Experiment Number : 07 **Date: 04-02-2026**

METHOD OVERLOADING AND METHOD OVERRIDING

Aim:

To understand and implement method overloading and method overriding.

PRE LAB EXERCISE

QUESTIONS

- ✓ What is method overloading?

  Method overloading is having multiple methods in the same class with the same name but different parameters (number or type).

- ✓ What is method overriding?

  Method overriding is when a subclass provides a new implementation of a method already defined in its parent class.

- ✓ Difference between overloading and overriding.

Overloading	Overriding
Happens in same class	Happens in parent–child classes
Same method name, different parameters	Same method name, same parameters
Compile-time concept	Runtime concept
No inheritance required	Requires inheritance

24BCS255

SELVA SHRIJITH

IN LAB EXERCISE

Objective:

To demonstrate compile-time and runtime polymorphism.

PROGRAMS:

1.Student Result System (Method Overriding)

Description:

- Base class Student has method displayResult().
- Subclasses UGStudent and PGStudent override the method to show different grading systems.

Code :

```
import java.util.Scanner;


// Base class
class Student {
    String name;


    void displayResult() {
        System.out.println("Student Result");
    }
}


// UG Student subclass
class UGStudent extends Student {
    int marks;
```

```
    UGStudent(String n, int m) {
        name = n;
        marks = m;
    }

    @Override
    void displayResult() {
        double percentage = (marks / 100.0) * 100;
        System.out.println("UG Student: " + name);
        System.out.println("Marks: " + marks);
        System.out.println("Percentage: " + percentage + "%");
    }
}

// PG Student subclass
class PGStudent extends Student {
    double gpa;

    PGStudent(String n, double g) {
        name = n;
        gpa = g;
    }

    @Override
```

```
    void displayResult() {
        System.out.println("PG Student: " + name);
        System.out.println("GPA: " + gpa + " / 10");
    }
}

// Main class
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Input for UG student
        System.out.print("Enter UG Student Name: ");
        String ugName = sc.nextLine();
        System.out.print("Enter UG Student Marks (out of 100): ");
        int ugMarks = sc.nextInt();
        sc.nextLine(); // consume newline

        // Input for PG student
        System.out.print("Enter PG Student Name: ");
        String pgName = sc.nextLine();
        System.out.print("Enter PG Student GPA (0-10): ");
        double pgGpa = sc.nextDouble();

        // Create objects
```

```
        Student s1 = new UGStudent(ugName, ugMarks);
        Student s2 = new PGStudent(pgName, pgGpa);

        System.out.println("\n--- Student Results ---");
        s1.displayResult();
        System.out.println();
        s2.displayResult();

        sc.close();
    }
}
```

OUTPUT:

Sample Input:

Enter UG Student Name: Ram

Enter UG Student Marks (out of 100): 85

Enter PG Student Name: Ravi

Enter PG Student GPA (0-10): 9.2

Output:

--- Student Results ---

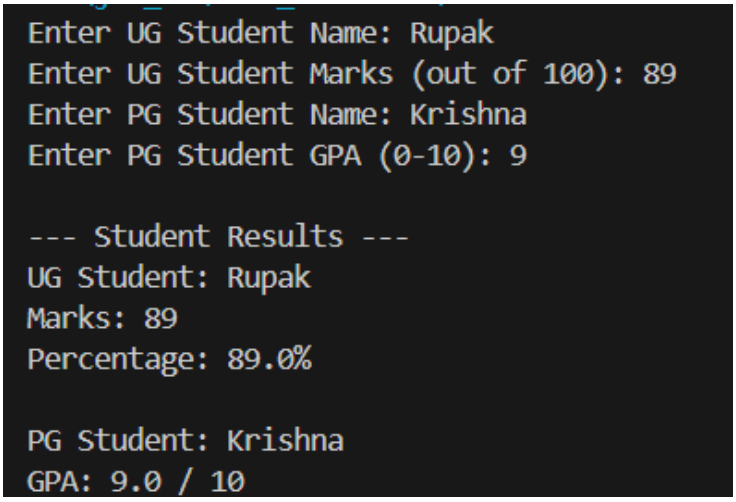
UG Student: Ram

Marks: 85

Percentage: 85.0%

PG Student: Ravi

GPA: 9.2 / 10



2. Calculator Program (Method Overloading)

Description:
Create a Calculator class with multiple add() methods to calculate:

- Addition of 2 integers
- Addition of 3 integers
- Addition of 2 double numbers

Code:

```
import java.util.Scanner;

class Calculator {
    int add(int a, int b) {
        return a + b;
    }

    int add(int a, int b, int c) {
        return a + b + c;
    }
```

```
    double add(double a, double b) {
        return a + b;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Calculator calc = new Calculator();

        System.out.print("Enter two integers: ");
        int x = sc.nextInt();
        int y = sc.nextInt();
        System.out.println("Sum of two integers: " + calc.add(x, y));

        System.out.print("Enter three integers: ");
        int p = sc.nextInt();
        int q = sc.nextInt();
        int r = sc.nextInt();
        System.out.println("Sum of three integers: " + calc.add(p, q, r));

        System.out.print("Enter two decimal numbers: ");
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        System.out.println("Sum of two doubles: " + calc.add(a, b));
```

24BCS255

SELVA SHRIJITH

```
        sc.close();
    }
}
```

Output:

Sample Input:

Enter two integers: 10 20

Enter three integers: 5 10 15

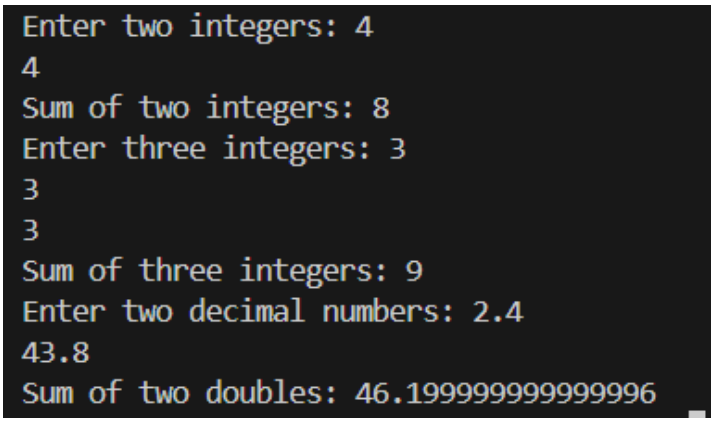
Enter two decimal numbers: 2.5 3.5

Output:

Sum of two integers: 30

Sum of three integers: 30

Sum of two doubles: 6.0



POST LAB EXERCISE

- ✓ Is return type important in method overloading and method overriding?

24BCS255

SELVA SHRIJITH

In overloading, return type alone is not enough to distinguish methods.

In overriding, the return type must be the same (or compatible/covariant).

✓ Can you overload a method by changing only the return type?

No. The parameter list must change. Return type alone cannot overload a method.

✓ Can static methods be overridden? Can they be overloaded?

Static methods cannot be truly overridden (they are hidden, not overridden).

Yes, static methods can be overloaded.

✓ Can a method be overridden if the parameter list is different?

No. If parameters change, it becomes overloading, not overriding.

Result:

Thus the method overloading and overriding concepts were implemented and executed successfully.

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	

24BCS255

SELVA SHRIJITH

Total	**30**	
	Faculty Signature	

24BCS255

SELVA SHRIJITH