1. **;create multilevel inheritance for**

**//Vehicle**

**//Four\_wheeler**

**//Petrol\_Four\_Wheeler**

**//FiveSeater\_Petrol\_Four\_Wheeler**

**//Baleno\_FiveSeater\_Petrol\_Four\_Wheeler**

**package** day\_4;

**class** Vehicle {

**void** type(String type) {

System.***out***.println("Type of Vehicle"+type);

}

}

**class** FourWheeler **extends** Vehicle {

**void** wheels(**int** wheels) {

System.***out***.println("Number of Wheels"+wheels);

}

}

**class** PetrolFourWheeler **extends** FourWheeler {

**void** fuelType(String fuelType) {

System.***out***.println("Fuel Type"+fuelType);

}

}

**class** FiveSeaterPetrolFourWheeler **extends** PetrolFourWheeler {

**void** seatingCapacity(**int** capacity) {

System.***out***.println("Seating Capacity"+capacity);

}

}

**class** BalenoFiveSeaterPetrolFourWheeler **extends** FiveSeaterPetrolFourWheeler {

**void** modelName(String modelName) {

System.***out***.println("Model Name"+modelName);

}

}

**public** **class** Day4\_Assign {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

BalenoFiveSeaterPetrolFourWheeler baleno = **new** BalenoFiveSeaterPetrolFourWheeler();

baleno.type("Car");

baleno.wheels(4);

baleno.fuelType("Petrol");

baleno.seatingCapacity(5);

baleno.modelName("Maruti Suzuki Baleno");

}

}

Demonstrate the use of the super keyword

2. super keyword in Java is used to refer to the parent class (superclass) from a subclass. It can be used in

1. accessing parent class members
2. to Invoke parent class constructors
3. Method Overriding

When overriding a method from the parent class, can use super to call the parent class method from the subclass method.

Advantages

1. Code Reusability: super helps to reuse code from the parent class.

2. Method Overriding: super allows to call the parent class method from the subclass method

**Create Hospital super class and access this class inside the patient child class and access properties from Hospital class**

3. **package** day\_4;

**class** Hospital {

String hospN;

String loc;

Hospital(String hospN, String loc) {

**this**.hospN=hospN;

**this**.loc=loc;

}

**void** displayHospitalInfo() {

System.***out***.println("Hospital Name:"+hospN);

System.***out***.println("Location:"+loc);

}

}

**class** Patient **extends** Hospital {

String patientN;

**int** patientA;

Patient(String hospitalName, String location, String patientN, **int** patientA) {

**super**(hospitalName, location);

**this**.patientN=patientN;

**this**.patientA=patientA;

}

**void** displayPatientInfo() {

System.***out***.println("Patient Name:"+patientN);

System.***out***.println("Patient Age:"+patientA);

}

**void** displayAllInfo() {

displayHospitalInfo();

displayPatientInfo();

}

}

**public** **class** Da4\_Assi1 {

**public** **static** **void** main(String[] args) {

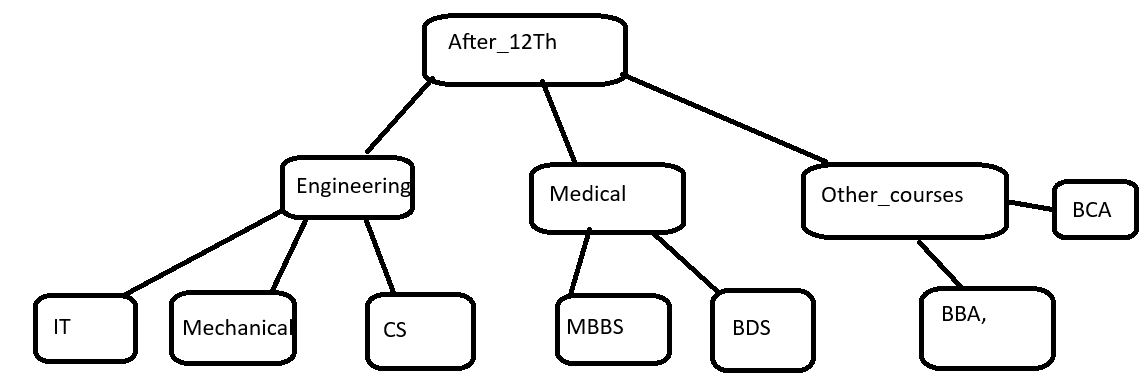
// **TODO** Auto-generated method stub

Patient patient = **new** Patient("Apollo Hospital", "Chennai", "John Doe", 30);

patient.displayAllInfo();

