Ex 1	The use of classes, objects, constructors, method overloading, inheritance, super, and abstract classes

Aim

To write a Java program that demonstrates the use of classes, objects, constructors, method overloading, inheritance, super, and abstract classes.

Definitions

Classes

In Java, a class serves as a blueprint or a template for creating objects. It defines the structure and behavior that objects of that class will possess. Think of it like a design for a house: the design specifies the number of rooms, windows, and doors, but it's not the actual house itself.

Objects

In Java, an object is a fundamental concept in object-oriented programming (OOP). It represents a real-world entity or a concept within your program.

Constructors

In Java, a constructor is a special type of method used to initialize objects of a class. It is automatically invoked when an object is created using the new keyword.

method overloading

Method overloading in Java is a feature that allows a class to have multiple methods with the same name but different parameter lists. It is a form of compile-time polymorphism, where the compiler determines which specific method to execute based on the method signature (the method name and the number, type, and order of its parameters).

Inheritance

Inheritance in Java is a fundamental concept of Object-Oriented Programming (OOP) that allows a class to inherit properties and behaviors (fields and methods) from another class. This establishes an "IS-A" relationship, where the inheriting class (subclass or child class) is a specialized version of the inherited class (superclass or parent class).

Super

In Java, the super keyword is a reference variable used within a subclass to refer to the immediate parent class (superclass) object. It is fundamentally tied to the concept of inheritance.

abstract classes

In Java, an abstract class is a class that cannot be instantiated directly, meaning you cannot create objects of an abstract class. It is declared using the abstract keyword in its class definition.

Procedure

open NetBeans IDE.

To create a Project go to File Menu → choose New Project → choose Java from Categories →choose Java Application from Projects →click next →specify the project name as EX1 →click Finish.

Right click on package ex1 → choose New → select Java Class → specify the class name as Person →click Finish.

Type the following codes in EX1.java:

EX1.java

```
package ex1;


/**
 *
 * @author PGLAB
 */
public class EX1 {
    public static void main(String[] args) {
        // Creating object using constructor
        Student student1 = new Student("Alice", 20, "S123", "Computer Science");
        Student student2 = new Student("Bob", 22, "S456");  // Using overloaded constructor


        // Calling methods
        student1.displayInfo();
        student1.study();
        student1.study("Data Structures");


        System.out.println();


        student2.displayInfo();
        student2.study();
    }
```

}

Type the following codes in Person.java:

Person.java

```
package ex1;


/**
 *
 * @author PGLAB
 */
// Abstract class
abstract class Person {
    String name;
    int age;

    // Constructor
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Abstract method
    abstract void displayInfo();
}

// Subclass inheriting from abstract class
class Student extends Person {
    String studentId;
    String course;

    // Constructor using super to call parent constructor
```

```
    public Student(String name, int age, String studentId, String course) {
        super(name, age);  // Call to superclass constructor
        this.studentId = studentId;
        this.course = course;
    }

    // Overloaded constructor (Method Overloading)
    public Student(String name, int age, String studentId) {
        this(name, age, studentId, "Undeclared");
    }

    // Overloaded method (Method Overloading)
    public void study() {
        System.out.println(name + " is studying " + course);
    }

    public void study(String subject) {
        System.out.println(name + " is studying " + subject);
    }

    // Implementation of abstract method
    @Override
    void displayInfo() {
        System.out.println("Student Info:");
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Student ID: " + studentId);
        System.out.println("Course: " + course);
    }
}
```

Right click on EX1.java file → choose Run File. You can see the following result in the output window.

Output

```
run:
Student Info:
Name: Alice
Age: 20
Student ID: S123
Course: Computer Science
Alice is studying Computer Science
Alice is studying Data Structures

Student Info:
Name: Bob
Age: 22
Student ID: S456
Course: Undeclared
Bob is studying Undeclared
BUILD SUCCESSFUL (total time: 0 seconds)
```

Result

Thus, a Java program that demonstrates the use of classes, objects, constructors, method overloading, inheritance, super, and abstract classes has been written and executed successfully.