1. **Answer**

class Vehicle {

void type() {

System.***out***.println("Four\_Wheeler");

}

}

class Four\_wheeler extends Vehicle {

void type1() {

System.***out***.println("Petrol\_Four\_Wheeler");

}

}

class Petrol\_Four\_Wheeler extends Four\_wheeler {

void type2() {

System.***out***.println("This Petrol\_Four\_Wheeler is a Five-Seater");

}

}

class FiveSeater\_Petrol\_Four\_Wheeler extends Petrol\_Four\_Wheeler {

void type3() {

System.***out***.println("This Five-Seater Petrol Four Wheeler is a Baleno");

}

}

class Baleno\_FiveSeater\_Petrol\_Four\_Wheeler extends FiveSeater\_Petrol\_Four\_Wheeler {

void type4() {

System.***out***.println("Baleno is manufactured by Maruti Suzuki");

}

}

public class MultiLevel {

public static void main(String[] args) {

Baleno\_FiveSeater\_Petrol\_Four\_Wheeler obj = new Baleno\_FiveSeater\_Petrol\_Four\_Wheeler();

obj.type();

obj.type1();

obj.type2();

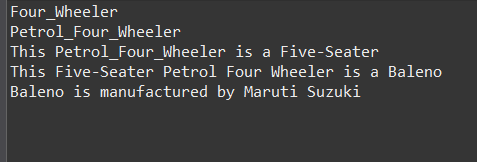
obj.type3();

obj.type4();

}

}

**OUTPUT:**

****

1. **Answer**

class Vehicle1 {

String brand = "Generic Vehicle";

Vehicle1() {

System.out.println("Vehicle constructor called");

}

void displayInfo() {

System.out.println("This is a vehicle");

}

}

class Car extends Vehicle1 {

String brand = "Car Brand";

Car() {

super();

System.out.println("Car constructor called");

}

void displayInfo() {

super.displayInfo();

System.out.println("This is a car");

}

void showBrandNames() {

System.out.println("Child brand: " + brand);

System.out.println("Parent brand: " + super.brand);

}

}

public class Superdemo {

public static void main(String[] args) {

Car myCar = new Car();

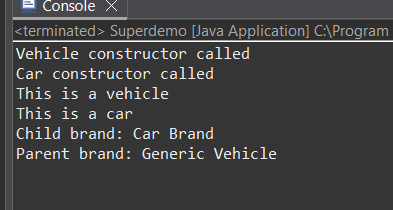
myCar.displayInfo();

myCar.showBrandNames();

}

}

**OUTPUT:**



1. **Answer**

class Hospital {

String hospitalName = "abc hospital";

String location = "Hyd";

void hospitalDetails() {

System.out.println("hospital name: " + hospitalName);

System.out.println("locatin: " + location);

}

}

class Patient extends Hospital {

String patientName;

int patientId;

Patient(String name, int id) {

this.patientName = name;

this.patientId = id;

}

void patientDetails() {

System.out.println("patient :" + patientName);

System.out.println("Patient id:" + patientId);

System.out.println("dmitted : " + hospitalName);

System.out.println("hospital lcation: " + location);

super.hospitalDetails();

}

}

public class HospitalDemo {

public static void main(String[] args) {

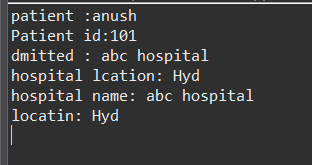
Patient p1 = new Patient("anush", 101);

p1.patientDetails();

}

}

**OUTPUT:**



1. **Answer**

class After\_12Th {

void displayOptions() {

System.out.println("streams after 12th: Engineering, Medical, Other Courses");

}

}

class Engineering extends After\_12Th {

void engineeringBranches() {

System.out.println("engineering branche: IT, Mechanical, CS");

}

}

class Medical extends After\_12Th {

void medicalBranches() {

System.out.println("medical branches: MBBS, etc");

}

}

class Other\_courses extends After\_12Th {

void otherCourseOptions() {

System.out.println("other courses: BBA, BCA");

}

}

public class HierarchicDemo {

public static void main(String[] args) {

Engineering eng = new Engineering();

eng.displayOptions();

eng.engineeringBranches();

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

Medical med = new Medical();

med.displayOptions();

med.medicalBranches();

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

Other\_courses oc = new Other\_courses();

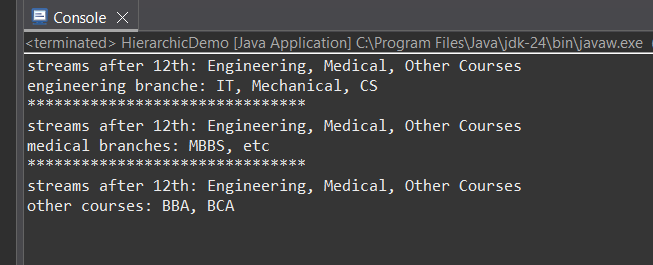
oc.displayOptions();

oc.otherCourseOptions();