}

}



4. **package** day\_4;

**class** After\_12Th {

String courseType;

After\_12Th(String courseType) {

**this**.courseType = courseType;

}

**void** displayCourseType() {

System.***out***.println("Course Type: " + courseType);

}

}

**class** Engineering **extends** After\_12Th {

String engineeringBranch;

Engineering(String courseType, String engineeringBranch) {

**super**(courseType);

**this**.engineeringBranch = engineeringBranch;

}

**void** displayEngineeringBranch() {

System.***out***.println("Engineering Branch: " + engineeringBranch);

}

}

**class** IT **extends** Engineering {

IT(String courseType, String engineeringBranch) {

**super**(courseType, engineeringBranch);

}

**void** displayITInfo() {

displayCourseType();

displayEngineeringBranch();

System.***out***.println("Specialization: Information Technology");

}

}

**class** Mechanical **extends** Engineering {

Mechanical(String courseType, String engineeringBranch) {

**super**(courseType, engineeringBranch);

}

**void** displayMechanicalInfo() {

displayCourseType();

displayEngineeringBranch();

System.***out***.println("Specialization: Mechanical Engineering");

}

}

**class** CS **extends** Engineering {

CS(String courseType, String engineeringBranch) {

**super**(courseType, engineeringBranch);

}

**void** displayCSInfo() {

displayCourseType();

displayEngineeringBranch();

System.***out***.println("Specialization: Computer Science");

}

}

**class** Medical **extends** After\_12Th {

String medicalSpecialization;

Medical(String courseType, String medicalSpecialization) {

**super**(courseType);

**this**.medicalSpecialization = medicalSpecialization;

}

**void** displayMedicalSpecialization() {

System.***out***.println("Medical Specialization: " + medicalSpecialization);

}

}

**class** MBBS **extends** Medical {

MBBS(String courseType, String medicalSpecialization) {

**super**(courseType, medicalSpecialization);

}

**void** displayMBBSInfo() {

displayCourseType();

displayMedicalSpecialization();

System.***out***.println("Degree: MBBS");

}

}

**class** BDS **extends** Medical {

BDS(String courseType, String medicalSpecialization) {

**super**(courseType, medicalSpecialization);

}

**void** displayBDSInfo() {

displayCourseType();

displayMedicalSpecialization();

System.***out***.println("Degree: BDS");

}

}

**class** Other\_courses **extends** After\_12Th {

String otherCourse;

Other\_courses(String courseType, String otherCourse) {

**super**(courseType);

**this**.otherCourse = otherCourse;

}

**void** displayOtherCourse() {

System.***out***.println("Other Course: " + otherCourse);

}

}

**class** BBA **extends** Other\_courses {

BBA(String courseType, String otherCourse) {

**super**(courseType, otherCourse);

}

**void** displayBBAInfo() {

displayCourseType();

displayOtherCourse();

System.***out***.println("Degree: BBA");

}

}

**class** BCA **extends** Other\_courses {

BCA(String courseType, String otherCourse) {

**super**(courseType, otherCourse);

}

**void** displayBCAInfo() {

displayCourseType();

displayOtherCourse();

System.***out***.println("Degree: BCA");

}

}

**public** **class** Day4\_3Q{

**public** **static** **void** main(String[] args) {

IT it = **new** IT("Engineering", "IT");

it.displayITInfo();

MBBS mbbs = **new** MBBS("Medical", "General Medicine");

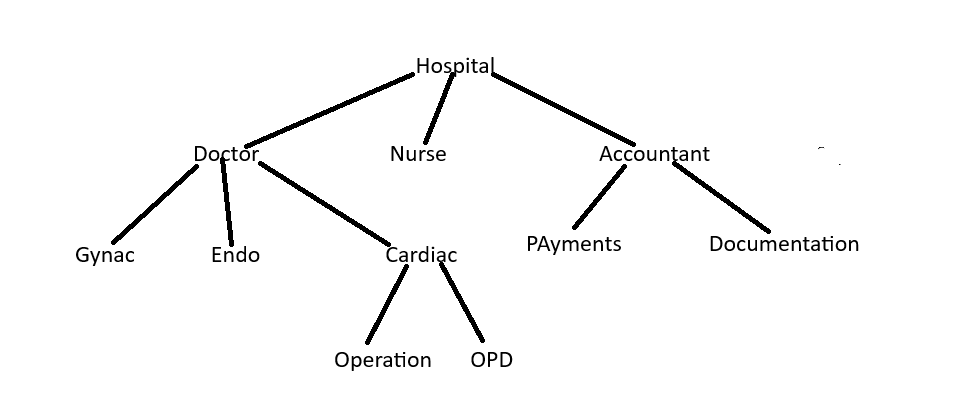
mbbs.displayMBBSInfo();

