CS4001NI Programming

30% Individual Coursework

2023-24 Spring

Student Name: Aasutosh Kumar Verma

London Met ID: 22085760

College ID: NP01AI4S230020

Group: AI3

Assignment Due Date: Friday, May 12, 2023

Assignment Submission Date: Friday, May 12, 2023

Table of Contents.

List of Figures

List of Tables

1 INTRODUCTION

This project, which focuses on the Java programming language, is the first one for the Programming module. Java is a popular language with a variety of uses, including business and web-based applications as well as desktop and mobile apps.

The proposal is based on a hypothetical situation where Bramos College in Kathmandu City seeks to enhance its system for maintaining student records. The institution now uses an ineffective manual method to keep track of student regular and dropout data. The college thus wants to create software to enhance these processes.

Three classes for students will be created to begin the project: Regular, Dropout, and Student. Common characteristics shared by the parent class, Student, include the enrollment details, date of birth, and course information along with fees. Regular will be the subclass.

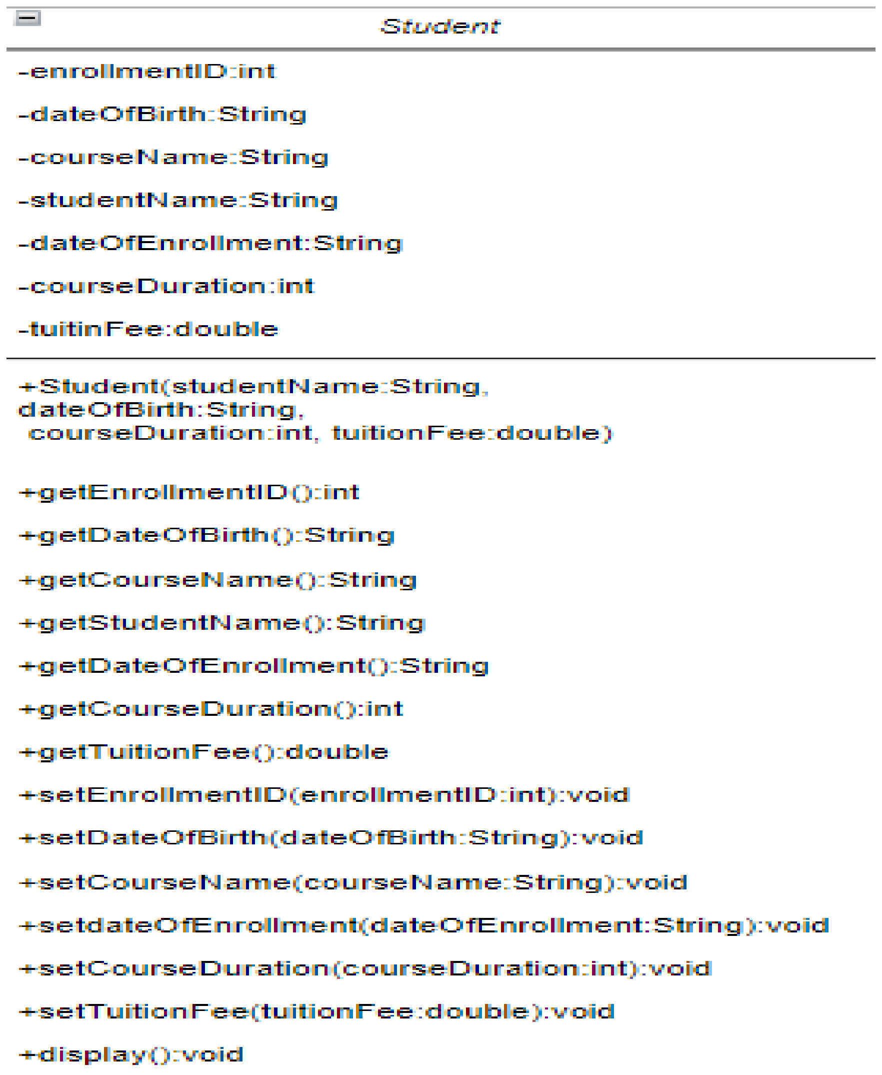
While the Dropout subclass will have qualities like remaining modules, months attended, and amount outstanding, the Regular subclass will have attributes like grades, attendance, modules details along with credit hrs and payment status.

The creation of the classes and the execution of actions in accordance with the provided specifications are the goals of this project. BlueJ, a robust programming environment that is perfect for learning Java programming, will be used to construct the project.

2 Class Diagram

Class diagram is a structured diagram which represents the composition of the system. It displays the attributes of the class and displays the relationships between classes it there is one. Class diagram showing very minimal descriptions of the class, it does not show the details of the of the program and about its methods. So, it is a can be taken as a blueprint of the program.

2.1 Class diagram of Student class



```
Student
-enrollmentID:int
-dateOfBirth:String
-courseName:String
-studentName:String
-dateOfEnrollment:String
-courseDuration:int
-tuitinFee:double

+Student(studentName:String,
dateOfBirth:String,
 courseDuration:int, tuitionFee:double)

+getEnrollmentID():int
+getDateOfBirth():String
+getCourseName():String
+getStudentName():String
+getDateOfEnrollment():String
+getCourseDuration():int
+getTuitionFee():double
+setEnrollmentID(enrollmentID:int):void
+setDateOfBirth(dateOfBirth:String):void
+setCourseName(courseName:String):void
+setdateOfEnrollment(dateOfEnrollment:String):void
+setCourseDuration(courseDuration:int):void
+setTuitionFee(tuitionFee:double):void
+display():void
```

Figure 1: Class diagram of Student class

2.2 Class diagram of Regular class

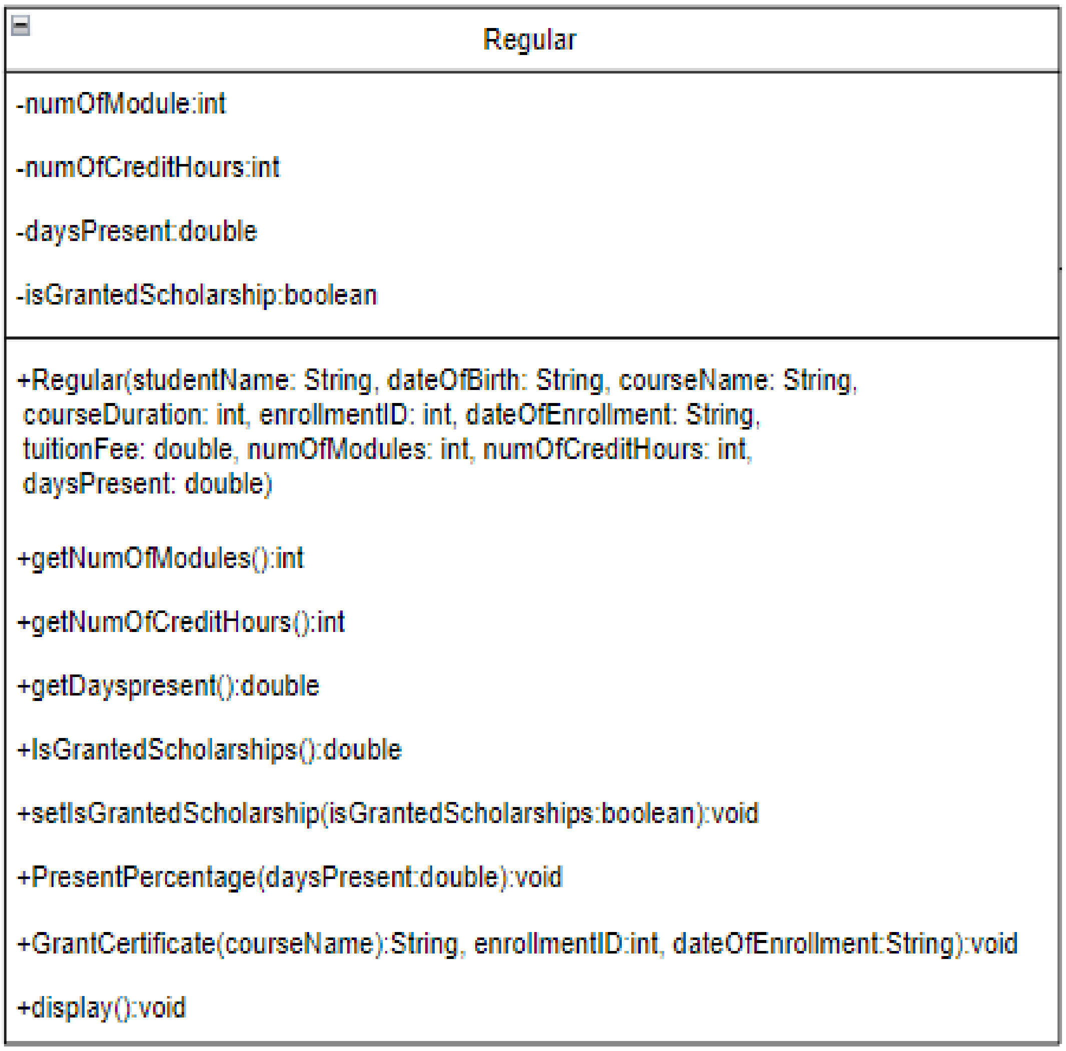

Regular
-numOfModule:int
-numOfCreditHours:int
-daysPresent:double
-isGrantedScholarship:boolean
+Regular(studentName: String, dateOfBirth: String, courseName: String, courseDuration: int, enrollmentID: int, dateOfEnrollment: String, tuitionFee: double, numOfModules: int, numOfCreditHours: int, daysPresent: double)
+getNumOfModules():int
+getNumOfCreditHours():int
+getDayspresent():double
+IsGrantedScholarships():double
+setIsGrantedScholarship(isGrantedScholarships:boolean):void
+PresentPercentage(daysPresent:double):void
+GrantCertificate(courseName):String, enrollmentID:int, dateOfEnrollment:String):void
+display():void

Figure 2: Class diagram of Regular class

2.3 Class diagram of Dropout class

\

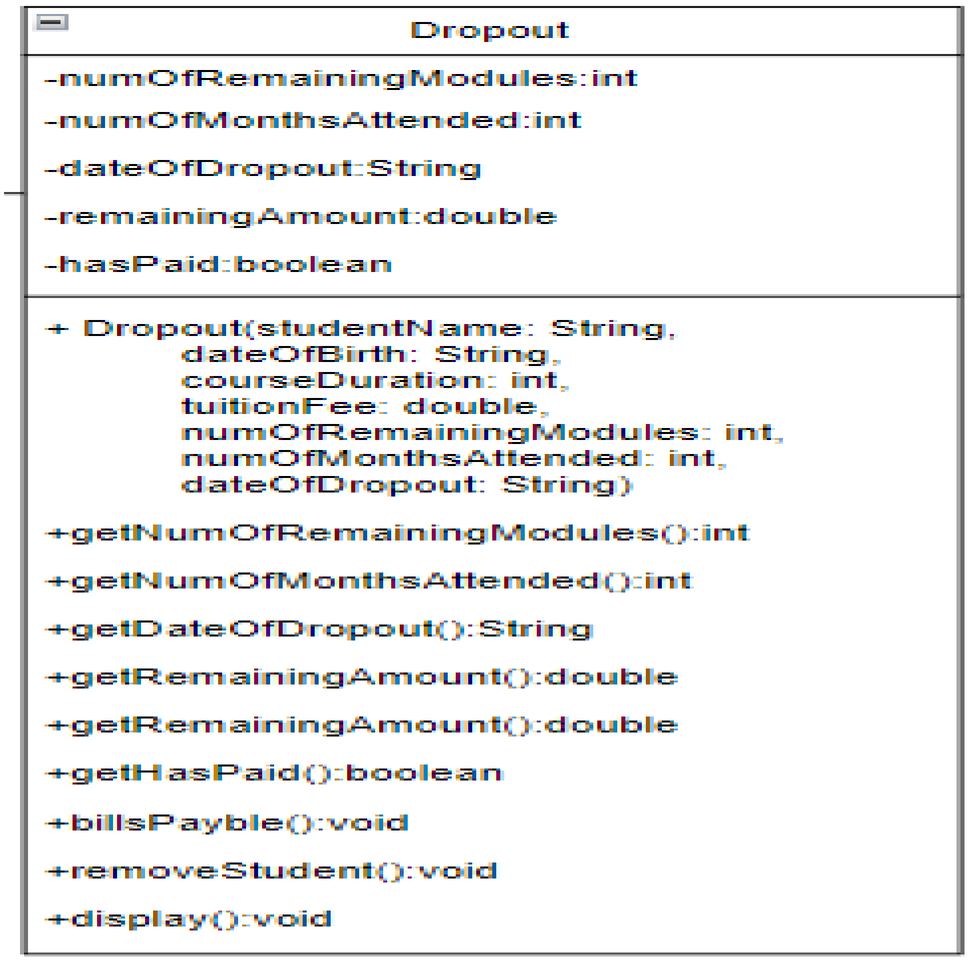

Dropout
-numOfRemainingModules:int
-numOfMonthsAttended:int
-dateOfDropout:String
-remainingAmount:double
-hasPaid:boolean
+ Dropout(studentName: String, dateOfBirth: String, courseDuration: int, tuitionFee: double, numOfRemainingModules: int, numOfMonthsAttended: int, dateOfDropout: String)
+getNumOfRemainingModules():int
+getNumOfMonthsAttended():int
+getDateOfDropout():String
+getRemainingAmount():double
+getRemainingAmount():double
+getHasPaid():boolean
+billsPayble():void
+removeStudent():void
+display():void