}

}

**OUTPUT:**



1. **Answer**

class Hospitals {

void name(String n) {

System.out.println("name of Hospital : " + n);

}

}

class Doctor extends Hospitals {

void no(int n) {

System.out.println("doctors: " + n);

}

}

class Gynac extends Doctor {

void no(int n) {

System.out.println("gynac doctors : " + n);

}

}

class Endo extends Doctor {

void no1(int n) {

System.out.println("endo doctors : " + n);

}

}

class Cardio extends Doctor {

void no2(int n) {

System.out.println("cardio doctors : " + n);

}

}

class Nurse extends Hospitals {

void nf(int n1) {

System.out.println("nurses : " + n1);

}

}

class Accountant extends Hospitals {

void nf3(int n2) {

System.out.println("accountants available : " + n2);

}

}

class Payments extends Accountant {

void nf4(String types) {

System.out.println("type of Payments : " + types);

}

}

class Documentation extends Accountant {

void nf5(String types) {

System.out.println("type of Documentation : " + types);

}

}

public class HospitalDemo1 {

public static void main(String[] args) {

Doctor e1 = new Doctor();

e1.name("my hospitals");

e1.no(9);

Gynac g1 = new Gynac();

g1.no(6);

Endo j1 = new Endo();

j1.no1(3);

Cardio c1 = new Cardio();

c1.no2(4);

Nurse p1 = new Nurse();

p1.nf(54);

Accountant a1 = new Accountant();

a1.nf3(6);

Payments pay1 = new Payments();

pay1.nf4("all types");

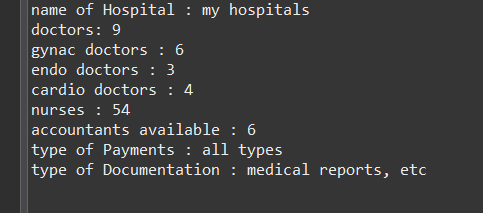
Documentation doc1 = new Documentation();

doc1.nf5("medical reports, etc");

}

}

**OUTPUT:**



**Polymorphism**

1. **Answer**

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 static void main(String[] args) {

Calculator obj = new Calculator();

System.***out***.println("Sum of 5 and 10: " + obj.add(5, 10));

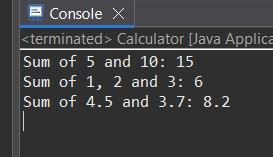
System.***out***.println("Sum of 1, 2 and 3: " + obj.add(1, 2, 3));

System.***out***.println("Sum of 4.5 and 3.7: " + obj.add(4.5, 3.7));

}

}

**OUTPUT:**



1. **Answer**

class Shape {

void area() {

System.out.println("area");

}

}

class Circle extends Shape {

double radius;

Circle(double radius) {

this.radius = radius;

}

@Override

void area() {

double result = 3.14 \* radius \* radius;

System.out.println("circle area: " + result);

}

}

class Rectangle extends Shape {

double length, width;

Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

@Override

void area() {

double result = length \* width;

System.out.println("rectangle area: " + result);

}

}

public class Shapes {

public static void main(String[] args) {

Shape obj = new Shape();

obj.area();

Shape obj1 = new Circle(5);

obj1.area();

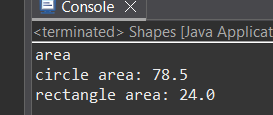
Shape obj2 = new Rectangle(4, 6);

obj2.area();

}

}

**OUTPUT:**

****

1. **Answer**

class Bank {

double getInterestRate() {

return 0.0;

}

}

class SBI extends Bank {

@Override

double getInterestRate() {

return 6.7;

}

}

class ICICI extends Bank {

@Override

double getInterestRate() {

return 7.0;

}

}

class HDFC extends Bank {

@Override

double getInterestRate() {

return 7.5;

}

}

public class Banks {

public static void main(String[] args) {

Bank sb = new SBI();

Bank ic = new ICICI();

Bank hd = new HDFC();

System.out.println("SBI interest rate: " + sb.getInterestRate() + "%");

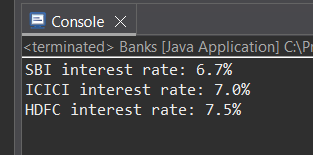
System.out.println("ICICI interest rate: " + ic.getInterestRate() + "%");

System.out.println("HDFC interest rate: " + hd.getInterestRate() + "%");

}

}

**OUTPUT:**



1. **Answer**

class Vehicles {

Vehicles() {

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

}

void run() {

System.***out***.println("Vehicle is running...");

}

}

class Bike extends Vehicles {

Bike() {

super();