BBA bba = **new** BBA("Other Courses", "Business Administration");

bba.displayBBAInfo();

}

}



5. **package** day\_4;

**class** Hospital\_1 {

**void** hospitalName(String name) {

System.***out***.println("Hospital Name:" + name);

}

}

**class** Doctor **extends** Hospital\_1 {

**void** doctorName(String name) {

System.***out***.println("Doctor Name:" + name);

}

}

**class** Gynac **extends** Doctor {

**void** gynacServices(String services) {

System.***out***.println("Gynac Services:" + services);

}

}

**class** Endo **extends** Doctor {

**void** endoServices(String services) {

System.***out***.println("Endo Services:" + services);

}

}

**class** Cardiac **extends** Doctor {

Cardiac() {}

}

**class** CardiacOperation **extends** Cardiac {

**void** operation(String op) {

System.***out***.println("Operation:" + op);

}

}

**class** CardiacOPD **extends** Cardiac {

**void** opd(String opd) {

System.***out***.println("Opd:" + opd);

}

}

**class** Nurse **extends** Hospital\_1 {

**void** nurseName(String name) {

System.***out***.println("Nurse Name:" + name);

}

}

**class** Accountant **extends** Hospital\_1 {

**void** accName(String name) {

System.***out***.println("Accountant Name:" + name);

}

}

**class** Payments **extends** Accountant {

**void** makePayments(String pay) {

System.***out***.println("Payments:" + pay);

}

}

**class** Documentation **extends** Accountant {

**void** maintainDocumentation(String doc) {

System.***out***.println("Documentation:" + doc);

}

}

**public** **class** Day4\_4Q {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Gynac gynac = **new** Gynac();

gynac.doctorName("Dr.Sneha");

gynac.gynacServices("Pregnancy Care");

System.***out***.println();

CardiacOperation cardiacOperation = **new** CardiacOperation();

cardiacOperation.doctorName("Dr.Roopa");

cardiacOperation.operation("Bypass Surgery");

System.***out***.println();

CardiacOPD cardiacOPD = **new** CardiacOPD();

cardiacOPD.doctorName("Dr.Roopa");

cardiacOPD.opd("Cardiac OPD");

System.***out***.println();

Nurse nurse = **new** Nurse();

nurse.nurseName("Nurse Jane");

System.***out***.println();

Payments payments = **new** Payments();

payments.accName("Accountant Seema");

payments.makePayments("Salary Payments");

System.***out***.println();

Documentation documentation = **new** Documentation();

documentation.accName("Accountant Seema");

documentation.maintainDocumentation("Patient Records");

}

}

Polymorphism

1. Create a class Calculator with the following overloaded add()

1.add(int a, int b)

2.add(int a, int b, int c)

3.add(double a, double b)

1. **package** day\_4;

**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** Poly\_1 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Calculator calculator = **new** Calculator();

**int** result1 = calculator.add(10, 20);

System.***out***.println("Result 1: " + result1);

**int** result2 = calculator.add(10, 20, 30);

System.***out***.println("Result 2: " + result2);

**double** result3 = calculator.add(10.5, 20.7);

System.***out***.println("Result 3: " + result3);

}

}

1. Create a base class Shape with a method area() that prints a message. Then create two subclasses Circleàoverride area() to calculator and print area of circle Rectangleà override area() to calculate and print area of a rectangle

2. **package** day\_4;

**interface** Shape1 {

**void** draw();

**double** area();

}

**class** Circle **implements** Shape1 {

**private** **double** radius;

**public** Circle(**double** radius) {

**this**.radius = radius;

}

**public** **void** draw() {

System.***out***.println("Drawing a circle with radius: " + radius);