Figure 3: Class diagram of Dropout class

2.4 Combined Class diagram

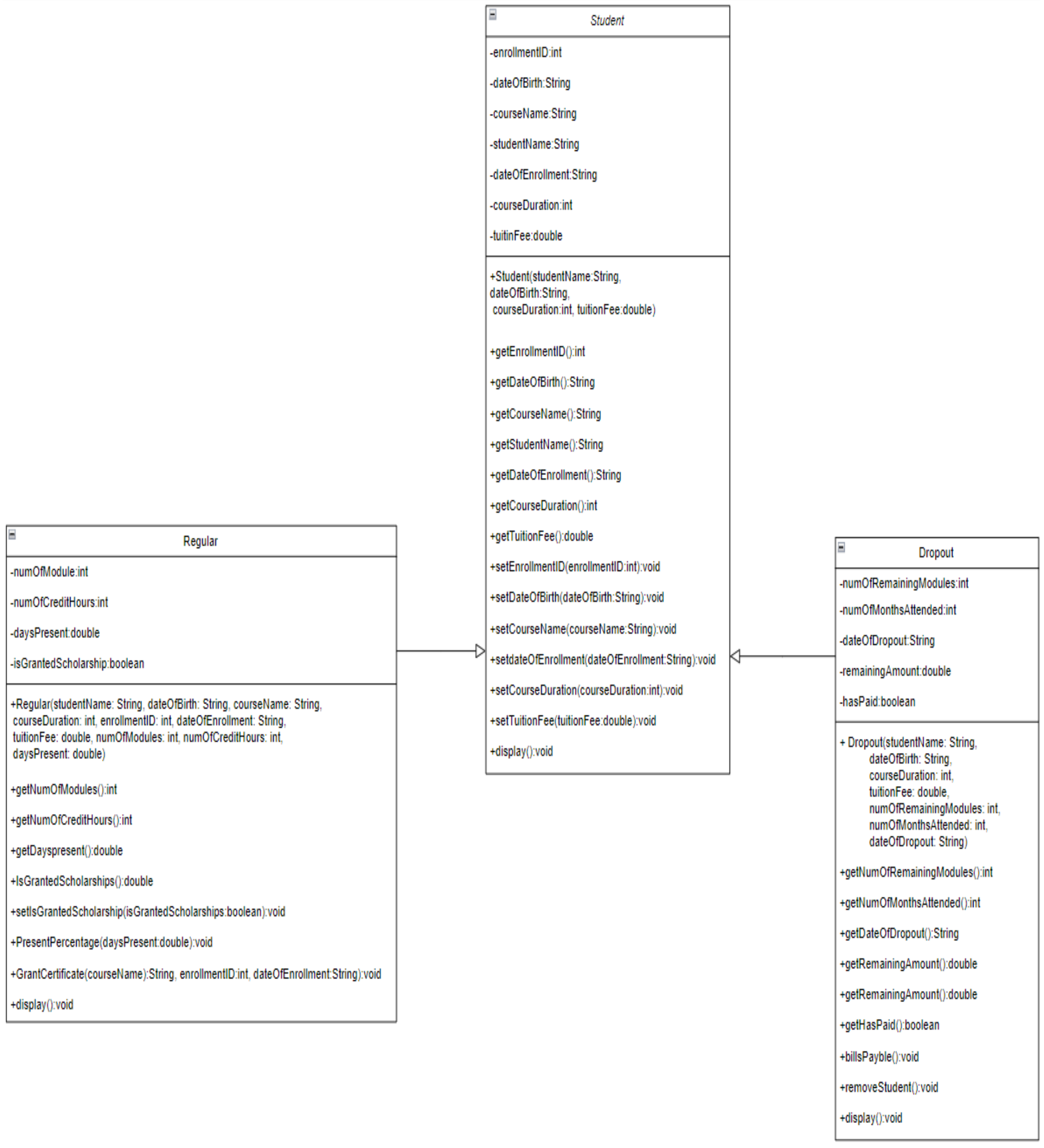

Student
-enrollmentID:int
-dateOfBirth:String
-courseName:String
-studentName:String
-dateOfEnrollment:String
-courseDuration:int
-tuitinFee:double
+Student(studentName:String, dateOfBirth:String, courseDuration:int, tuitionFee:double)
+getEnrollmentID():int
+getDateOfBirth():String
+getCourseName():String
+getStudentName():String
+getDateOfEnrollment():String
+getCourseDuration():int
+getTuitionFee():double
+setEnrollmentID(enrollmentID:int):void
+setDateOfBirth(dateOfBirth:String):void
+setCourseName(courseName:String):void
+setdateOfEnrollment(dateOfEnrollment:String):void
+setCourseDuration(courseDuration:int):void
+setTuitionFee(tuitionFee:double):void
+display():void
Regular
-numOfModule:int
-numOfCreditHours:int
-daysPresent:double
-isGrantedScholarship:boolean
+Regular(studentName: String, dateOfBirth: String, courseName: String, courseDuration: int, enrollmentID: int, dateOfEnrollment: String, tuitionFee: double, numOfModules: int, numOfCreditHours: int, daysPresent: double)
+getNumOfModules():int
+getNumOfCreditHours():int
+getDayspresent():double
+IsGrantedScholarships():double
+setIsGrantedScholarship(isGrantedScholarships:boolean):void
+PresentPercentage(daysPresent:double):void
+GrantCertificate(courseName):String, enrollmentID:int, dateOfEnrollment:String):void
+display():void
Dropout
-numOfRemainingModules:int
-numOfMonthsAttended:int
-dateOfDropout:String
-remainingAmount:double
-hasPaid:boolean
+ Dropout(studentName: String, dateOfBirth: String, courseDuration: int, tuitionFee: double, numOfRemainingModules: int, numOfMonthsAttended: int, dateOfDropout: String)
+getNumOfRemainingModules():int
+getNumOfMonthsAttended():int
+getDateOfDropout():String
+getRemainingAmount():double
+getRemainingAmount():double
+getHasPaid():boolean
+billsPayble():void
+removeStudent():void
+display():void

Figure 4:Combined class diagram

3 Pseudo Code

Pseudo Code is a description of the program or the code which is detailed yet simplified and is written in a simple and natural language which gives an overview on the program or the code. It is written in short phrases to write code for programs.

3.1 Pseudo Code for the parent class Student

```
Create Class Student

Declare enrollmentID as an integer and set access modifier as private
Declare dateOfBirth as a string and set access modifier as private
Declare courseName as a string and set access modifier as private
Declare studentName as a list of strings and set access modifier as private
Declare dateOfEnrollment as a string and set access modifier as private
Declare courseDuration as a list of int and set access modifier as private
Declare tuitionFee as a double and set access modifier as private



Create a constructor for the Student class that takes studentName, dateOfBirth
, courseDuration, and tuitionFee as parameters and initializes the
corresponding instance variables

Initialize the variable enrollmentID to 0
this.dateOfBirth = dateOfBirth
    courseName = ""
    this.studentName = studentName
    dateOfEnrollment = ""
    this.courseDuration = courseDuration
    this.tuitionFee = tuitionFee
```

Construct a getter method for **EnrollmentID** having access modifier public and returns the value as **enrollmentID**.

Construct a getter method for **DateOfBirth** having access modifier public and returns the value as for **dateOfBirth**.

Construct a getter method for **CourseName** having access modifier public and returns the value as **courseName**.

Construct a getter method for **StudentName** having access modifier public and returns the value as **studentname.**

Construct a getter method for **DateOfEnrollment** having access modifier public and returns the value as **dateOfEnrollment**.

Construct a getter method for **CourseDuration** having access modifier public and returns the value as **courseDuration**.

Construct a getter method for **TuitionFee** having access modifier public and returns the value as **tuitionFee**.

Construct a Setter method for **EnrollmentID** having no return type.

Construct a Setter method for **DateOfBirth** having no return type.

Construct a Setter method for **CourseName** having no return type.

Construct a Setter method for **StudentName** having no return type.

Construct a Setter method for **DateOfEnrollment** having no return type.

Construct a Setter method for **CourseDuration** having no return type.

Construct a Setter method for **TuitionFee** having no return type.

```
Create a display method that has no return type
If enrollmentID==0
Print “Enrollment ID not declared: ”
Else if courseName==”” or null
Print “course Name is not mentioned: ”
Else if dateOfEnrollment==”” or null
Print “:Date of Enrollment value is missing ”
Else
Print “EnrollmentId:”followed by enrollmentID value.
Print”DateOfBirth:”followed by dateOfBirth value.
Print”CourseName:” followed by courseName value.
Print”StudentName:” followed by studentName value.
Print”DateOfEnrollment:”followed by dateOfEnrolment value.
Print”CourseDuration:”followed by courseDuration value.
Print”TuitionFee:”followed by tuitionFee value
Print”Newline( for gap or space)
```

3.2 Pseudo Code for sub class Regular

Create a class Regular

Declare **numOfModules** as an integer and set a private access modifier.

Declare **numOfCreditHours** as an interger and set a private access modifier.

Declare **daysPresent** as an double and set a private access modifier.

Declare **isGrantedscholarship** as an boolean and set a private access modifier

Create a parametrized constructor in Regular class and pass **studentName** as String, **dateOfBirth** as String, **courseName** as String, **courseDuration** as integer, **enrollmentID** as Integer, **dateOfEnrollment** as String, **tuitionFee** as double, **numOfmodules** as integer, **numOfCreditHours** as integer and **daysPresent** as double parameters.

Invoke variables:-**studentName**, **dateOfBirth**, **courseDuration,** and **tuitionFee** in the constructor from Student class using keyword super.

```
//calling setter method using super keyword
      super.setCourseName(courseName);
       super.setEnrollmentID(enrollmentID);
       super.setDateOfEnrollment(dateOfEnrollment)
```

invoke variable **numOfModule**s in the constructor using the keyword this.

invoke variable **numOfCreditHours** in the constructor using the keyword this.
invoke **variabledaysPresent** in the constructor using the keyword this.
initialized **isGrantedScholarship** to false using keyword this.

Create a getter method for **numOfModules** having a public access modifier.
Create a getter method for **numOfCreditHours** having a public access modifier.
Create a getter method for **daysPresent** having a public access modifier.
Create a getter method for **isGrantedScholarship** having a public access modifier.
Create a setter method for i**sGrantedScholarship** having no return type.

Create a method named **PresentPercentage** with a public access modifier that takes **daysPresent** as a double parameter.

Calculate the present percentage by dividing the number of days present by the total number of days in the course, which is obtained by multiplying the course duration by 30. Store the result in a variable named **presentpercentage**.

Initialize an empty string variable named **disPlay**.

Set the **daysPresent** variable in the class to the value of the **daysPresent** parameter.

Check if the number of days present is greater than the total number of days in the course, which is obtained by multiplying the course duration by 30. If so, print a message stating that the student cannot attend more than the course days.

Otherwise, check the present percentage to determine the grade of the student. If the present percentage is between 80 and 100 (inclusive), set the value of **isGrantedScholarship** to true and set the value of **disPlay** to "A".

If the present percentage is less than 80 and greater than or equal to 60, set the value of **disPlay** to "B".

If the present percentage is less than 60 and greater than or equal to 40, set the value of **disPlay** to "C".

If the present percentage is less than 40 and greater than or equal to 20, set the value of **disPlay** to "D".

Otherwise, set the value of **disPlay** to "E".

Print a message stating the student's attendance percentage and grade using the values of **presentpercentage** and **disPlay**, respectively.

The method takes in three parameters: "**courseName**" as a string, **"enrollmentID**" as an integer, and **"dateOfEnrollment**" as a string.

Within the method, the student's name, the course name**, enrollment ID**, and **date of enrollment** are displayed using getter methods from the superclass "Student" using the "super" keyword.

If the student has been granted a scholarship **(isGrantedScholarship** is true), then a message is displayed stating that the scholarship has been granted. If the student has not met the attendance requirements for the certificate (**isGrantedScholarship** is false), then a message is displayed stating that the student has not met the attendance requirements.

Create a method display having no return type.

Call the display method of Student class use the keyword super.

```
    Print (“NameOfModules())”
    Print (“NumberOfCreditHours())”
 Print (“Days present ())”
```

3.3 Pseudo Code for sub class Dropout

Create Class Dropout

Declare **numOfRemainingModules** as a integer and set access modifier as private

Declare **numOfMonthsAttended** as a integer and set access modifier as private

Declare **dateOfDropout** as a String and set access modifier as private.

Declare **remainingAmoun**t as a double and set access modifier as private.

Declare **hasPaid** as a Boolean and set access modifier as private.

Create a parametrized constructor of the Dropout class and pass **dateOfBirth** in String, **courseDuration** in int, **tuitionFee** in double, **numOfRemainingModules** in int, **numOfMonthsAttended** in int, **dateOfDropoutparameters** in String.

Invoke variables: - **dateOfBirth**, **studentName**, **courseDuration**, **tuitionFee** the constructor from Student class using keyword super.
//calling setter method using super keyword
super.setCourseDuration(**courseDuration**);
super.setTuitionFee**(tuitionFee**);
super.setCourseName**("")**;
super.setEnrollmentID(**0**);
super.setDateOfEnrollment(**""**);
invoke variable **numOfRemainingModules** in the constructor using the keyword this.

invoke variable **numOfMonthsAttended** in the constructor using the keyword this.

invoke variable **dateOfDropout** in the constructor using the keyword this.

invoke variable **remainingAmoun**t in the constructor using the keyword this.

invoke variable **hasPaid** in the constructor using the keyword this.set it to false.

Construct a getter method for **NumOfRemainingModules** having access modifier public and returns the value as **NumOfRemainingModules**

Construct a getter method for **NumOfMonthsAttendedhaving** access modifier public and returns the value as **NumOfMonthsAttended.**

Construct a getter method for **DateOfDropout** having access modifier public and returns the value as **DateOfDropout**

Construct a getter method for **RemainingAmount** having access modifier public and returns the value as **RemainingAmount.**

Construct a getter method for **HasPaid** having access modifier public and returns the value as **HasPaid.**

Create a **billsPayble method**:

Initialize a variable **disPlay** as an empty string.

Calculate the remaining amount of tuition fee by subtracting the number of months attended from the course duration and multiplying the result with the tuition fee.

Check if the remaining amount of tuition fee is equal to zero.

If the remaining amount of tuition fee is equal to zero, set the value of **hasPaid** to true.

Assign the string "All bills cleared by the student" to **disPlay** if the remaining amount of tuition fee is equal to zero.

If the remaining amount of tuition fee is not equal to zero, assign the string "The total amount of bills needed to be paid Rs" followed by the value of **remainingAmount** to **disPlay.**

Print the value of **disPlay.**

Create a **remove method:**

If the student has paid all their bills, then.

Set the **date of birth** of the student to an empty string.

Set the **course name** of the student to an empty string.

Set the **name of the student** to an empty string.

Set the **date of enrollment** of the student to an empty string.

Set the duration of the course to 0.

Set the **tuition fee** for the course to 0.

Set the **number of remaining modules** to 0.

Set the **number of months attended** to 0.

Set the **date of dropout** to an empty string.

Set the **remaining amount** of tuition fee to 0.

Set the **enrollment ID** of the student to 0.

Otherwise,

Print the message "All bills not cleared."

End If

End the method **removeStudent**()

Create **display method:**

```
// Call the display method of the superclass (Student)
super.display();

// Print the additional information related to a dropped-out student
System.out.println("Number of remaining modules: " +
numOfRemainingModules);
System.out.println("Number of months attended: " + numOfMonthsAttended);
System.out.println("Date of dropout: " + dateOfDropout);
System.out.println("Remaining amount: Rs " + remainingAmount);
}
```

4 Method Description

4.1 Methods in Student class

Getter Method:

Getter Method helps us to access the variable which are set to private, it helps to make your data or program more secure. This method helps to return a value. Getter method in Student class is used to access the private variable. It starts with “get” keyword and is followed by the variable name.

Setter Method:

Setter method helps to assign/set values to the variable. This method does not return the any value rather it sets the variable for variable used in the program. Setter Method in Student class is used to set values to the variables. It starts with “set” keyword followed by the variable name.

Display Method:

The creation of Display Method in Student Class is to display several values according to the question. It is to display the Enrollment ID , Date Of Birth, Course Name, Student Name, Date Of Enrollment, Course Duration and Tuition Fee. The use of **if** statement has happened in this method with the condition that if the enrollmentID is equal to zero and courseName and dateOfEnrollment are empty, an error message will be displayed else Enrollment ID , Date Of Birth, Course Name, Student Name, Date Of Enrollment, Course Duration and Tuition Fee.will be displayed as followed by value according to their data type.

4.2 Methods in Regular class

Getter Method:

Getter Method helps us to access the variable which are set to private, it helps to make your data or program more secure. This method helps to return a value. Getter method in Regular class is used to access the private variable. It starts with “get” keyword and is followed by the variable name.

Setter Method:

Setter method helps to assign/set values to the variable. This method does not return the any value rather it sets the variable for variable used in the program. Setter Method in Regular class is used to set values to the variables. It starts with “set” keyword followed by the variable name.

PresentPrecentage method:

The PresentPercentage method is a method that calculates the percentage of days a customer has attended a course. It takes two parameters: CourseDuration, which is the total duration of the course in days, and DaysPresent, which is the number of days the customer has attended the course.

The method starts with an if statement that checks whether the instrument is available for rent. If the condition is true, which means the instrument is not available for rent, it prints out a message saying that the instrument is not available. If the condition is false, the program moves to the else part of the statement.

In the else part, the super keyword is used to call the setter method for NameoftheCustomer, MobileNumofCustomer, and Panofthecustomer from the parent class. This updates the value for these variables based on the customer details provided in the subclass.

Finally, the program calculates the percentage of days the customer has attended the course by dividing the number of days attended by the total course duration and multiplying the result by 100. This value is then returned by the method.

GrandCertificate Method

The GrantCertificate method takes two parameters - CourseDuration and DaysPresent. CourseDuration is an integer value representing the total duration of the course in days. DaysPresent is a double value representing the number of days the student was present during the course.

The method first calculates the percentage of days the student was present in the course using the formula (DaysPresent / CourseDuration) * 100. If this percentage is greater than or equal to the required percentage for certification, which is 80% in this case, then the method generates a certificate by printing a message to the console stating that -The scholarship has been granted.If the percentage is less than the required percentage, then the method prints a message stating that -The student has not met the attendance requirements for the certificate.

In summary, the GrantCertificate method checks if the student has met the attendance requirements for the course and grants a certificate if the student has met the requirements.