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

}

*@Override*

void run() {

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

}

}

public class PolyDemo {

public static void main(String[] args) {

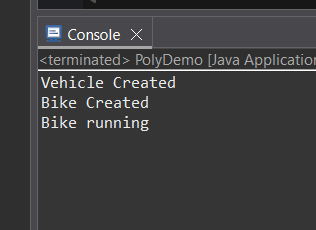
Vehicles obj = new Bike();

obj.run();

}

}

**OUTPUT:**



**Combined question answer**

abstract class SmartDevice {

abstract void turnOn();

abstract void turnOff();

abstract void performFunction();

}

class SmartPhone extends SmartDevice {

*@Override*

void turnOn() {

System.***out***.println("smartphone turn on");

}

*@Override*

void turnOff() {

System.***out***.println("smartphone turn off");

}

*@Override*

void performFunction() {

System.***out***.println("smartphone: calling and browsing");

}

}

class SmartWatch extends SmartDevice {

*@Override*

void turnOn() {

System.***out***.println("smartwatch turn on");

}

*@Override*

void turnOff() {

System.***out***.println("smartwatch turn off");

}

*@Override*

void performFunction() {

System.***out***.println("smartwatch: track fitness and time.");

}

}

class SmartSpeaker extends SmartDevice {

*@Override*

void turnOn() {

System.***out***.println("smartspeaker turn on");

}

*@Override*

void turnOff() {

System.***out***.println("smartspeaker turn off");

}

*@Override*

void performFunction() {

System.***out***.println("smartspeaker: plays music and respond to voice commands.");

}

}

public class Combined {

public static void main(String[] args) {

SmartDevice[] devices = {

new SmartPhone(),

new SmartWatch(),

new SmartSpeaker()

};

for (SmartDevice device : devices) {

device.turnOn();

device.performFunction();

device.turnOff();

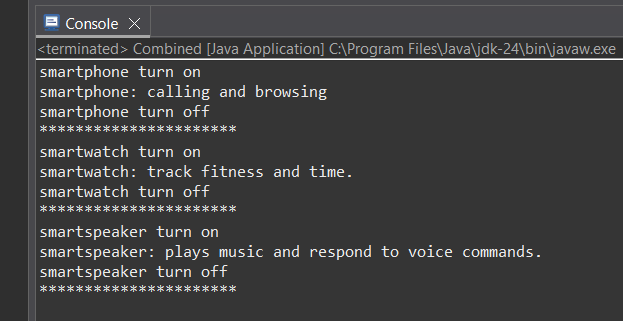
System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

}

}

**OUTPUT:**



**2. Answer**

interface Bank1 {

void deposit(double amt);

void withdraw(double amt);

double getBalance();

}

abstract class Account implements Bank1 {

protected double balance;

public Account(double initialBalance) {

this.balance = initialBalance;

}

*@Override*

public void deposit(double amt) {

if (amt > 0) {

balance += amt;

System.***out***.println("deposited: " + amt);

} else {

System.***out***.println("invalid deposit amount.");

}

}

*@Override*

public double getBalance() {

return balance;

}

}

class SavingsAccount extends Account {

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

public SavingsAccount(double initialBalance) {

super(initialBalance);

}

*@Override*

public void withdraw(double amt) {

if (amt > 0 && (balance - amt) >= ***MIN\_BALANCE***) {

balance -= amt;

System.***out***.println("withdrawl: " + amt);

} else {

System.***out***.println("withdrawal failed. minimum balance of " + ***MIN\_BALANCE*** + " must maintained.");

}

}

}

class CurrentAccount extends Account {

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

public CurrentAccount(double initialBalance) {

super(initialBalance);

}

*@Override*

public void withdraw(double amt) {

if (amt > 0 && (balance - amt) >= -***OVER\_LIMIT***) {

balance -= amt;

System.***out***.println("withdrawl: " + amt);

} else {

System.***out***.println("withdrawal failed. over limit " + ***OVER\_LIMIT*** + " exceeded.");

}

}

}

public class Accounts {

public static void main(String[] args) {

Bank1 savings = new SavingsAccount(5000);

Bank1 current = new CurrentAccount(2000);

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*savings account\*\*\*\*\*\*\*");

savings.deposit(2000);

savings.withdraw(5500);

System.***out***.println("balance: " + savings.getBalance());

System.***out***.println("\*\*\*\*\*\*\*\*\*\*current account \*\*\*\*\*\*\*\*\*\*");

current.deposit(1000);

current.withdraw(3500);