}

**public** **double** area() {

**return** Math.***PI*** \* radius \* radius;

}

}

**class** Rectangle **implements** Shape1 {

**private** **double** length;

**private** **double** width;

**public** Rectangle(**double** length, **double** width) {

**this**.length = length;

**this**.width = width;

}

**public** **void** draw() {

System.***out***.println("Drawing a rectangle with length: " + length + " and width: " + width);

}

**public** **double** area() {

**return** length \* width;

}

}

**public** **class** Poly\_2 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Shape1 c = **new** Circle(5.0);

Shape1 r = **new** Rectangle(4.0, 6.0);

c.draw();

System.***out***.println("Area of circle: " + c.area());

r.draw();

System.***out***.println("Area of rectangle: " + r.area());

}

}

1. Create a Bank class with a method getInterestRate() create subclasses: SBIàreturn 6.7% ICICIàreturn 7.0% HDFCàreturn 7.5%

3. **package** day\_4;

**class** Bank1 {

**public** **double** getInterestRate() {

**return** 0;

}

}

**class** SBI **extends** Bank1 {

**public** **double** getInterestRate() {

**return** 6.7;

}

}

**class** ICICI **extends** Bank1 {

**public** **double** getInterestRate() {

**return** 7.0;

}

}

**class** HDFC **extends** Bank1 {

**public** **double** getInterestRate() {

**return** 7.5;

}

}

**public** **class** Poly3\_q {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Bank1 sbi = **new** SBI();

Bank1 icici = **new** ICICI();

Bank1 hdfc = **new** HDFC();

System.***out***.println("SBI Interest Rate: " + sbi.getInterestRate() + "%");

System.***out***.println("ICICI Interest Rate: " + icici.getInterestRate() + "%");

System.***out***.println("HDFC Interest Rate: " + hdfc.getInterestRate() + "%");

}

}

1. Runtime Polymorphism with constructor Chaining create a class vehicle with a constructor that prints “Vehicle Created”

Create a subclass Bike that override a method and uses super() in constructor

4. **package** day\_4;

**class** Vehicle2 {

Vehicle2() {

System.***out***.println("Vehicle Created");

}

**void** display() {

System.***out***.println("This is a Vehicle");

}

}

**class** Bike2 **extends** Vehicle2 {

Bike2() {

**super**();

System.***out***.println("Bike Created");

}

**void** display() {

System.***out***.println("This is a Bike");

}

}

**public** **class** Poly\_5 {

**public** **static** **void** main(String[] args) {

Vehicle2 vehicle = **new** Vehicle2();

vehicle.display();

System.***out***.println();

Vehicle2 bike = **new** Bike2();

bike.display();

}

}

**Combined questions**

**Combined question**

**Create an abstract class SmartDevice with methods like turnOn(), turnOff(), and performFunction().  
Create child classes:**

* **SmartPhone: performs calling and browsing.**
* **SmartWatch: tracks fitness and time.**
* **SmartSpeaker: plays music and responds to voice commands.**
* **Write code to store all objects in an array and use polymorphism to invoke their performFunction().**

1. **package** day\_4;

//combined question1

**abstract** **class** SmartDevice {

**abstract** **void** turnOn();

**abstract** **void** turnOff();

**abstract** **void** performFunction();

}

**class** SmartPhone **extends** SmartDevice {

**void** turnOn() {

System.***out***.println("SmartPhone is turned on");

}

**void** turnOff() {

System.***out***.println("SmartPhone is turned off");

}

**void** performFunction() {

System.***out***.println("SmartPhone is making a call and browsing the internet");

}

**void** makeCall() {

System.***out***.println("SmartPhone is calling");

}

**void** browseInternet() {

System.***out***.println("SmartPhone is browsing the internet...");

}

}

**class** SmartWatch **extends** SmartDevice {

**void** turnOn() {

System.***out***.println("SmartWatch is turned on");

}

**void** turnOff() {

System.***out***.println("SmartWatch is turned off");

}

**void** performFunction() {

System.***out***.println("SmartWatch is tracking fitness and displaying time");

}

**void** trackFitness() {

System.***out***.println("SmartWatch is tracking fitness");

}

**void** displayTime() {

System.***out***.println("SmartWatch is displaying time");

}

}

**class** SmartSpeaker **extends** SmartDevice {

**void** turnOn() {

System.***out***.println("SmartSpeaker is turned on");

}

**void** turnOff() {

System.***out***.println("SmartSpeaker is turned off");