Display Method

It is used to display the information about a regular student, including the number of modules, number of credit hours, and days present.

The method starts by calling the display () method of the parent class (Student) using the super keyword. This is done to display the common information about a student, such as the name, roll number, and course name.

After displaying the common information, the method displays the number of modules, number of credit hours, and days present specific to a regular student. It does this by accessing the instance variables numOfModules, numOfCreditHours, and daysPresent, respectively.

Finally, the method prints out the information using the System.out.println() method. The output is in the form of a string concatenated with the value of the respective instance variables.

4.3Method in Dropout class

Getter Method:

Getter Method helps us to access the variable which are set to private, it helps to make your data or program more secure. This method helps to return a value. Getter method in 'dropout class' is used to access the private variable. It starts with "get" keyword and is followed by the variable name.

Setter Method:

Setter method helps to assign/set values to the variable. This method does not return the any value rather it sets the variable for variable used in the program. Setter Method in 'dropout class' is used to set values to the variables. It starts with "set" keyword followed by the variable name.

BillsPayble Method:

The ‘billsPayble()’ method calculates the remaining tuition fees that the student needs to pay by subtracting the tuition fees paid from the total tuition fees and then sets the ‘hasPaid’ field accordingly. If the remaining amount is 0, then the method sets the ‘hasPaid’ field to true and displays a message that all bills have been cleared. Otherwise, the method displays the total amount of bills needed to be paid.

RemoveStudent Method:

The removeStudent() method removes the student from the system if they have paid all their bills. Otherwise, it prints an error message stating that all bills have not been cleared.

Display Method:

The ‘display ()’ method overrides the parent class's ‘display()’ method to display information related to a regular student who has dropped out of the course. It displays the student's name, date of birth, course name, enrollment ID, course duration, tuition fee, number of remaining modules, number of months attended, date of dropout, and remaining amount to be paid.

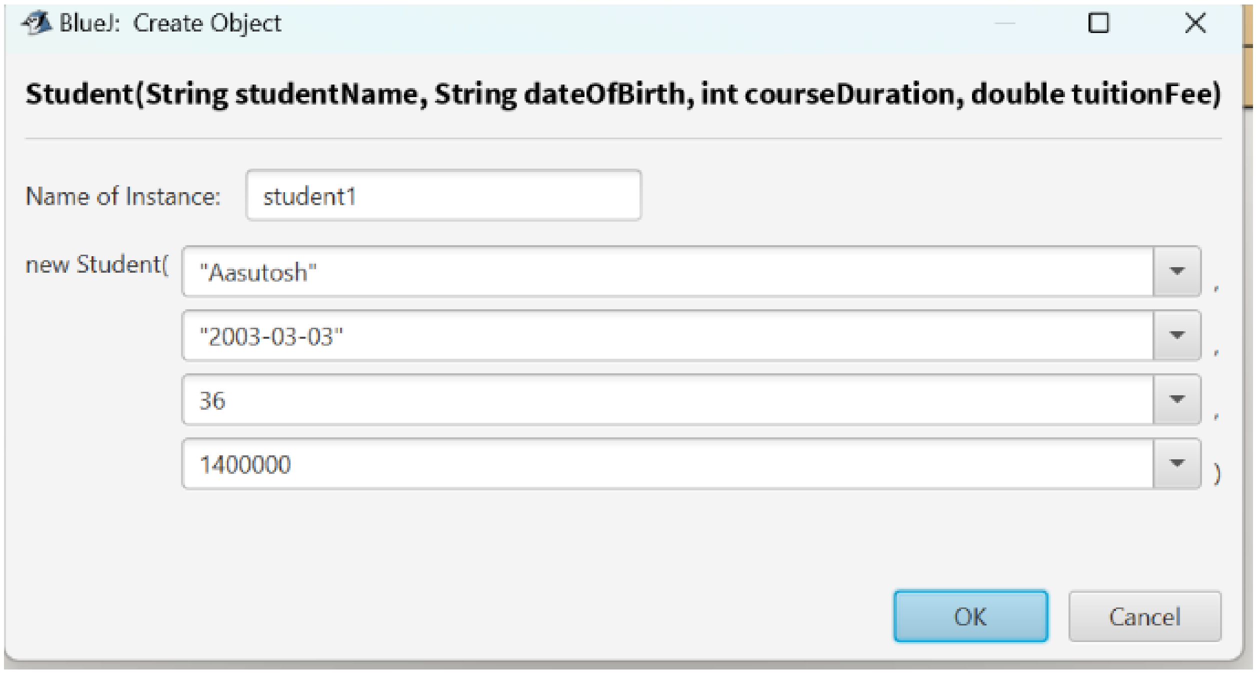
5 Testing

5.1 Inspecting Student class,creating object inserting details and re-inspecting the Student class

Test no.1	Inspecting student Class,Creating object inserting details and after inspecting again initializing value of enrolment id ,courseName and dateOfenrollment and reinspecting it
Action	Creating a Regular Student object with the following details: StudentName:" Aasutosh" DateOfBirth:"2003-03-03" CourseDuration:36 TuitionFee:1400000 1 Inspecting the object to ensure that all values are initialized correctly. 2 Use the setter methods to insert values for enrolment id, course name, and date of enrollment. 3 Inspect the object again to ensure that the values have been updated correctly.
Expected Result	1.The Student object is created successfully with all details initialized correctly. 2.All values of the Student object are correct and match the values provided. 3.The values for enrolment id, course name, and date of enrollment are inserted correctly.

	4.The updated values of the student object match the values inserted.
Actual Result	After inserting all the values, the student object was properly formed and the values for Enrolment ID, Course Name, and Date of Enrollment were properly updated.
Conclusion	**The test was successful.**

Table 1: First test

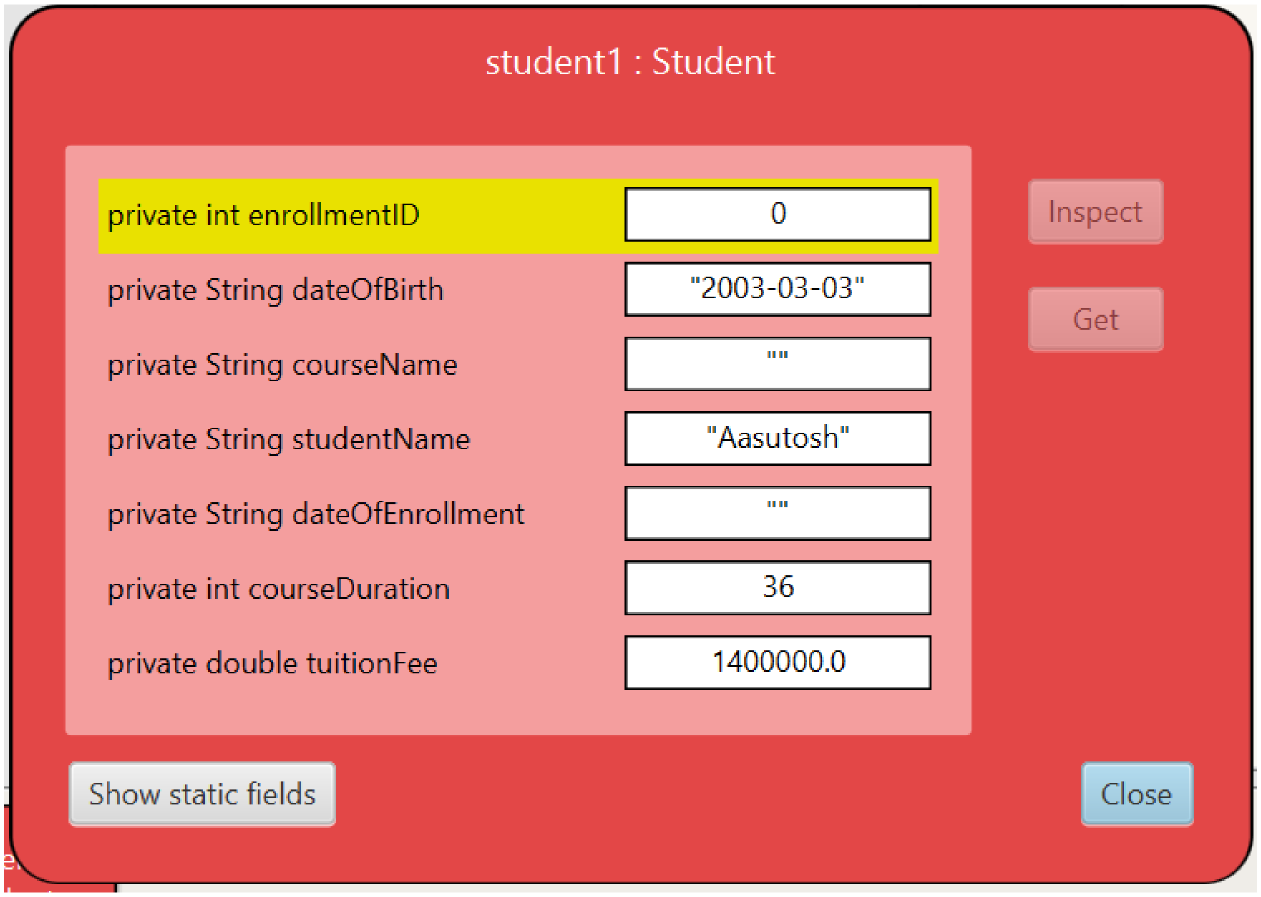


Student(String studentName, String dateOfBirth, int courseDuration, double tuitionFee)

Name of Instance: student1

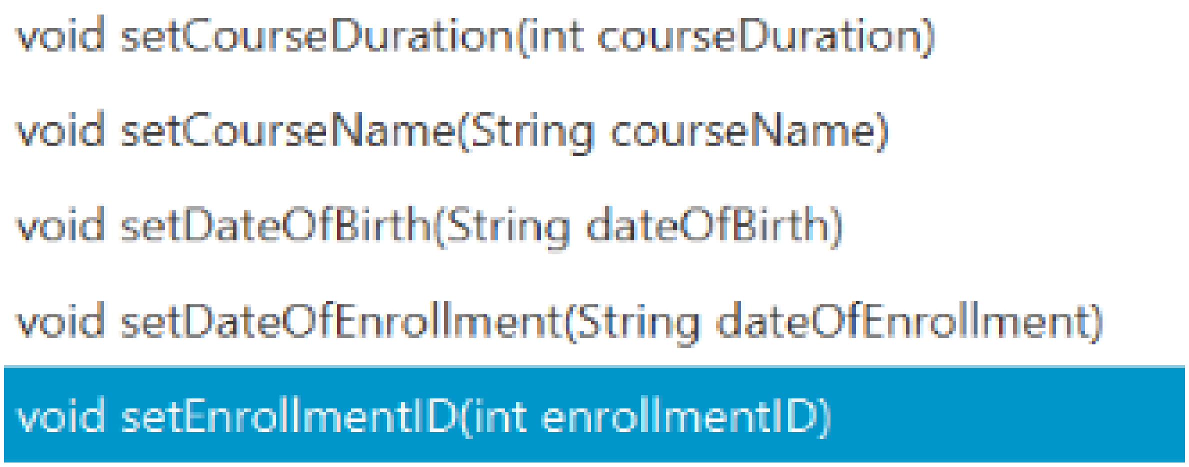
new Student("Aasutosh" , "2003-03-03" , 36 , 1400000)

Figure 5:Assigning value in Student



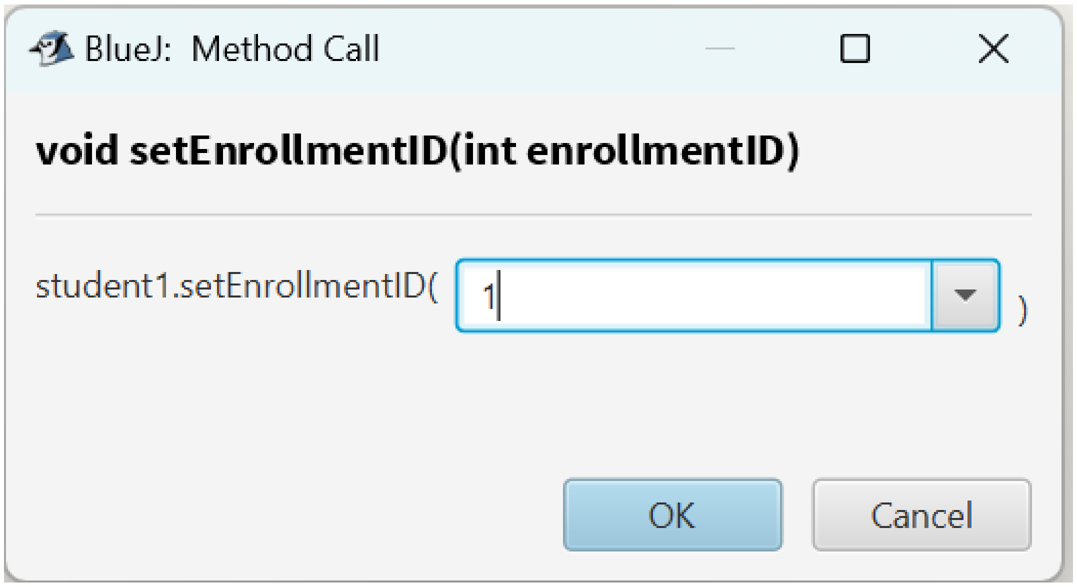
```
private int enrollmentID              0
private String dateOfBirth            "2003-03-03"
private String courseName             ""
private String studentName            "Aasutosh"
private String dateOfEnrollment       ""
private int courseDuration            36
private double tuitionFee             1400000.0
```

Figure 6: Inspection of Student



```
void setCourseDuration(int courseDuration)
void setCourseName(String courseName)
void setDateOfBirth(String dateOfBirth)
void setDateOfEnrollment(String dateOfEnrollment)
void setEnrollmentID(int enrollmentID)
```

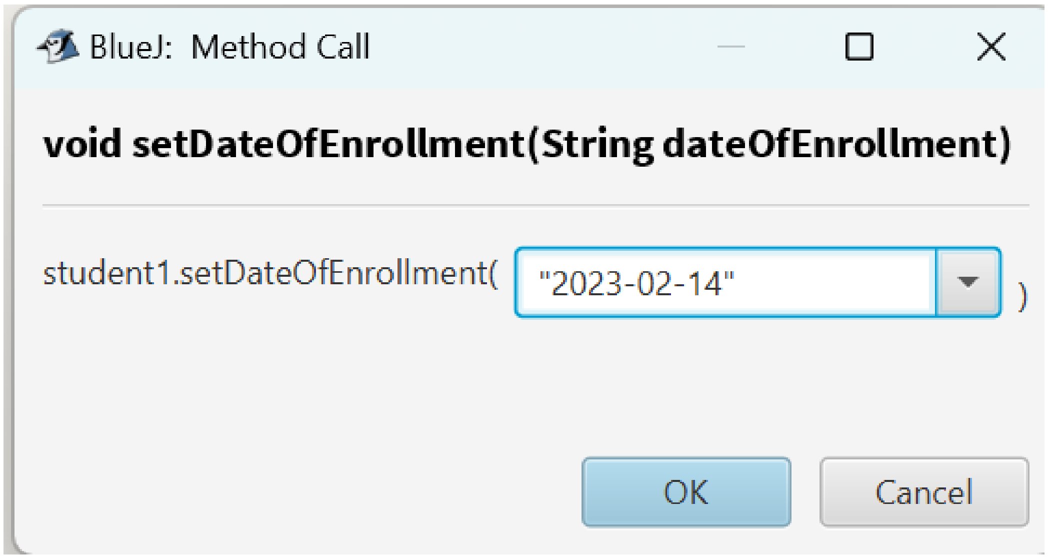
Figure 7:Selecting method for inserting data


BlueJ: Method Call

void setEnrollmentID(int enrollmentID)

student1.setEnrollmentID(1)

