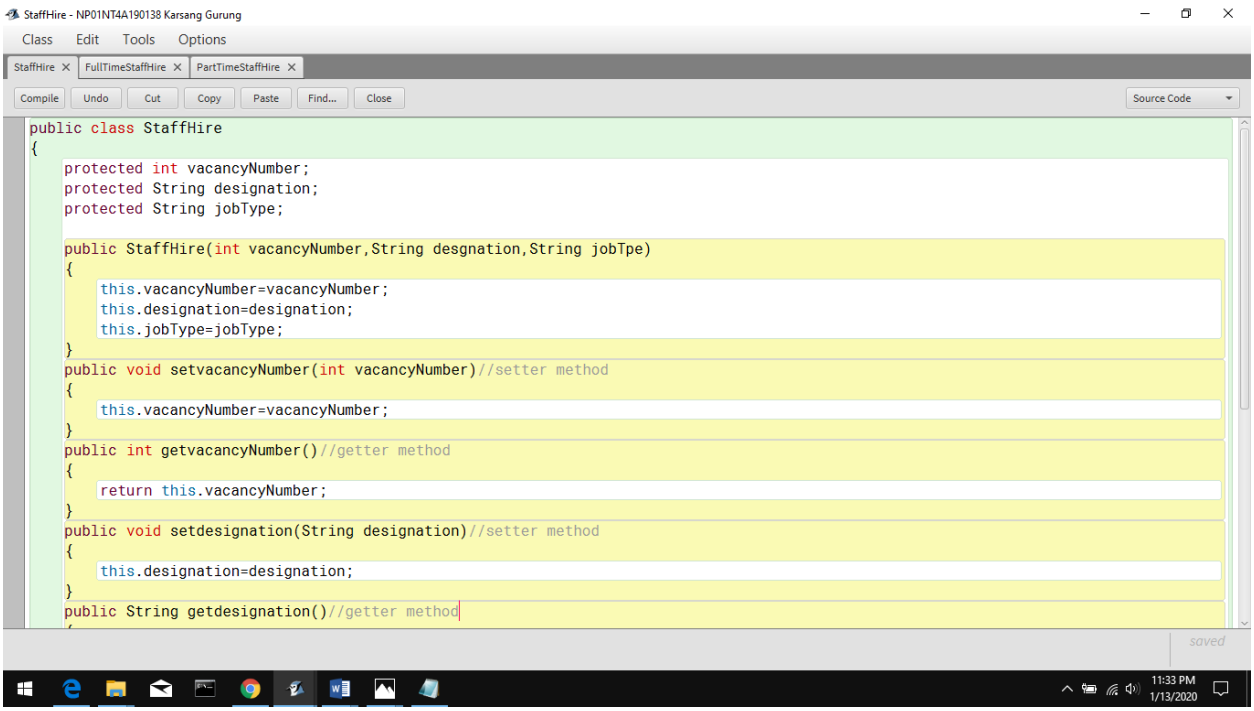



BlueJ: Terminal Window - NP01NT4A190138 Karsang Guru...
Options
workingHour=9
staffName=Karsang Gurung
joiningDate=22 Feb, 2019
qualification=MBA
appointedBy=CEO
salary=75000
workingHour=9
staffName=Karsang Gurung
joiningDate=22 Feb, 2019
qualification=MBA
appointedBy=CEO
vacancyNumber=13
designation=CEO
jobType=Part time
vacancyNumber=13
designation=CEO
jobType=Part time
Can only enter input while your programming is r