}

**void** performFunction() {

System.***out***.println("SmartSpeaker is playing music and responding to voice commands");

}

**void** playMusic() {

System.***out***.println("SmartSpeaker is playing music");

}

**void** respondToVoiceCommands() {

System.***out***.println("SmartSpeaker is responding to voice commands");

}

}

**public** **class** Poly\_4Q {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

SmartDevice smartPhone = **new** SmartPhone();

SmartDevice smartWatch = **new** SmartWatch();

SmartDevice smartSpeaker = **new** SmartSpeaker();

System.***out***.println("SmartPhone:");

smartPhone.turnOn();

smartPhone.performFunction();

smartPhone.turnOff();

System.***out***.println("\nSmartWatch:");

smartWatch.turnOn();

smartWatch.performFunction();

smartWatch.turnOff();

System.***out***.println("\nSmartSpeaker:");

smartSpeaker.turnOn();

smartSpeaker.performFunction();

smartSpeaker.turnOff();

}

}

Design an interface Bank with methods deposit(), withdraw(), and getBalance().  
Implement this in SavingsAccount and CurrentAccount classes.

* Use inheritance to create a base Account class.
* Demonstrate method overriding with customized logic for withdrawal (e.g., minimum balance in SavingsAccount).

2. **package** day\_4;

**interface** Bank {

**void** deposit(**double** amount);

**void** withdraw(**double** amount);

**double** getBalance();

}

**abstract** **class** Account **implements** Bank {

**protected** **double** bal;

**public** Account(**double** iniBalance) {

**this**.bal=iniBalance;

}

**public** **void** deposit(**double** amount) {

bal+=amount;

System.***out***.println("Deposited: "+amount);

}

**public** **double** getBalance() {

**return** bal;

}

}

**class** SavingsAccount **extends** Account {

**private** **static** **final** **double** ***MIN\_BALANCE*** =1000;

**public** SavingsAccount(**double** initialBalance) {

**super**(initialBalance);

}

@Override

**public** **void** withdraw(**double** amount) {

**if** (bal-amount>=***MIN\_BALANCE***) {

bal-= amount;

System.***out***.println("Withdrawn:"+amount);

} **else** {

System.***out***.println("Insufficient balance"+***MIN\_BALANCE***);

}

}

}

**class** CurrentAccount **extends** Account {

**private** **static** **final** **double** ***OVERDRAFT\_LIMIT*** = 5000;

**public** CurrentAccount(**double** initialBalance) {

**super**(initialBalance);

}

**public** **void** withdraw(**double** amount) {

**if** (bal-amount >= -***OVERDRAFT\_LIMIT***) {

bal-=amount;

System.***out***.println("Withdrawn: "+amount);

} **else** {

System.***out***.println("Overdraft limit exceeded"+***OVERDRAFT\_LIMIT***);

}

}

}

**public** **class** Combined\_Q1 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

SavingsAccount sA= **new** SavingsAccount(1500);

sA.deposit(500);

sA.withdraw(1000);

sA.withdraw(1000);

System.***out***.println("Savings Account Balance: " + sA.getBalance());

System.***out***.println();

CurrentAccount cA= **new** CurrentAccount(1000);

cA.deposit(500);

cA.withdraw(2000);

cA.withdraw(5000);

System.***out***.println("Current Account Balance: " + cA.getBalance());

}

}

Create a base class Vehicle with method start().  
Derive Car, Bike, and Truck from it and override the start() method.

* Create a static method that accepts Vehicle type and calls start().
* Pass different vehicle objects to test polymorphism.

3. **package** day\_4;

**class** Vehicle3 {

**void** start() {

System.***out***.println("Vehicle started");

}

}

**class** Car **extends** Vehicle3 {

@Override

**void** start() {

System.***out***.println("Car started");

}

}

**class** Bike **extends** Vehicle3 {

**void** start() {

System.***out***.println("Bike started");

}

}

**class** Truck **extends** Vehicle3 {

**void** start() {

System.***out***.println("Truck started");

}

}

**class** VehicleUtil {

**static** **void** startVehicle(Vehicle3 vehicle) {

vehicle.start();

}

}

**public** **class** Combined\_Q3 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Vehicle3 car = **new** Car();

Vehicle3 bike = **new** Bike();

Vehicle3 truck = **new** Truck();

VehicleUtil.*startVehicle*(car);

VehicleUtil.*startVehicle*(bike);

VehicleUtil.*startVehicle*(truck);

}

}

Design an abstract class Person with fields like name, age, and abstract method getRoleInfo().  
Create subclasses:

* Student: has course and roll number.
* Professor: has subject and salary.
* TeachingAssistant: extends Student and implements getRoleInfo() in a hybrid way.
* Create and print info for all roles using overridden getRoleInfo().