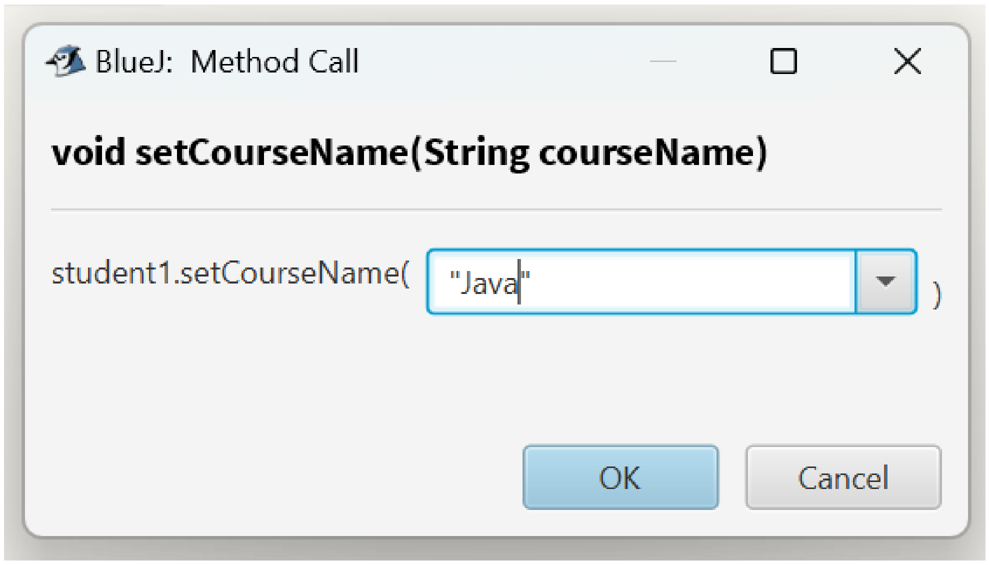
Figure 8: Inserting enrolment id value in Student


BlueJ: Method Call

void setDateOfEnrollment(String dateOfEnrollment)

student1.setDateOfEnrollment("2023-02-14")

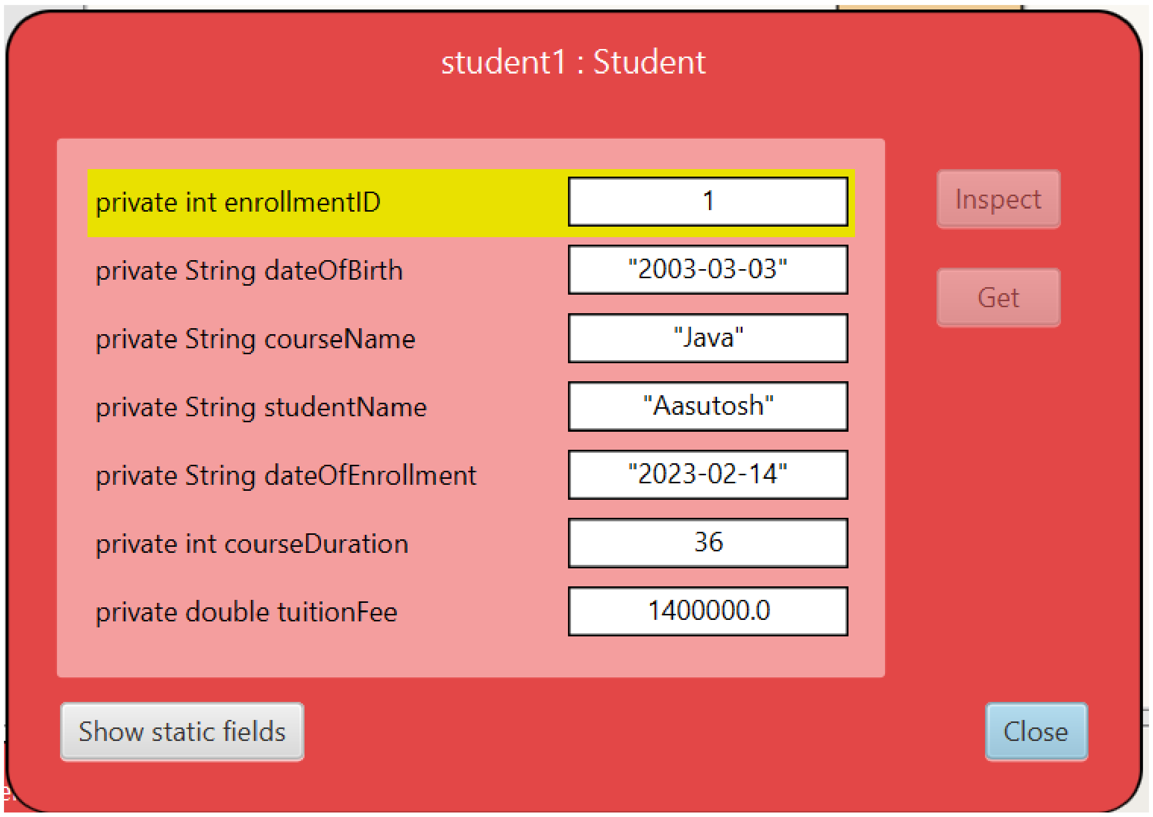
Figure 9: Inserting dateOfenrolment value in Student


BlueJ: Method Call

void setCourseName(String courseName)

student1.setCourseName("Java")

Figure 10: Inserting Coursename value in Student


student1 : Student

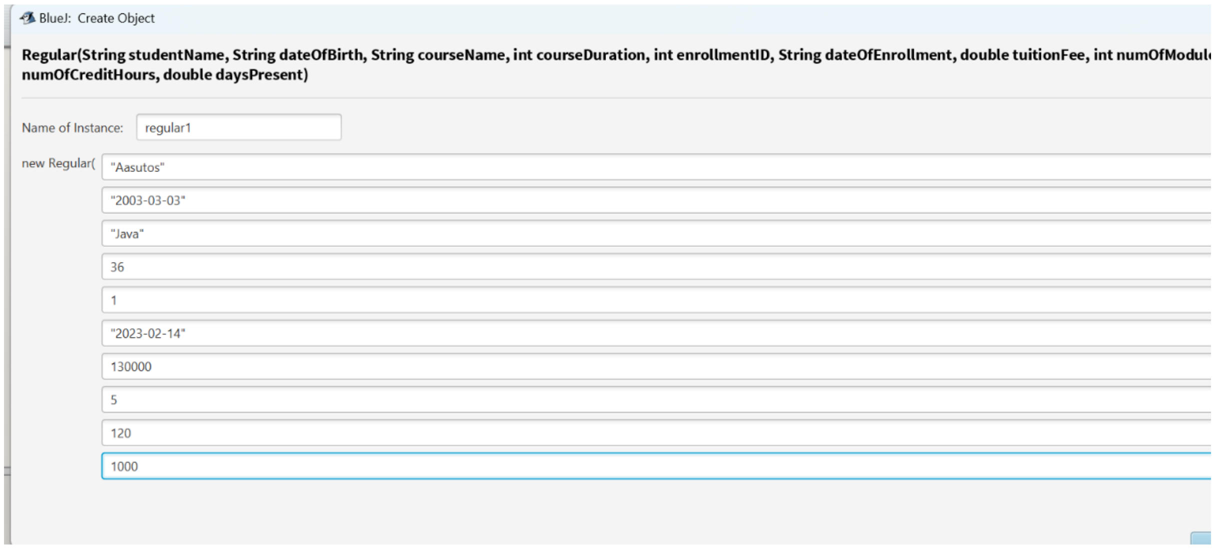
private int enrollmentID	1
private String dateOfBirth	"2003-03-03"
private String courseName	"Java"
private String studentName	"Aasutosh"
private String dateOfEnrollment	"2023-02-14"
private int courseDuration	36
private double tuitionFee	1400000.0

Figure 11: Re-Inspection Of student

5.2 Inspecting Regular class,creating object and re-inspecting the regular class value and checking it output.

Test no: 2	**Inspecting Regular class,Creating object,and re-inspecting it.**
Action	Creating a ‘Regular’Object with following details: Student name:”Aasutos” Date Of Birth:”2003-03-03” Course Duration:36 Coursename:”Java” Enrollment ID:1 Date Of Enrollment:”2023-02-14” Tuition Fee:130000 Num of Modules:5 CreditHours:120 Days Present:1000
Expected Result	The ‘Regular’Object should be created with the specified details.
Actual Result	The ‘Regular’ Object was created with the specified details.
Conclusion	***The test was successful***

Table 2: Second Test



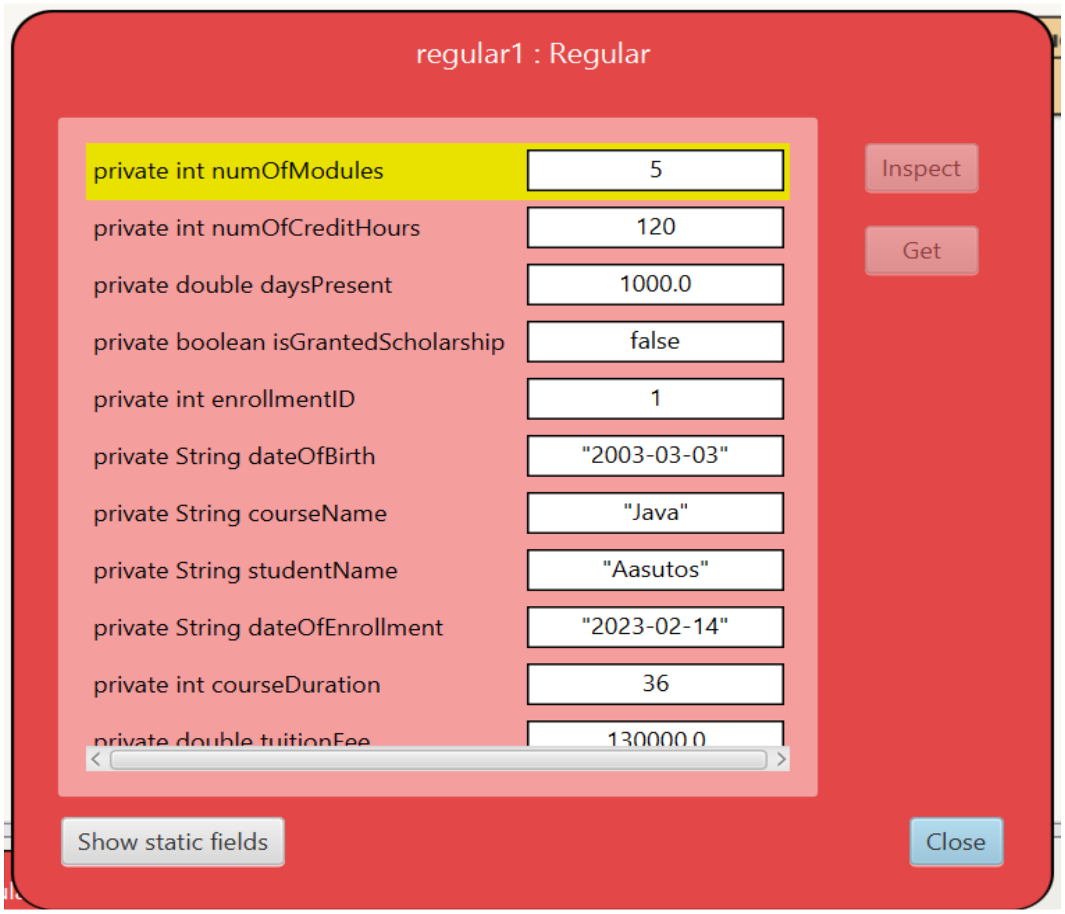
BlueJ: Create Object

Regular(String studentName, String dateOfBirth, String courseName, int courseDuration, int enrollmentID, String dateOfEnrollment, double tuitionFee, int numOfModul numOfCreditHours, double daysPresent)

Name of Instance: regular1

new Regular("Aasutos"
"2003-03-03"
"Java"
36
1
"2023-02-14"
130000
5
120
1000

Figure 12: Assigning student value to Regular



regular1 : Regular

private int numOfModules	5
private int numOfCreditHours	120
private double daysPresent	1000.0
private boolean isGrantedScholarship	false
private int enrollmentID	1
private String dateOfBirth	"2003-03-03"
private String courseName	"Java"
private String studentName	"Aasutos"
private String dateOfEnrollment	"2023-02-14"
private int courseDuration	36
private double tuitionFee	130000.0

Figure 13: Inspection of ‘Regular’ class

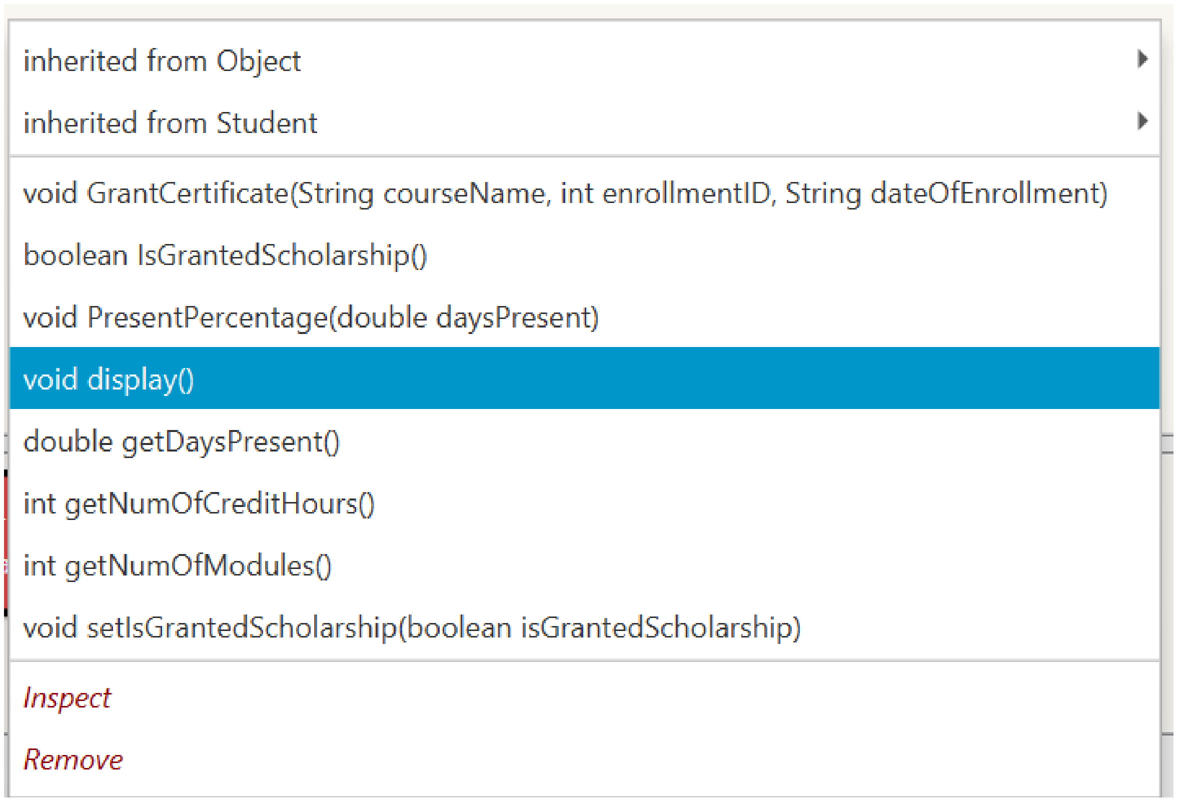
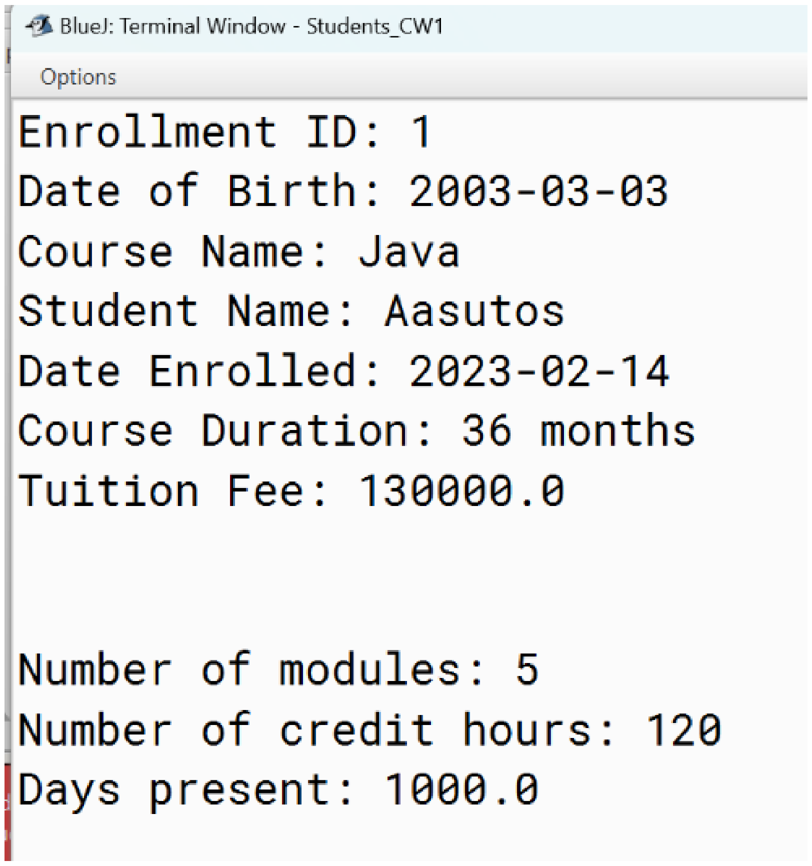

inherited from Object
inherited from Student
void GrantCertificate(String courseName, int enrollmentID, String dateOfEnrollment)
boolean IsGrantedScholarship()
void PresentPercentage(double daysPresent)
void display()
double getDaysPresent()
int getNumOfCreditHours()
int getNumOfModules()
void setIsGrantedScholarship(boolean isGrantedScholarship)
Inspect
Remove

Figure 14: Checking Output of object.