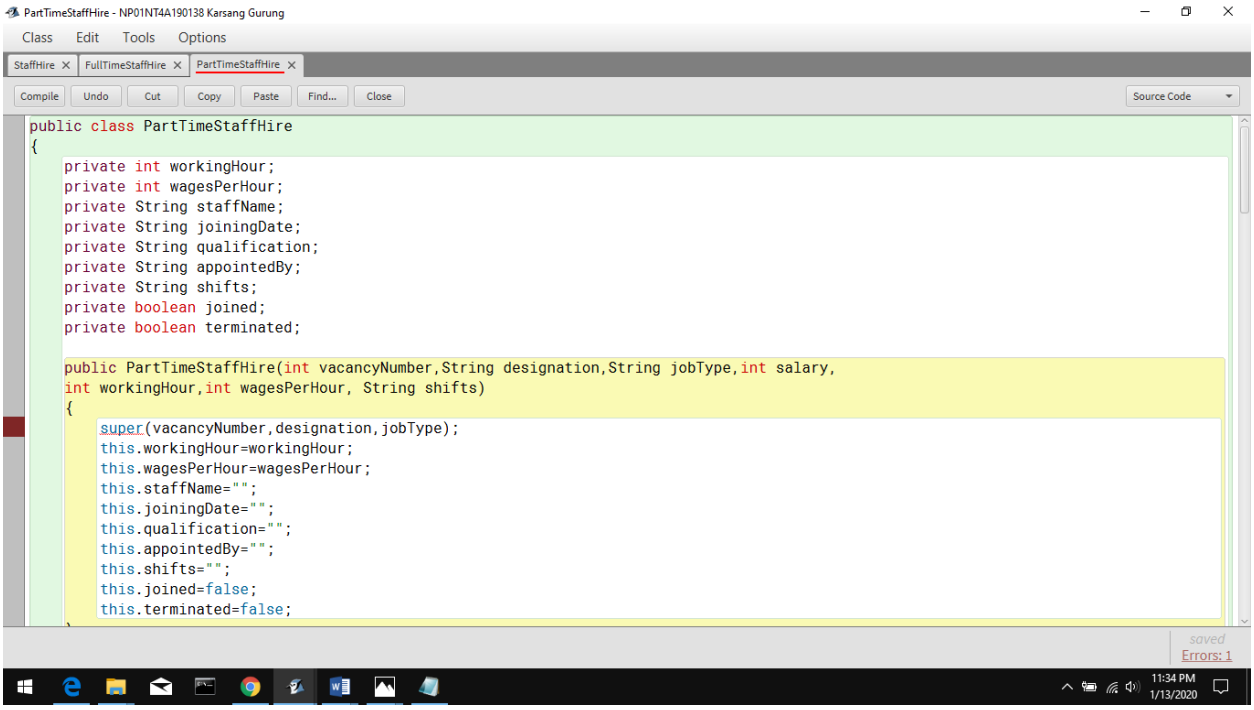
6. Error Handling



Error: passing value without double quotation
Solution: String was written in double quotation



Error: spelling of designation and jobType was mistake so the values was shown null while inspection
Solution: spelling was corrected



Error: ‘extends’ not written while creating subclass
Solution: ‘extends’ written

7. Conclusion

This assignment had been very challenging one but helpful too. It has been useful for us to learn more about methods, class, super class, pseudocode and many terms related to java programming language and also it has helped to broaden our knowledge in java. This has also helped us to hon our research skills.