4. **package** day\_4;

**abstract** **class** Person {

**protected** String name;

**protected** **int** age;

**public** Person(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

**abstract** **void** getRoleInfo();

}

**class** Student **extends** Person {

**private** String course;

**private** **int** rollNumber;

**public** Student(String name, **int** age, String course, **int** rollNumber) {

**super**(name, age);

**this**.course = course;

**this**.rollNumber = rollNumber;

}

**void** getRoleInfo() {

System.***out***.println("Name"+name);

System.***out***.println("Age"+age);

System.***out***.println("Course" + course);

System.***out***.println("Roll Number"+rollNumber);

System.***out***.println("Role Student");

}

}

**class** Professor **extends** Person {

**private** String subject;

**private** **double** salary;

**public** Professor(String name, **int** age, String subject, **double** salary) {

**super**(name, age);

**this**.subject = subject;

**this**.salary = salary;

}

**void** getRoleInfo() {

System.***out***.println("Name"+name);

System.***out***.println("Age"+age);

System.***out***.println("Subject"+subject);

System.***out***.println("Salary"+salary);

System.***out***.println("Role Professor");

}

}

**class** TeachingAssistant **extends** Student {

**private** String subject;

**public** TeachingAssistant(String name, **int** age, String course, **int** rollNumber, String subject) {

**super**(name, age, course, rollNumber);

**this**.subject = subject;

}

@Override

**void** getRoleInfo() {

**super**.getRoleInfo();

System.***out***.println("Subject: " + subject);

System.***out***.println("Role: Teaching Assistant");

}

}

**public** **class** Combined\_4Q {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Person student = **new** Student("A", 20, "B.Tech", 1234);

Person professor = **new** Professor("B", 40, "Mathematics", 50000);

Person teachingAssistant = **new** TeachingAssistant("C", 22, "B.Tech", 5678, "Mathematics");

System.***out***.println("Student info");

student.getRoleInfo();

System.***out***.println();

System.***out***.println("Professor info");

professor.getRoleInfo();

System.***out***.println();

System.***out***.println("Teaching assistant info");

teachingAssistant.getRoleInfo();

}

}

5.Create:

* Interface Drawable with method draw()
* Abstract class Shape with abstract method area()  
  Subclasses: Circle, Rectangle, and Triangle.
* Calculate area using appropriate formulas.
* Demonstrate how interface and abstract class work together.

5. **package** day\_4;

**interface** Drawable {

**void** draw();

}

**abstract** **class** Shape **implements** Drawable {

**abstract** **double** area();

}

**class** Circle2 **extends** Shape {

**private** **double** r;

**public** Circle2(**double** r) {

**this**.r=r;

}

**public** **void** draw() {

System.***out***.println("Drawing a circle");

}

**public** **double** area() {

**return** Math.***PI***\*r\*r;

}

}

**class** Rectangle2 **extends** Shape {

**private** **double** l;

**private** **double** w;

**public** Rectangle2(**double** l, **double** w) {

**this**.l=l;

**this**.w=w;

}

**public** **void** draw() {

System.***out***.println("Drawing a rectangle");

}

**public** **double** area() {

**return** l\*w;

}

}

**class** Triangle **extends** Shape {

**private** **double** b;

**private** **double** h;

**public** Triangle(**double** b, **double** h) {

**this**.b=b;

**this**.h=h;

}

**public** **void** draw() {

System.***out***.println("Drawing a triangle");

}

**public** **double** area() {

**return** 0.5\*b\*h;

}

}

**public** **class** Combined\_5 {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Shape circle=**new** Circle2(5);

Shape rectangle=**new** Rectangle2(4,6);

Shape triangle=**new** Triangle(3, 7);

System.***out***.println("Circle:");

circle.draw();

System.***out***.println("Area"+circle.area());

System.***out***.println("\nRectangle:");

rectangle.draw();

System.***out***.println("Area"+rectangle.area());

System.***out***.println("\nTriangle:");

triangle.draw();

System.***out***.println("Area"+triangle.area());

}

}