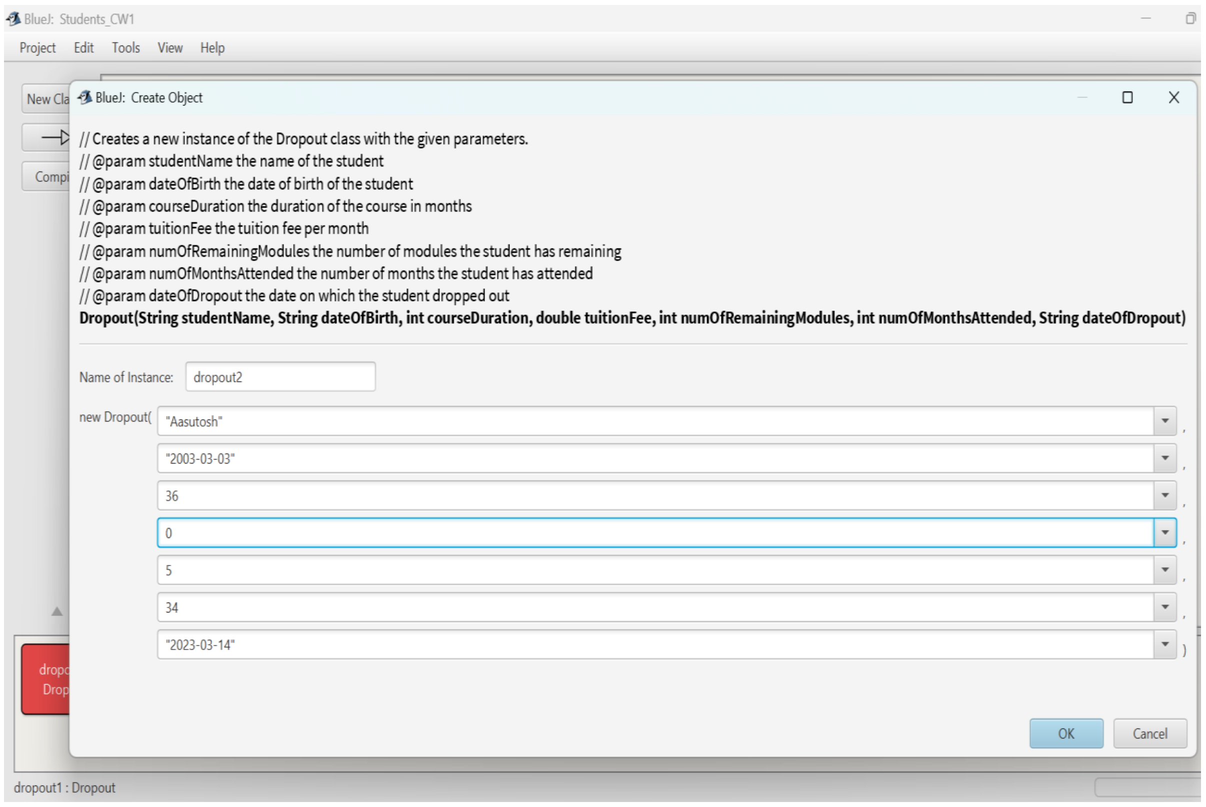
```
Enrollment ID: 1
Date of Birth: 2003-03-03
Course Name: Java
Student Name: Aasutos
Date Enrolled: 2023-02-14
Course Duration: 36 months
Tuition Fee: 130000.0


Number of modules: 5
Number of credit hours: 120
Days present: 1000.0
```

Figure 15: Displaying Output of created Object.

5.3 Inspecting ‘Dropout’ class, creating object and re-inspecting the ‘Dropout’ class value and checking it output.

Test no:3	***Inspecting ‘Dropout’ class, creating object and reinspecting the ‘Dropout’ class value and checking it output.***
Action	***Create a Dropout object with valid input parameters***
Expected Result	***The Dropout object should be successfully created with the provided details***
Actual Result	***The Dropout object was created successfully with the specified details***
Conclusion	**The test was successful**

Table 3: Third table

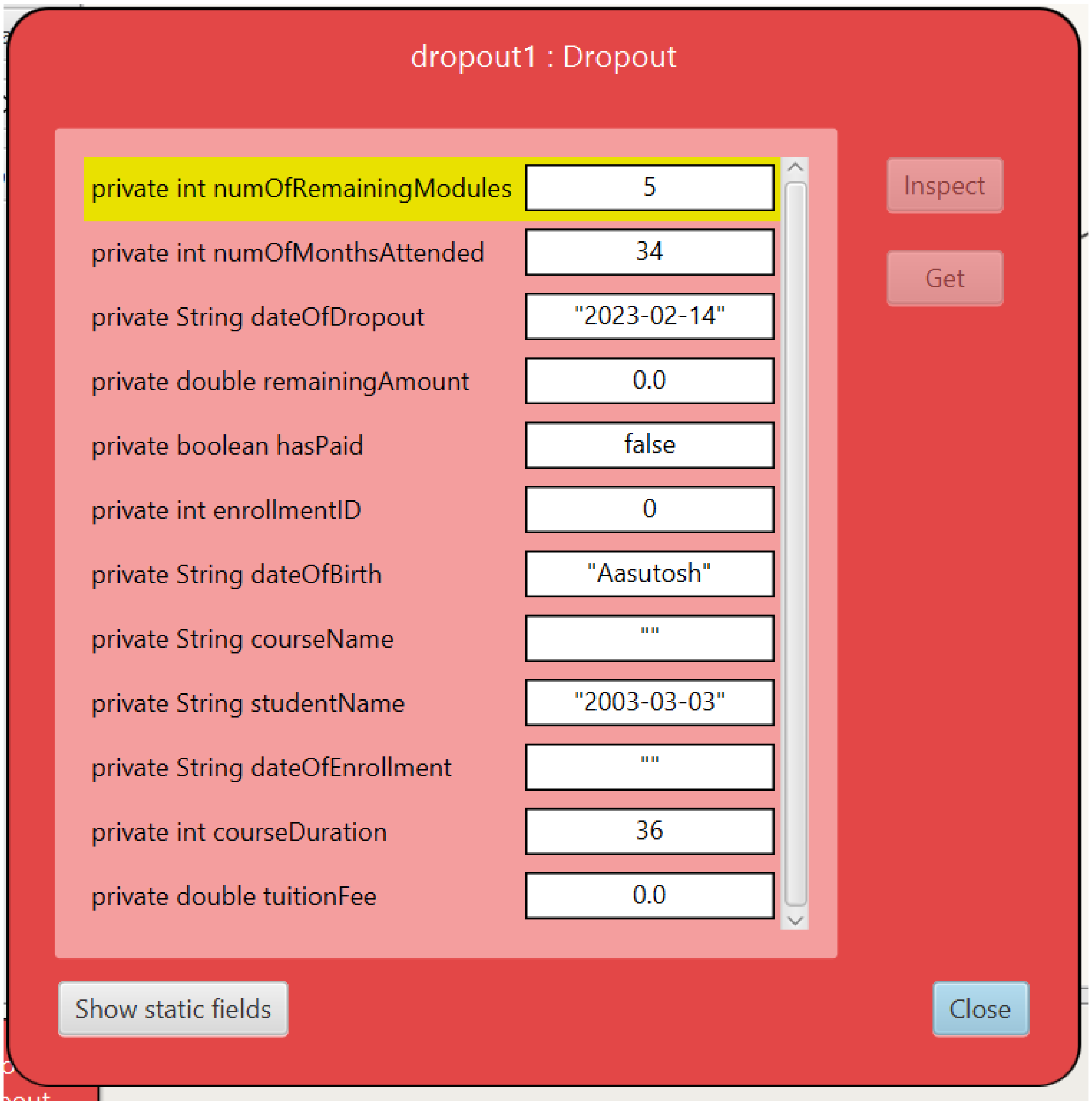


```
// Creates a new instance of the Dropout class with the given parameters.
// @param studentName the name of the student
// @param dateOfBirth the date of birth of the student
// @param courseDuration the duration of the course in months
// @param tuitionFee the tuition fee per month
// @param numOfRemainingModules the number of modules the student has remaining
// @param numOfMonthsAttended the number of months the student has attended
// @param dateOfDropout the date on which the student dropped out
Dropout(String studentName, String dateOfBirth, int courseDuration, double tuitionFee, int numOfRemainingModules, int numOfMonthsAttended, String dateOfDropout)
```

Name of Instance: dropout2

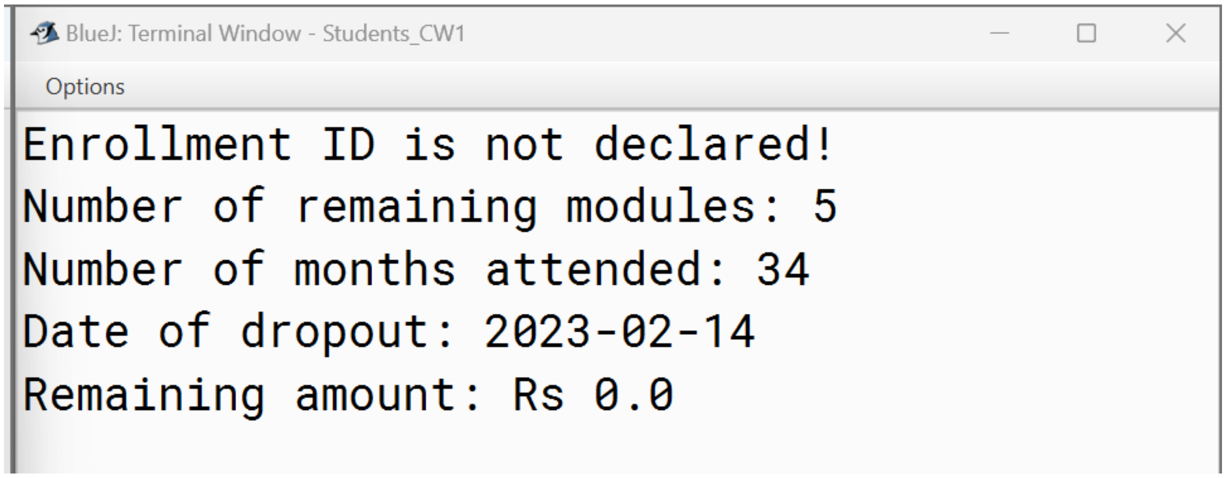
```
new Dropout( "Aasutosh",
             "2003-03-03",
             36,
             0,
             5,
             34,
             "2023-03-14" )
```

Figure 16: Assigning student value to Dropout.


dropout1 : Dropout

```
private int numOfRemainingModules      5
private int numOfMonthsAttended        34
private String dateOfDropout           "2023-02-14"
private double remainingAmount         0.0
private boolean hasPaid                false
private int enrollmentID               0
private String dateOfBirth             "Aasutosh"
private String courseName              ""
private String studentName             "2003-03-03"
private String dateOfEnrollment        ""
private int courseDuration             36
private double tuitionFee              0.0
```

Figure 17: Inspection of dropout with specific details



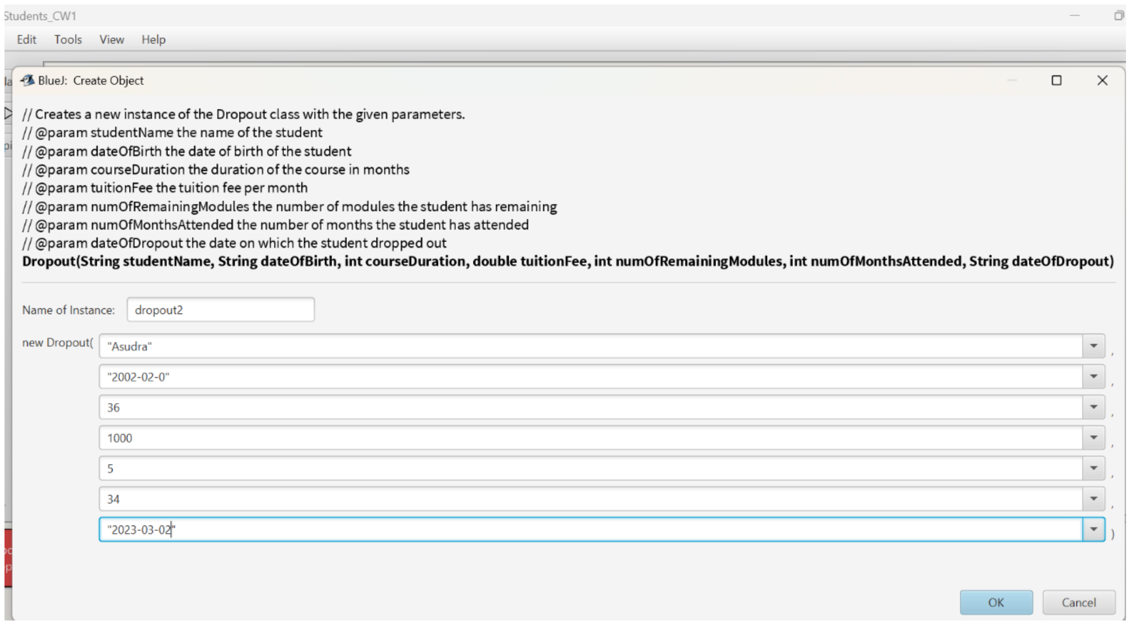
```
Enrollment ID is not declared!
Number of remaining modules: 5
Number of months attended: 34
Date of dropout: 2023-02-14
Remaining amount: Rs 0.0
```

Figure 18: Displaying output of created object with specific details

<u>5.4 Inspecting 'Dropout' class, creating object and re-inspecting the 'Dropout' class value and checking bills payable output.</u>

Test no:4	***Inspecting 'Dropout' class, creating object and reinspecting the 'Dropout' class value and checking it bills payable output.***
Action	***Create a Dropout object with valid input parameters***
Expected Result	***The Dropout object should be successfully created with the provided details***
Actual Result	***The Dropout object was created successfully with the specified details***
Conclusion	**The test was successful**