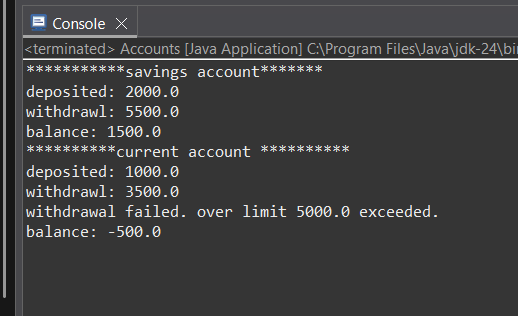
current.withdraw(6000);

System.***out***.println("balance: " + current.getBalance());

}

}

**OUTPUT:**



**3. Answer**

class Vehicle3 {

public void start() {

System.***out***.println("vehicle starts");

}

}

class Cars extends Vehicle3 {

*@Override*

public void start() {

System.***out***.println("car is starts with key ignition");

}

}

class Bikes extends Vehicle3 {

*@Override*

public void start() {

System.***out***.println("bike is starts with kick");

}

}

class Trucks extends Vehicle3 {

*@Override*

public void start() {

System.***out***.println("truck is starts with engine");

}

}

public class VehStart {

public static void testStart(Vehicle3 v) {

v.start();

}

public static void main(String[] args) {

Vehicle3 car = new Cars();

Vehicle3 bike = new Bikes();

Vehicle3 truck = new Trucks();

*testStart*(car);

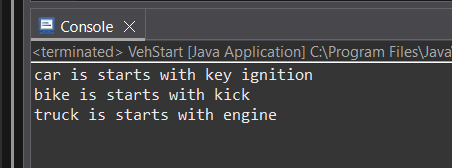
*testStart*(bike);

*testStart*(truck);

}

}

**OUTPUT:**



**4.Answer**

abstract class Person {

String name;

int age;

Person(String name, int age) {

this.name = name;

this.age = age;

}

abstract void getRoleInfo();

}

class Student extends Person {

String course;

int rollNumber;

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

super(name, age);

this.course = course;

this.rollNumber = rollNumber;

}

*@Override*

void getRoleInfo() {

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

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

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

System.***out***.println("rollno: " + rollNumber);

}

}

class Professor extends Person {

String subject;

double salary;

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

super(name, age);

this.subject = subject;

this.salary = salary;

}

*@Override*

void getRoleInfo() {

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

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

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

System.***out***.println("sal: " + salary);

}

}

class TeachingAssistant extends Student {

TeachingAssistant(String name, int age, String course, int rollNumber) {

super(name, age, course, rollNumber);

}

*@Override*

void getRoleInfo() {

System.***out***.println("teaching assistant Nnme: " + name);

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

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

System.***out***.println("rollno: " + rollNumber);

System.***out***.println("role: Both student and assistant");

}

}

public class School {

public static void main(String[] args) {

Student s = new Student("Anush", 22, "c.s", 101);

Professor p = new Professor("Dr. rao", 45, "e.c.e", 90000);

TeachingAssistant ta = new TeachingAssistant("kiran", 24, "m.l", 102);

s.getRoleInfo();

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

p.getRoleInfo();

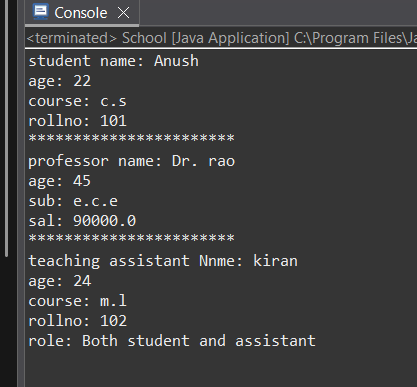
System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

ta.getRoleInfo();

}

}

**OUTPUT:**

****

1. **Answer**

interface Drawable {

void draw();

}

abstract class Shape1 implements Drawable {

abstract double area();

}

class Circle1 extends Shape1 {

private double radius;

public Circle1(double radius) {

this.radius = radius;

}

*@Override*

double area() {

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

}

*@Override*

public void draw() {

System.***out***.println("drawing a circle of radius " + radius);

}

}

class Rectangle1 extends Shape1 {

private double length, width;

public Rectangle1(double length, double width) {

this.length = length;

this.width = width;

}

*@Override*

double area() {

return length \* width;

}

*@Override*

public void draw() {

System.***out***.println("drawing a rectangle of length " + length + " and width " + width);

}

}

class Triangle1 extends Shape1 {

private double base, height;

public Triangle1(double base, double height) {

this.base = base;

this.height = height;

}

*@Override*

double area() {

return 0.5 \* base \* height;

}

*@Override*

public void draw() {

System.***out***.println("drawing a triangle of base " + base + " and height " + height);

}

}

public class ShapeDemo {

public static void main(String[] args) {

Shape1[] shapes = {

new Circle1(5),

new Rectangle1(4, 6),

new Triangle1(3, 7)

};

for (Shape1 sp : shapes) {

sp.draw();

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

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

}

}

}

**OUTPUT:**