8. Appendix

```
public class StaffHire
{
   protected int vacancyNumber;
   protected String designation;
   protected String jobType;

   public StaffHire(int vacancyNumber,String designation,String jobType)
   {
      this.vacancyNumber=vacancyNumber;
      this.designation=designation;
      this.jobType=jobType;
   }
   public void setvacancyNumber(int vacancyNumber)//setter method
   {
      this.vacancyNumber=vacancyNumber;
   }
   public int getvacancyNumber()//getter method
   {
      return this.vacancyNumber;
   }
   public void setdesignation(String designation)//setter method
   {
      this.designation=designation;
   }
   public String getdesignation()//getter method
   {
      return this.designation;
   }
   public void setjobType(String jobType)//setter method
   {
      this.jobType=jobType;
   }
   public String getjobType()//getter method
   {
      return this.jobType;
   }
   public void display()
   {
      System.out.println("vacancyNumber="+getvacancyNumber());
      System.out.println("designation="+getdesignation());
      System.out.println("jobType="+getjobType());
   }
}
```

```
public class FullTimeStaffHire extends StaffHire
{
   protected int salary;
   protected int workingHour;
   protected String staffName;
   protected String joiningDate;
   protected String qualification;
   protected String appointedBy;
   protected boolean joined;

   public FullTimeStaffHire(int vacancyNumber, String designation, String jobType, int
salary, int workingHour)
   {
      super(vacancyNumber,designation,jobType);//calling superclass conductor
      this.salary=salary;
      this.workingHour=workingHour;//giving value to the empty string
      this.staffName="";
      this.joiningDate="";
      this.qualification="";
      this.appointedBy="";
      this.joined=true;
   }

   public void setSalary(int Salary)//setter method
   {
      this.salary=salary;
   }
   public int getSalary()//getter method
   {
      return this.salary;
   }
   public void setworkingHour(int workingHour)//setter method
   {
      this.workingHour=workingHour;
   }
   public int getworkingHour()//getter method
   {
      return this.workingHour;
   }
   public void setstaffName(String staffName)//setter method
   {
      this.staffName=staffName;
   }
   public String getstaffName()//getter method
   {
      return this.staffName;
```

```
    }
    public void setjoiningDate(String joiningDate)//setter method
    {
        this.joiningDate=joiningDate;
    }
    public String getjoiningDate()//getter method
    {
        return this.joiningDate;
    }
    public void setqualfication(String qualification)//setter method
    {
        this.qualification=qualification;
    }
    public String getqualification()//getter method
    {
        return this.qualification;
    }
    public void setappointedBy(String appointedBy)//setter method
    {
        this.appointedBy=appointedBy;
    }
    public String getappointedBy()//getter method
    {
        return this.appointedBy;
    }
    public void setjoined(boolean joined)//setter method
    {
        this.joined=joined;
    }
    public boolean getjoined()//getter
    {
        return this.joined;
    }
    public void checkjoined(int salary)
    {
        if(joined)
        {
            System.out.println("It is not possible to change the salary");
        }
        else
        {
            this.salary=salary;
        }
    }
    public void workingHour(int workingHour)
    {
```

```
        this.workingHour=workingHour;
    }
    public void display()
    {
    System.out.println("salary="+salary);
    System.out.println("workingHour="+workingHour);
    System.out.println("staffName="+staffName);
    System.out.println("joiningDate="+joiningDate);
    System.out.println("qualification="+qualification);
    System.out.println("appointedBy="+appointedBy);
    }
}
```

```
public class PartTimeStaffHire extends StaffHire
{
    private int workingHour;
    private int wagesPerHour;
    private String staffName;
    private String joiningDate;
    private String qualification;
    private String appointedBy;
    private String shifts;
    private boolean joined;
    private boolean terminated;

    public PartTimeStaffHire(int vacancyNumber,String designation,String jobType,int
salary,
    int workingHour,int wagesPerHour, String shifts)
    {
        super(vacancyNumber,designation,jobType);
        this.workingHour=workingHour;
        this.wagesPerHour=wagesPerHour;
        this.staffName="";
        this.joiningDate="";
        this.qualification="";
        this.appointedBy="";
        this.shifts="";
        this.joined=false;
        this.terminated=false;
    }
    public void setworkingHour(int workingHour)//setter method
    {
        this.workingHour=workingHour;
    }
    public int getworkingHour()//getter Method
    {
        return this.workingHour=workingHour;
    }
    public void setwagesPerHour(int wagesPerHour)//setter method
    {
        this.wagesPerHour=wagesPerHour;
    }
    public int getwagesPerHour()//getter method
    {
        return this.wagesPerHour=wagesPerHour;
    }
    public void setstaffName(String staffName)//setter method
    {
        this.staffName=staffName;
```

```
    }
    public String getstaffName()//getter method
    {
        return this.staffName=staffName;
    }
    public void setjoiningDate(String joiningDate)//setter method
    {
        this.joiningDate=joiningDate;
    }
    public String getjoiningDate()//getter method
    {
        return this.joiningDate=joiningDate;
    }
    public void setqualification(String qualification)//setter method
    {
        this.qualification=qualification;
    }
    public String getqualification()//getter method
    {
        return this.qualification=qualification;
    }
    public void setappointedBy(String appointedBy)//setter method
    {
        this.appointedBy=appointedBy;
    }
    public String getappointedBy()//getter method
    {
        return this.appointedBy;
    }
    public void setshifts(String shifts)//setter method
    {
        this.shifts=shifts;
    }
    public String getshifts()//getter method
    {
        return this.shifts=shifts;
    }
    public void setjoined(boolean joined)//setter method
    {
        this.joined=joined;
    }
    public boolean getjoined()//getter method
    {
        return this.joined;
    }
    public void setterminated(boolean terminated)//setter method
```

```
    {
        this.terminated=terminated;
    }
    public boolean getterminated()//getter method
    {
        return this.terminated;
    }
    public void checkworkingShifts(String shifts)
    {
        if (joined)
        {
            //nothing to be printed
        }
    else
        this.shifts=shifts;
    }
    public void PartTimeStaffHire(String staffName, String joiningDate,String
qualification,String appointedBy)
    {
        if(joined)
        {
            System.out.println("The staff has already joined.");
            System.out.println("The staff name is"+staffName);
            System.out.println("The joining date is"+joiningDate);
        }
        else
            this.staffName=staffName;
            this.joiningDate=joiningDate;
            this.qualification=qualification;
            this.appointedBy=appointedBy;
            this.joined=true;
            this.terminated=false;
        }
    public void termination()
    {
        if(terminated)
        {
            System.out.println("The staff is already terminated");
        }
        else
        this.staffName="";
        this.joiningDate="";
        this.qualification="";
        this.appointedBy="";
        this.joined=false;
        this.terminated=true;
```

```
    }
}
```