Table 4: Fourth table

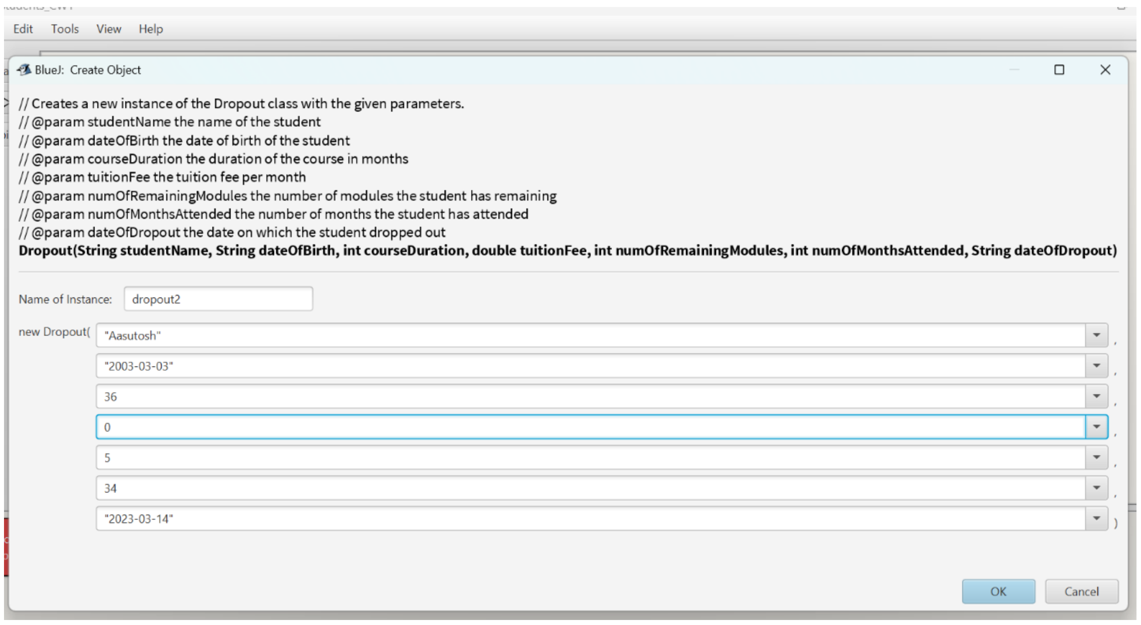


```
// Creates a new instance of the Dropout class with the given parameters.
// @param studentName the name of the student
// @param dateOfBirth the date of birth of the student
// @param courseDuration the duration of the course in months
// @param tuitionFee the tuition fee per month
// @param numOfRemainingModules the number of modules the student has remaining
// @param numOfMonthsAttended the number of months the student has attended
// @param dateOfDropout the date on which the student dropped out
Dropout(String studentName, String dateOfBirth, int courseDuration, double tuitionFee, int numOfRemainingModules, int numOfMonthsAttended, String dateOfDropout)
```

Name of Instance: dropout2

new Dropout("Asudra", "2002-02-0", 36, 1000, 5, 34, "2023-03-02")

Figure 19 Assigning student value to Dropout. Having tuition fee set to 1000.

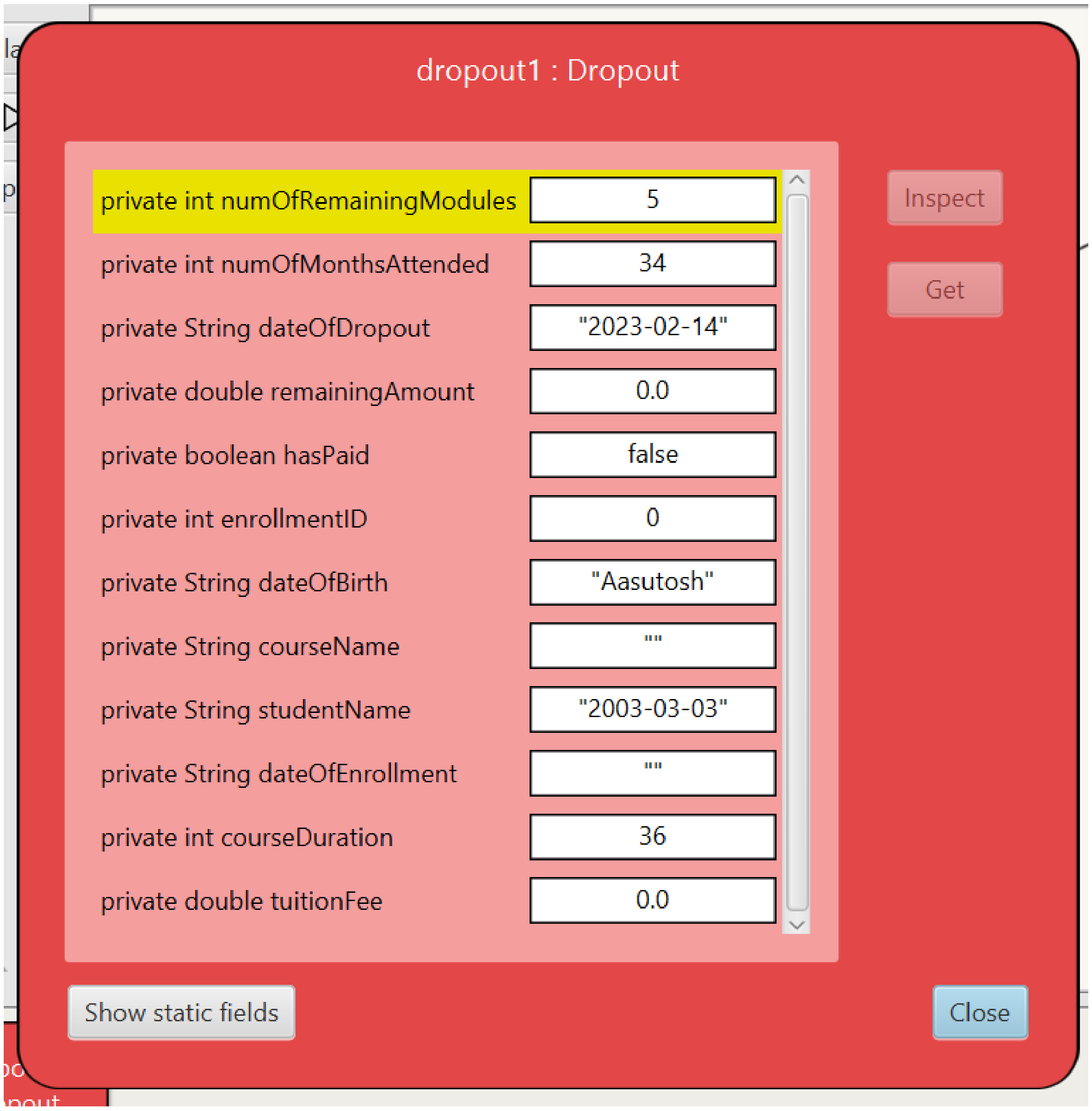


```
// Creates a new instance of the Dropout class with the given parameters.
// @param studentName the name of the student
// @param dateOfBirth the date of birth of the student
// @param courseDuration the duration of the course in months
// @param tuitionFee the tuition fee per month
// @param numOfRemainingModules the number of modules the student has remaining
// @param numOfMonthsAttended the number of months the student has attended
// @param dateOfDropout the date on which the student dropped out
Dropout(String studentName, String dateOfBirth, int courseDuration, double tuitionFee, int numOfRemainingModules, int numOfMonthsAttended, String dateOfDropout)
```

Name of Instance: dropout2

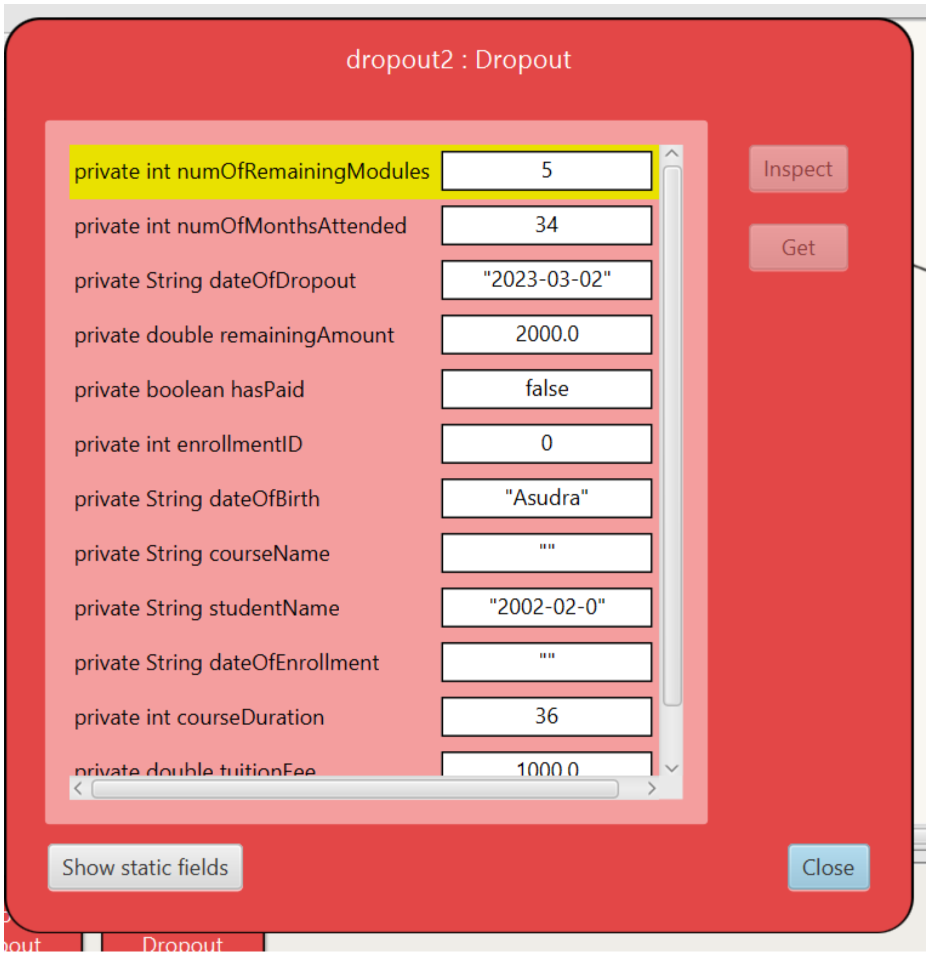
new Dropout("Aasutosh", "2003-03-03", 36, 0, 5, 34, "2023-03-14")

Figure 20 Assigning student value to Dropout having tuition fee set to zero.



private int numOfRemainingModules	5
private int numOfMonthsAttended	34
private String dateOfDropout	"2023-02-14"
private double remainingAmount	0.0
private boolean hasPaid	false
private int enrollmentID	0
private String dateOfBirth	"Aasutosh"
private String courseName	""
private String studentName	"2003-03-03"
private String dateOfEnrollment	""
private int courseDuration	36
private double tuitionFee	0.0

Figure 21: Inspection of dropout1 with specific details



private int numOfRemainingModules	5
private int numOfMonthsAttended	34
private String dateOfDropout	"2023-03-02"
private double remainingAmount	2000.0
private boolean hasPaid	false
private int enrollmentID	0
private String dateOfBirth	"Asudra"
private String courseName	""
private String studentName	"2002-02-0"
private String dateOfEnrollment	""
private int courseDuration	36
private double tuitionFee	1000.0

Figure 22: Inspection of dropout2 with specific details

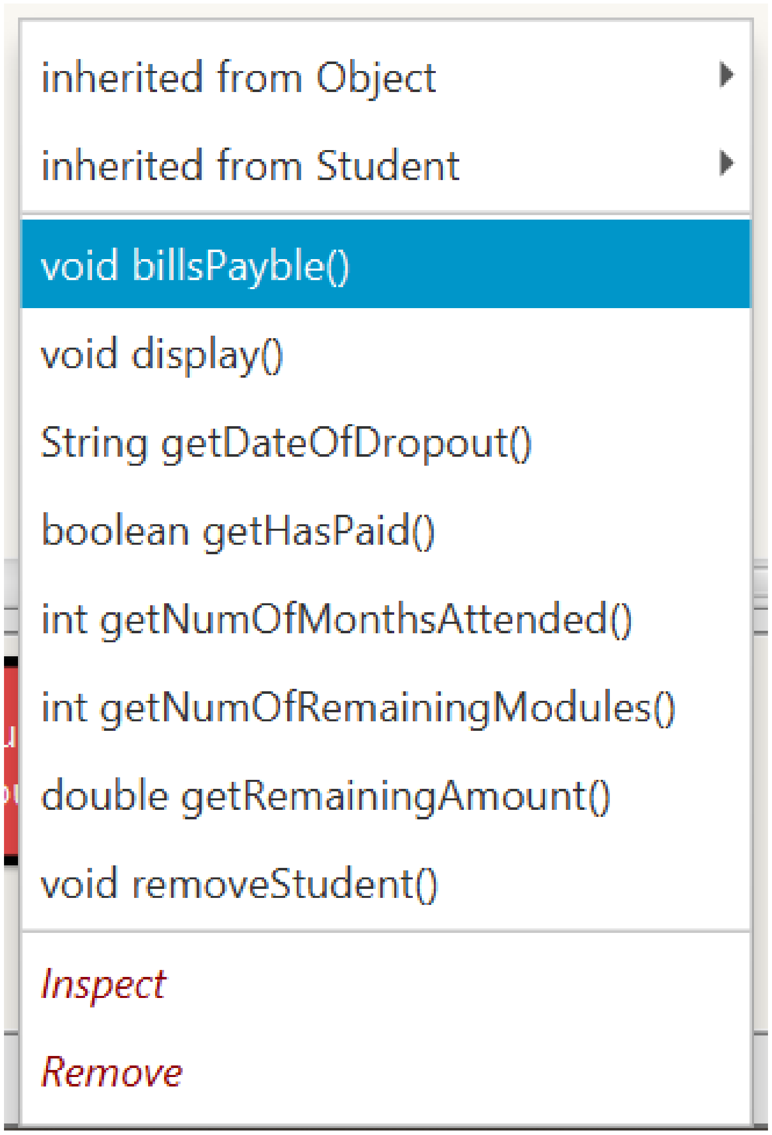
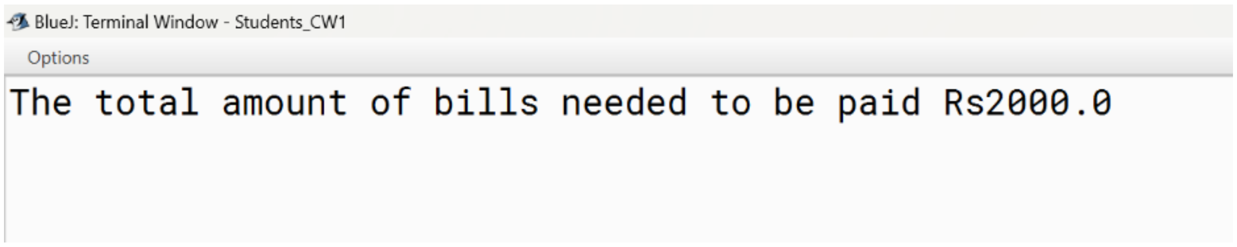
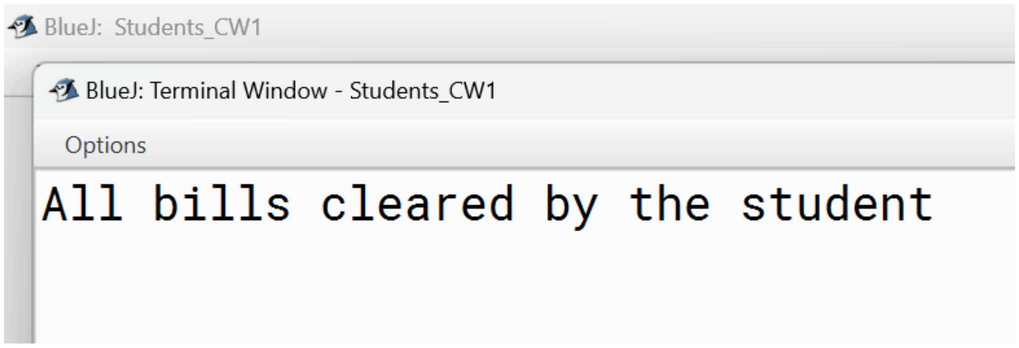


