Practical-4 Method Overloading and Introduction to Inheritance

• Method Overloading

If a class has multiple methods having same name but different in parameters, it is known as Method Overloading.

It is called Static Polymorphism

```
public class OverloadDemo {
    // Method to add two integers
     private int add(int a, int b) {
        return a + b;
     }
     // Overloaded method to add two double values
     private double add(double a, double b) {
        return a + b;
     }
     // Overloaded method to add three integers
     private int add(int a, int b, int c) {
        return a + b + c;
     }
     public static void main(String[] args) {
        OverloadDemo example = new OverloadDemo();

        // Using the first add method
        System.out.println("Sum of two integers: " + example.add(10, 20));

        // Using the second add method
        System.out.println("Sum of two doubles: " + example.add(10.5, 20.5));

        // Using the third add method
        System.out.println("Sum of three integers: " + example.add(10, 20, 30));
     }
}
```

• Java Inheritance

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object.

The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class

Inheritance represents the **IS-A relationship** which is also known as a *parent-child* relationship.

- Terms used in Inheritance

  - **Class:**
  - **Sub Class/Child Class:** Subclass is a class which inherits the other class. It is also called a derived class, extended class, or child class.
  - **Super Class/Parent Class:** Superclass is the class from where a subclass inherits the features. It is also called a base class or a parent class.

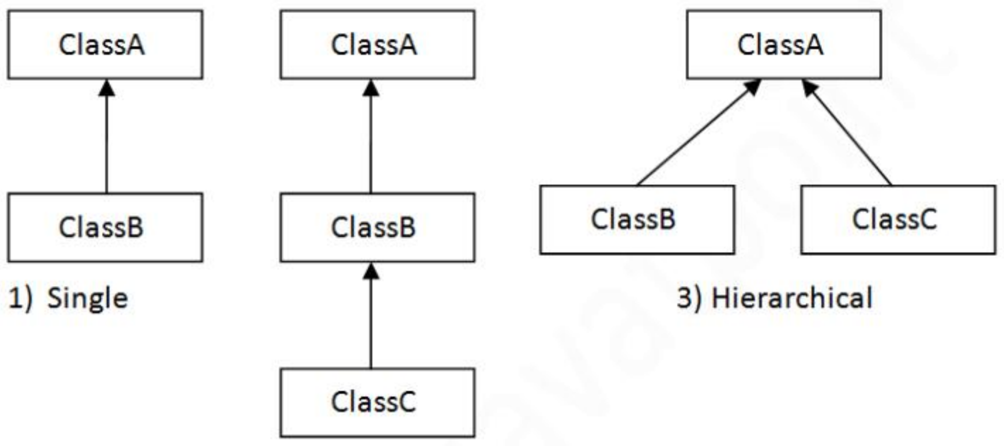
- The syntax of Java Inheritance

```
class Subclass-name extends Superclass-name
{
//methods and fields
}
```

The **extends keyword** indicates that you are making a new class that derives from an existing class.

```
class Employee
{
    float salary=40000;
}
class Programmer extends Employee
{
     int bonus=10000;
    public static void main(String args[])
      {
            Programmer p=new Programmer();
        System.out.println("Programmer salary is:"+p.salary);
        System.out.println("Bonus of Programmer is:"+p.bonus);
    }
}
```

- Types of inheritance in java



Note: Multiple inheritance is not supported in Java through class.

- Single inheritance

TestInheritance.java

```
class Animal
{
void eat()
{
        System.out.println("eating...");
   }
}   // end of animal class

class Dog extends Animal
{
        void bark()

        { System.out.println("barking...");
        }
}

class TestInheritance{
     public static void main(String args[]){
         Dog d=new Dog();
         d.bark();
         d.eat();
       }
}
```

- Multilevel inheritance Example

```
 class Vehicle {
  int numOfWheels=4;

    void printwheel() {
    System.out.println("From Vehicle class  no of wheels \n"+numOfWheels);
  }
    void dispdetail()
   { System.out.println("From Vehicle class  no of wheels  \n"+numOfWheels);
   // System.out.println("\n Model :" + model + "Color :" + color);
   }

   }

 class Car extends Vehicle{
     private String model;
    private String color;

 void setmodel(String m){
       model =m;
    }

void setcolor(String c){
       color=c;
    }

void dispdetail() // method overridding
   {
     System.out.println("\n Model :" + model + "Color :" + color);
   }
 } // END OF CLASS CAR

   class SUV extends Car {
     private int noofseats;
       int getseats(){
           return noofseats;
     }
     void setseat(int n){
           noofseats=n;
         }
    }
```

```
public class MultilevelInheritance{
    public static void main(String arg[]){
      SUV creta = new SUV();
      creta.setseat(5);
      creta.setmodel("Hudai creta SUV");
      creta.setcolor("White");
      creta.dispdetail(); // called car class method
      creta.printwheel(); // call vehical class method
      System.out.println("No of seat" + creta.getseats());
    }
  }
```

Method Overriding

method overriding is a feature that allows a subclass to provide a specific implementation for a method that is already defined in its parent class.

Exercise

1	Create a class Bank having properties : account no, balance, type of account . create constructor to initialize the value. Implement withdraw() and deposit() method which withdraw and deposit given amount. Take necessary validation during withdraw the amount. Minimum balance should be 5000. Create three object of bank and perform withdraw and deposit on object
2	Create a class Operations **int addition(int int)** : return two numbers. **String addition(String , String)** : concat the string and return concated string **int subtract(int , int)** : return substraction of two numbers **String Subtract(String, int , int)** : delete the number of characters from given string.(integer values are starting and ending index) int maximum (int,int) : return the maximum from two numbers String multiply(String, String) : return the greater string from two strings. (Hint use compareTo() method) Create an object of Operation class and perform all methods.
3	Create java program for following Class Item Property : itemname,price Method . setValues(String,int) : set the values : getValues() : display item information

	Class Order Property : noofunit, Method : setUnit(int) , int getUnit(), calAmount() : calculate the total amount(noofunit*price) Create three object of order and display all details.
4	Create java class Person having name , and age as property. Setname(String) : set the name Setage(int) : set the age String getName() : return name Int getAge() : return age Class Teacher inherits Person Property : salary,experience, Methods : setSalary(int), setExperiance(int), int getSalary(), int getExperiance() , Class Student inherits Person Property : coursename, marks Methods : setCoursename(String) , setMarks(int) , String getCoursename() , int getMarks() Create java program to create Teacher and Students object and display all the detail.