Figure 23: Showing process to check bills.



```
The total amount of bills needed to be paid Rs2000.0
```

Figure 24:Bills output of student name Asudra.



```
All bills cleared by the student
```

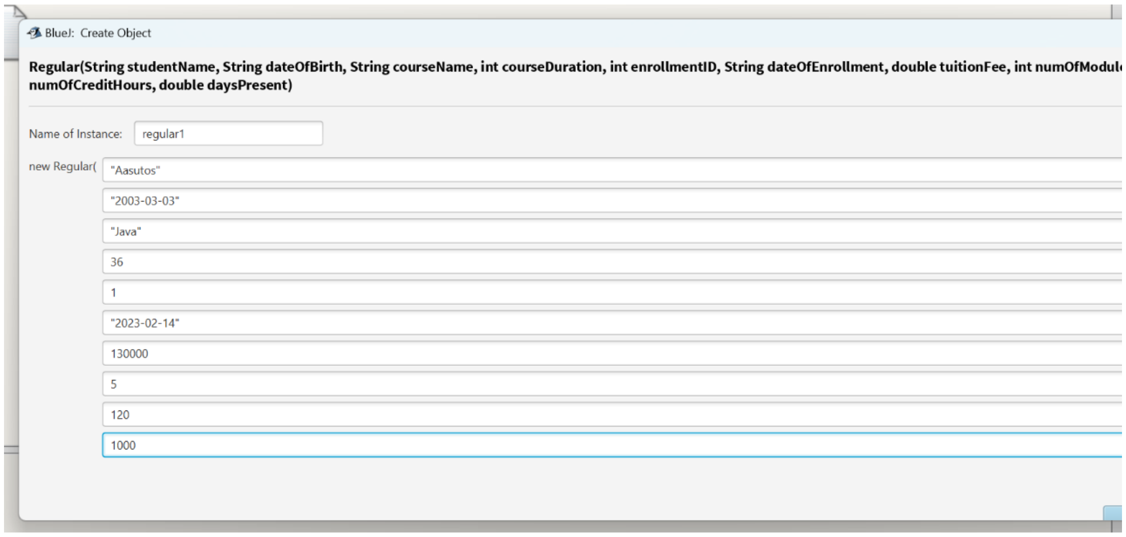
Figure 25:Bills output of student name Aasutosh.

5.5 Inspecting Regular class, creating object and re-inspecting the regular class value and checking present grade output.

Test no: 5	**Inspecting Regular class, creating object, and re-inspecting it.**
Action	Creating a ‘Regular ‘Object with following details: Student name:”Aasutos” Date Of Birth:”2003-03-03” Course Duration:36 Coursename:” Java” Enrollment ID:1 Date Of Enrollment:”2023-02-14” Tuition Fee:130000 Num of Modules:5 CreditHours:120 Days Present:1000

Expected Result	The ‘Regular ‘Object should be created with the specified details.
Actual Result	The ‘Regular’ Object was created with the specified details.
Conclusion	***The test was successful***

Table 5: Fifth table

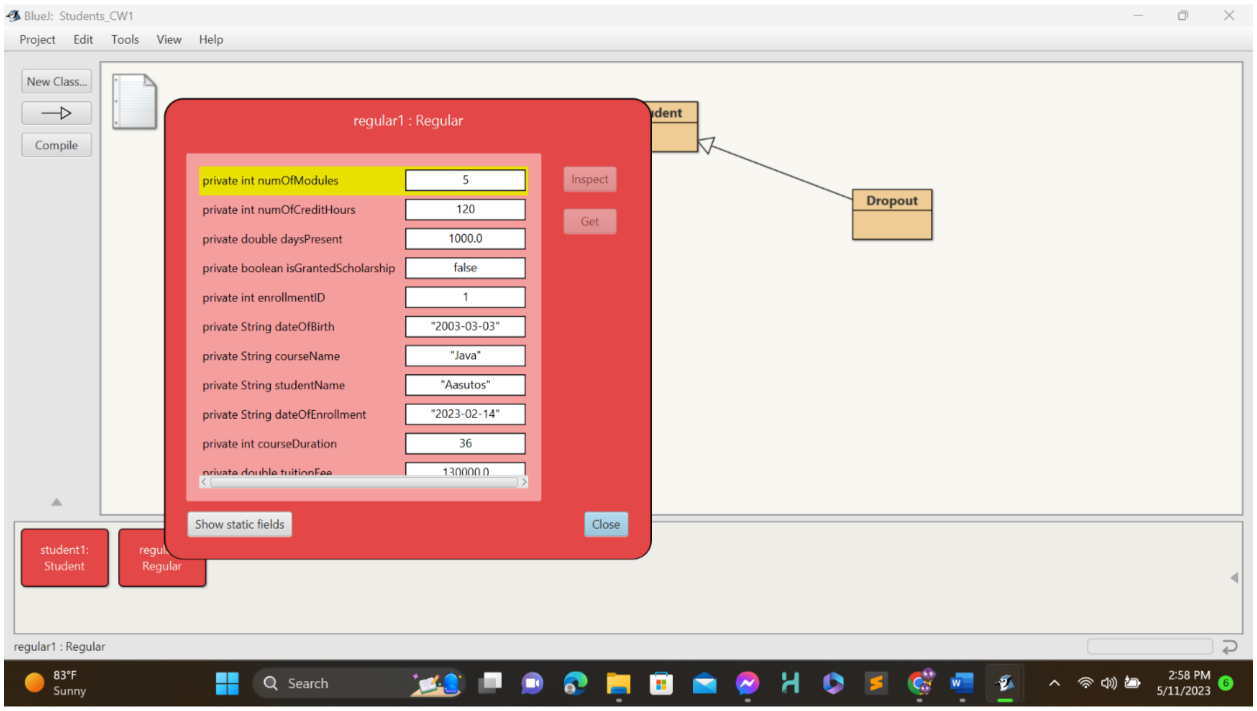

BlueJ: Create Object

Regular(String studentName, String dateOfBirth, String courseName, int courseDuration, int enrollmentID, String dateOfEnrollment, double tuitionFee, int numOfModul numOfCreditHours, double daysPresent)

Name of Instance: regular1

new Regular("Aasutos"
"2003-03-03"
"Java"
36
1
"2023-02-14"
130000
5
120
1000

Figure 26: Assigning student value to Regular present days 1000.



```
private int numOfModules                 5
private int numOfCreditHours             120
private double daysPresent               1000.0
private boolean isGrantedScholarship     false
private int enrollmentID                 1
private String dateOfBirth               "2003-03-03"
private String courseName                "Java"
private String studentName               "Aasutos"
private String dateOfEnrollment          "2023-02-14"
private int courseDuration               36
private double tuitionFee                130000.0
```

Figure 27: Inspection of regular with specific details

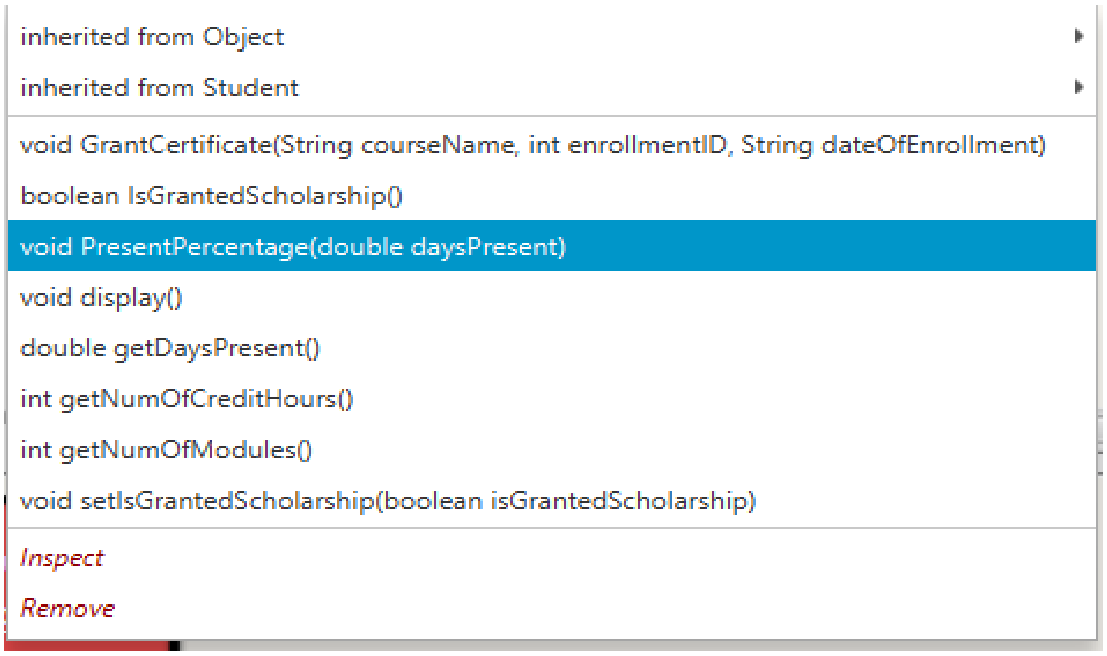
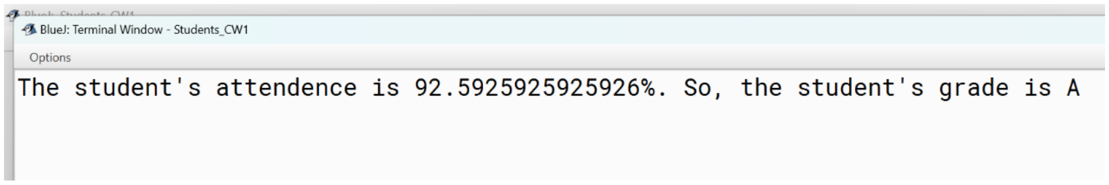

inherited from Object
inherited from Student
void GrantCertificate(String courseName, int enrollmentID, String dateOfEnrollment)
boolean IsGrantedScholarship()
void PresentPercentage(double daysPresent)
void display()
double getDaysPresent()
int getNumOfCreditHours()
int getNumOfModules()
void setIsGrantedScholarship(boolean isGrantedScholarship)
Inspect
Remove

Figure 28: Showing process to check output.



BlueJ: Terminal Window - Students_CW1

Options

```
The student's attendence is 92.5925925925926%. So, the student's grade is A
```

Figure 29: Displaying Output Of object regular1.

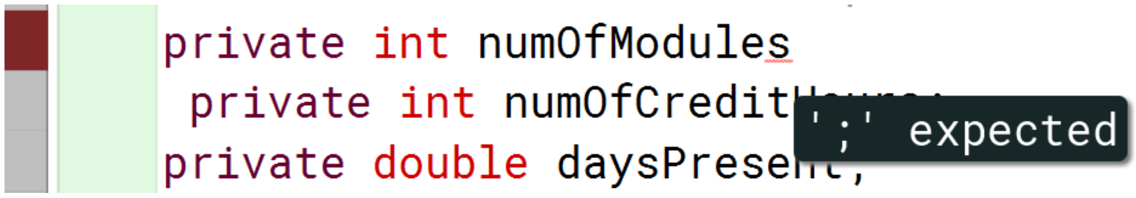
6.Error Detection and Correction

6.1 Syntax Error

In a respective programming language, there is a specific pattern or sequence of characters and keywords and if it is not written accordingly the error occurs.

Detection

As you can see below, there is an error in **“private int numOfModules”**where a semicolon is missing.



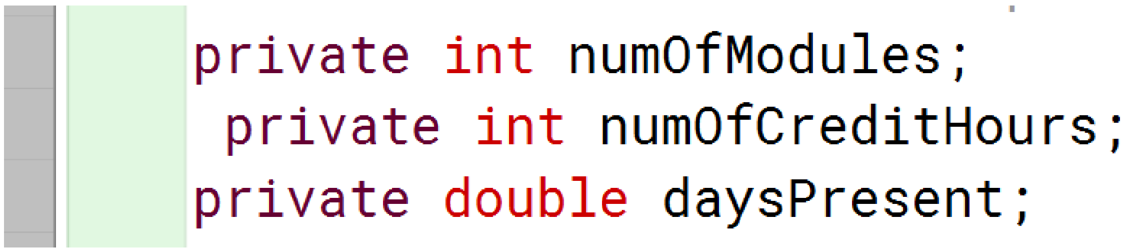
```
private int numOfModules
 private int numOfCreditHours;
private double daysPresent;
```

```
';' expected
```

Figure 30: Showing syntax error.

Correction

The above-mentioned error has been corrected y adding a semicolon" private int numOfModules:".



```
private int numOfModules;
 private int numOfCreditHours;
private double daysPresent;
```

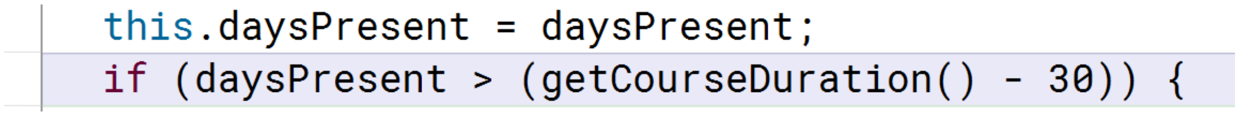
Figure 31:Showing correction of syntax error.

6.2 Logical Error

The logical error is the error on the logic, it does not show any error while compiling Yet it may produce unexpected out or may not run the program.

Detection

In the figure below, there is a logical error which is unknown to the compiler because it is the error on the logic."getCourseDuration()-30"

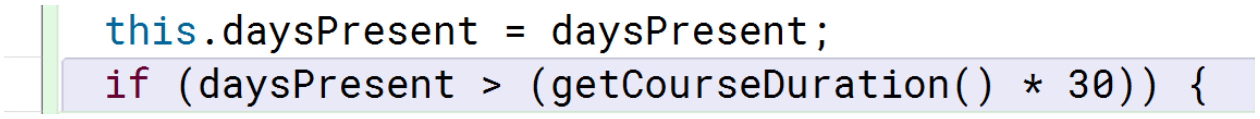


```
this.daysPresent = daysPresent;
if (daysPresent > (getCourseDuration() - 30)) {
```

Figure 32: **Showing logical error.**

Correction

The error in the code is corrected by replacing”-”with”*”.



```
this.daysPresent = daysPresent;
if (daysPresent > (getCourseDuration() * 30)) {
```

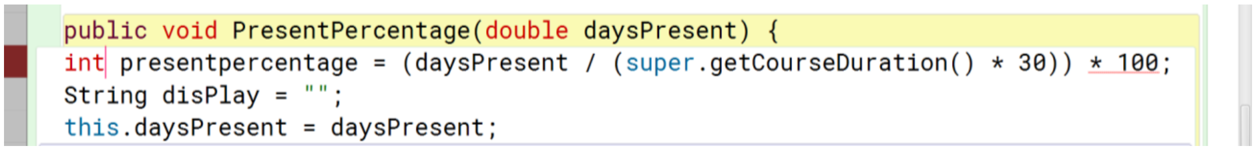
Figure 33 :Showing correction of logical error.

6.3 Semantic Error

Semantic Error is an Error where the usage of the code such as keyword, datatype is incorrect, but the syntax is correct.

Detection

As you can see in the use of int id wrong in the below diagram. The datatype used In front of persentpercentage is wrong.

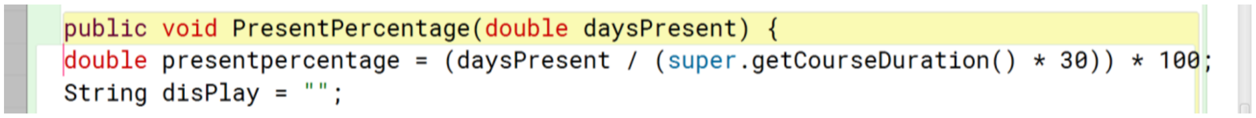


```
public void PresentPercentage(double daysPresent) {
int presentpercentage = (daysPresent / (super.getCourseDuration() * 30)) * 100;
String disPlay = "";
this.daysPresent = daysPresent;
```

Figure 34: Showing Semantic error.

Correction

The correction of the error is by using the right data type, which is Double, I used double instead of int.



```
public void PresentPercentage(double daysPresent) {
double presentpercentage = (daysPresent / (super.getCourseDuration() * 30)) * 100;
String disPlay = "";
```

Figure 35: Showing correction of Semantic error.

7 Conclusion

In this assignment, we have successfully implemented a software solution for the given scenario using Java programming language. We created three classes, with one of them being the parent class and two sub-classes. We utilized constructors for all three classes, and the use of getter and setter methods was essential. The "this" and "super" keywords were used extensively throughout the implementation.

Through this assignment, we gained a deeper understanding of object-oriented programming concepts and how to apply them to real-life scenarios. The assignment helped us to enhance our coding skills and understand the importance of testing and debugging. Overall, this was a great learning experience that has equipped us with the necessary skills to tackle more complex programming challenges in the future.

Due to the new program introduced in my subject module, I had to adjust to a new learning environment. The program demanded every little important thing which gave me a much in-depth view on JAVA programming language. The coursework was simple yet complex and it was remarkably interesting.

I’d like to thank our module lecturer Ujjwal Sir and our tutor Saroj Sir for helping me with the subject and my queries. Their guidance and support helped me overcome the challenges I faced while learning the course material…

8 Appendix

7.1 Student

```
/**
 *This class represents a student object.
 *It contains attributes for enrollment ID, date of birth, course name,
 *student name, date of enrollment, course duration in months, and tuition fee in number.
 *
 * @author Aasutosh Kumar Verma
 * @since 5-5-2023
 */

public class Student
{
    //Attributes
    private int enrollmentID;
    private String dateOfBirth;
    private String courseName;
    private String studentName;
    private String dateOfEnrollment;
    private int courseDuration; //in months
    private double tuitionFee;

    //Constructor

/**

This constructor initializes a new Student object with the given attributes.
@param studentName the name of the student
@param dateOfBirth the date of birth of the student
@param courseDuration the duration of the course in months
@param tuitionFee the tuition fee of the course
*/
    public Student(String studentName, String dateOfBirth, int courseDuration,
double tuitionFee)
     {
        //Initialized attributes with default values
       this.enrollmentID = 0;
       this.dateOfBirth = dateOfBirth;
       this.courseName = "";
       this.studentName = studentName;
       this.dateOfEnrollment = "";
       this.courseDuration = courseDuration;
       this.tuitionFee = tuitionFee;
     }

     //getters
     /**
  * Getter method for enrollment ID
  * @return the enrollment ID
  */
      public int getEnrollmentID()
    {
       return this.enrollmentID;
    }

    /**
  * Getter method for date of birth
  * @return the date of birth
  */
      public String getDateOfBirth()
    {
       return this.dateOfBirth;
    }

    /**
  * Getter method for course Of Name
  * @return the course Of Name
  */
      public String getCourseName()
    {
       return this.courseName;
    }

    /**
```

```
* Getter method for studentName
* @return the studentName
*/
    public String getStudentName()
  {
     return this.studentName;
  }

  /**
* Getter method for dateOfEnrollment
* @return the dateOfEnrollment
*/
    public String getDateOfEnrollment()
  {
     return this.dateOfEnrollment;
  }

  /**
* Getter method for courseDuration
* @return the courseDuration
*/
    public int getCourseDuration()
   {
     return this.courseDuration;
   }

   /**
* Getter method for tuitionFee
* @return the tuitionFee
*/
      public double getTuitionFee()
   {
       return this.tuitionFee;
   }
   //setters
   /**
* Setter method for enrollment ID
* @param enrollmentID the ID to set
*/
   public void setEnrollmentID(int enrollmentID)
       {
       this.enrollmentID = enrollmentID;
       }

       /**
* Setter method for date of birth
* @param dateOfBirth the date of birth to set
*/
   public void setDateOfBirth(String dateOfBirth)
      {
       this.dateOfBirth = dateOfBirth;
      }

            /**
* Setter method for course Name
* @param courseName the course Name to set
```

```
*/
  public void setCourseName(String courseName)
    {
      this.courseName = courseName;
    }

        /**
* Setter method for student Name
* @param studentName the student Name to set
*/
    public void setStudentName(String studentName)
    {
      this.studentName = studentName;
    }

        /**
* Setter method for date of Enrollment
* @param dateOfEnrollment the date Of Enrollment to set
*/
    public void setDateOfEnrollment(String dateOfEnrollment)
    {
      this.dateOfEnrollment = dateOfEnrollment;
    }

        /**
* Setter method for course Duration
* @param courseDuration the courseDuration to set
*/
```

```
        public void setCourseDuration(int courseDuration)
        {
          this.courseDuration = courseDuration;
        }

               /**
    * Setter method for tuition Fee
    * @param tuitionFee the tuitinFee to set
    */
        public void setTuitionFee(double tuitionFee)
        {
          this.tuitionFee= tuitionFee;
        }
        /**
           *This method displays the details of the student.
            *If any required detail is missing, an appropriate message is displayed.
             */
        //Display method
        public void display()
        //using if & else if condition to check wheather its is declared, missing or not
        {
            //Checking if enrolmentID is declared
           if  (enrollmentID == 0) {
          System.out.println("Enrollment ID is not declared!");
          }
          //Checking if CourseName is missing
          else if (courseName == "" || courseName == null) {
          System.out.println("Course Name is not mentioned!");
          }
          //Checking if dateOfEnrollment is missing
          else if (dateOfEnrollment.equals("") || dateOfEnrollment == null) {
          System.out.println("Date of Enrollment Value is missing!");
          }
          else {
              //Display student information details if all required details are available
          System.out.println("Enrollment ID: " + getEnrollmentID());
          System.out.println("Date of Birth: " + getDateOfBirth());
          System.out.println("Course Name: " + getCourseName());
          System.out.println("Student Name: " + getStudentName());
          System.out.println("Date Enrolled: " + getDateOfEnrollment());
          System.out.println("Course Duration: " + getCourseDuration() + " months");
          System.out.println("Tuition Fee: " + getTuitionFee());
         //for spacing or visual separation between current output and any previous
  output a blank line
          System.out.println();
          System.out.println();
        }
        }


  }
```

7.2 Regular

```
/**
```

```
 *This sub class represents a regular object.
 *It contains attributes for
 * @author Aasutosh Kumar Verma
 * @since 5-5-2023
 */
//
public class Regular extends Student
{
    //initialise instance variables
    private int numOfModules;
     private int numOfCreditHours;
    private double daysPresent;
    private boolean isGrantedScholarship;
    //constructor
     /**
     * Constructor for the Regular class
     *
     * @param studentName the name of the regular student
     * @param dateOfBirth the date of birth of the regular student
     * @param courseName the name of the course for the regular student
     * @param courseDuration the duration of the course for the regular student
     * @param enrollmentID the enrollment ID for the regular student
     * @param dateOfEnrollment the date of enrollment for the regular student
     * @param tuitionFee the tuition fee for the regular student
     * @param numOfModules the number of modules for the regular student
     * @param numOfCreditHours the number of credit hours for the regular
student
     * @param daysPresent the number of days present for the regular student
     */
    public Regular(String studentName, String dateOfBirth,  String courseName,
int courseDuration, int enrollmentID, String dateOfEnrollment, double tuitionFee,
int numOfModules, int numOfCreditHours, double daysPresent)
    {
        // initialise instance variables
        super(studentName, dateOfBirth, courseDuration, tuitionFee);
        super.setCourseName(courseName);
        super.setEnrollmentID(enrollmentID);
        super.setDateOfEnrollment(dateOfEnrollment);
        this.numOfModules = numOfModules;
        this.numOfCreditHours = numOfCreditHours;
        this.daysPresent = daysPresent;
        this.isGrantedScholarship = false;
    }

     /**
     * Getter method for the number of modules for the regular student
     *
     * @return the number of modules for the regular student
     */
    public int getNumOfModules()
   {
       return numOfModules;
    }

    /**
     * Getter method for the number of credit hours for the regular student
     *
     * @return the number of credit hours for the regular student
     */
     public int getNumOfCreditHours() {
        return numOfCreditHours;
    }

     /**
     * Getter method for the number of days present for the regular student
     *
     * @return the number of days present for the regular student
     */
    public double getDaysPresent() {
        return daysPresent;
    }

      /**
     * Getter method for whether or not the regular student is granted scholarship
     *
     * @return true if the regular student is granted scholarship, false otherwise
     */
    public boolean IsGrantedScholarship() {
        return isGrantedScholarship;
    }

    /**
     * Setter method for setting whether or not the regular student is granted
scholarship
     *
     * @param isGrantedScholarship true if the regular student is granted
scholarship, false otherwise
```

```
     */
    public void setIsGrantedScholarship(boolean isGrantedScholarship) {
        this.isGrantedScholarship = isGrantedScholarship;
    }

    /**
     * Method for calculating the present percentage of the regular student
     *
     * @param daysPresent the number of days present for the regular student
     */
    public void PresentPercentage(double daysPresent) {
    double presentpercentage = (daysPresent / (super.getCourseDuration() * 30))
* 100;
    String disPlay = "";
    this.daysPresent = daysPresent;
    if (daysPresent > (getCourseDuration() * 30)) {
    System.out.println("you can't attent more than course days");
    } else {
    if (presentpercentage >= 80 & presentpercentage <= 100) {
        setIsGrantedScholarship(true);
    disPlay="A";
    }

    else if (presentpercentage < 80 & presentpercentage >=60) {
    disPlay="B";
    }

    else if (presentpercentage < 60 & presentpercentage >=40) {
    disPlay="C";
    }


    else if (presentpercentage < 40 & presentpercentage >=20) {
    disPlay="D";
    }


    else { disPlay="E";}
    System.out.println("The student's attendence is" +" "+ presentpercentage +
"%. So, the student's grade is"+ " "+disPlay);
    }


}


/**


Grants certificate to a student and displays the information about the student's
graduation, including any granted scholarship.
@param courseName the name of the course
@param enrollmentID the enrollment ID of the student
@param dateOfEnrollment the date of enrollment of the student
*/
 public void GrantCertificate(String courseName, int enrollmentID, String
dateOfEnrollment) {
     System.out.print(super.getStudentName() + " has graduated from " +
getCourseName() + " with enrollment ID " + getEnrollmentID() + " and enrollment
date " + getDateOfEnrollment() + ".");
    if (isGrantedScholarship) {
        System.out.print(" The scholarship has been granted.");
    } else
    System.out.println(“The student has not met the attendance requirements for
the certificate.”);

    }


    /**


Displays the information about a Regular student, including the number of
modules, number of credit hours, and days present.

*/

    public void display () {

    super.display();

    System.out.println("Number of modules: " + this.numOfModules);

    System.out.println("Number of credit hours: " + this.numOfCreditHours);

    System.out.println("Days present: " + this.daysPresent);

}


}
```

7.3 Dropout

```
/**


Dropout is a subclass of Student that represents a student who has dropped out
of a course.


It contains information about the remaining modules, months attended, date of
dropout, remaining amount of tuition fee, and whether the student has paid all
their bills.

* @author Aasutosh Kumar Verma

* @since 5-5-2023
```

```
*/
public class Dropout extends Student.
{
    private int numOfRemainingModules;
    private int numOfMonthsAttended;
    private String dateOfDropout;
    private double remainingAmount;
    private boolean hasPaid;


    //constructor
    /**


Creates a new instance of the Dropout class with the given parameters.
@param studentName the name of the student
@param dateOfBirth the date of birth of the student
@param courseDuration the duration of the course in months
@param tuitionFee the tuition fee per month
@param numOfRemainingModules the number of modules the student has
remaining
@param numOfMonthsAttended the number of months the student has attended
@param dateOfDropout the date on which the student dropped out
*/
    public Dropout(String studentName, String dateOfBirth, int courseDuration,
double tuitionFee, int numOfRemainingModules, int numOfMonthsAttended,
String dateOfDropout)
    {
        super(dateOfBirth, studentName, courseDuration, tuitionFee);
        super.setCourseDuration(courseDuration);
        super.setTuitionFee(tuitionFee);
        this.numOfRemainingModules = numOfRemainingModules;
        this.numOfMonthsAttended = numOfMonthsAttended;
        this.dateOfDropout = dateOfDropout;
        super.setCourseName("");
        super.setEnrollmentID(0);
        super.setDateOfEnrollment("");
        this.remainingAmount = 0;
        this.hasPaid = false;
    }
//getters
/**

Returns the number of remaining modules the student has.
@return the number of remaining modules
*/
    public int getNumOfRemainingModules() {
        return numOfRemainingModules;
    }

    /**

Returns the number of months the student has attended the course.
@return the number of months attended
*/
    public int getNumOfMonthsAttended() {
        return this.numOfMonthsAttended;
    }
    /**

Returns the date on which the student dropped out of the course.
@return the date of dropout
*/
    public String getDateOfDropout() {
        return dateOfDropout;
    }

    /**

Returns the remaining amount of tuition fee that the student needs to pay.
@return the remaining amount of tuition fee
*/
    public double getRemainingAmount() {
        return remainingAmount;
    }

    /**

Returns whether the student has paid all their bills.
@return true if the student has paid all their bills, false otherwise
*/
    public boolean getHasPaid() {
        return hasPaid;
    }

/**
```

```
Calculates the remaining tuition fees that the student needs to pay and sets the
hasPaid field accordingly.

*/

   //This method calculates the remaining tuition fees that the student needs to
pay.

    public void billsPayble() {

        String disPlay = "";

        this.remainingAmount = (getCourseDuration() - numOfMonthsAttended) *
getTuitionFee();

        if (this.remainingAmount == 0) {

            hasPaid = true;

            disPlay = "All bills cleared by the student";

        } else {

            disPlay =  "The total amount of bills needed to be paid Rs" + "" +
this.remainingAmount;

        }

        System.out.println(disPlay);

    }

    /**


Removes the student from the system if they have paid all their bills.

Otherwise, prints an error message.

*/

        public void removeStudent() {


        if (this.hasPaid) {

            super.setDateOfBirth("");

            super.setCourseName("");

            super.setStudentName("");
            super.setDateOfEnrollment("");
            super.setCourseDuration(0);
            super.setTuitionFee(0);
            this.numOfRemainingModules = 0;
            this.numOfMonthsAttended = 0;
            this.dateOfDropout = "";
            this.remainingAmount = 0;
            super.setEnrollmentID(0);
        } else {
            System.out.println("All bills not cleared");
        }

    }
    /**
```

Overrides the display method from the student class to add additional information related to a student who has dropped out of the course.

Displays the student's name, date of birth, course name, enrollment ID, course duration, tuition fee, number of remaining modules, number of months attended, date of dropout and remaining amount to be paid.

@see Student#display()

*/

```
     @Override
    public void display () {
        super.display();
        System.out.println("Number of remaining modules: " +
numOfRemainingModules);
        System.out.println("Number of months attended: " +
numOfMonthsAttended);
        System.out.println("Date of dropout: " + dateOfDropout);
        System.out.println("Remaining amount: Rs " + remainingAmount);
    }
    }
```

8 References

Glossary, G., n.d. Gartner GUI. [Online]

Available at: https://www.gartner.com

javatpoint, n.d. semantics error. [Online]

Available at: https://www.javatpoint.com/

techvidvan, n.d. techvidvan JAVA. [Online]

Available at: https://techvidvan.com

Airth, M., n.d. javapoint. [Online]

Available https://www.javatpoint.com at:

Masai Online

Available at: - https://youtu.be/J3xHwEOKnYk
Brandon Grasley

Available at: - https://youtu.be/-0-oDrBAN1k

United Top tech

Available at: - https://youtu.be/iM6ydbg6_SQ

Tutus Funny

Available at: - https://youtu.be/KLjgsfF3KCA