**Experiment no 5**

**Implement java programs to demonstrate Inheritance and method overriding.**

**Single Inheritance.**

class Shape{

public void area() { System.out.println("display area");

}

}

class Triangle extends Shape { public void area(int l, int h)

{

System.out.println(0.5 \* l \* h);

}

}

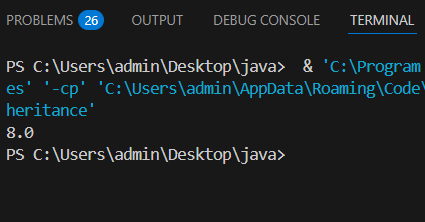
class Inheritance{

public static void main(String[] args){ Triangle t = new Triangle(); t.area(2, 8);

}

}

# Output :

****

class Maths{ int l;

int h;

void shapes() {

System.out.println("Different types of shapes ");

}

}

class geometry extends Maths{ void area(int l , int h) {

this.l=l; this.h=h;

System.out.println("calculating area of triangle ");

}

}

class Triangle extends geometry{ void print() {

System.out.println(0.5\*l\*h);

}

}

public class Mulinheritance{

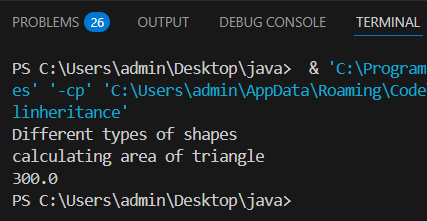
public static void main(String[] args) { Triangle t = new Triangle(); t.shapes();

t.area(30,20);

t.print();

}

}



# Hierarchical Inheritance

class Science{ void subject() {

System.out.println("Physics ");

}

void sub(){ System.out.println("Chemistry");

}

}

class Phy extends Science{ void topic(){

System.out.println("topic like : laws of motion ");

}

}

class Chm extends Science{ void chemical\_formulas() {

System.out.println("chemical\_formulas like : co2, NaCL");

}

}

public class Heinheritance{

public static void main(String[] args){

Phy p = new Phy(); p.subject();

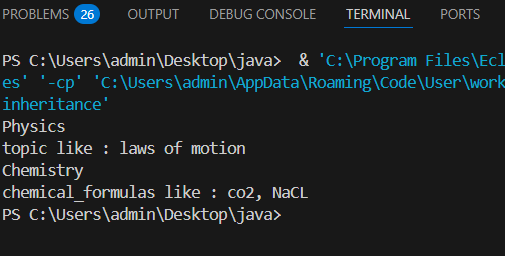
p.topic();

Chm c = new Chm(); c.sub(); c.chemical\_formulas();

}

}

# Output :

****

class Colour{ void show()

{

System.out.println("it is parent class ");

}

}

class Red extends Colour{ @Override void show()

{

System.out.println("it is child class ");

}

}

class Overriding{

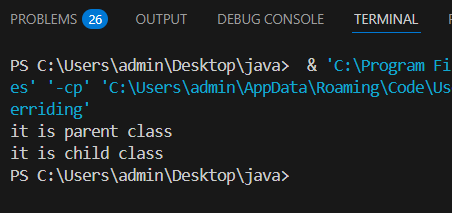
public static void main(String[] args){ Colour obj = new Colour(); obj.show();

Colour obj2 = new Red(); obj2.show();

}

}

# Output :

****

class Parent { void show()

{

System.out.println("it is parent class ");

}

}

class Child extends Parent{ @Override void show()

{

super.show();

System.out.println("it is child class ");

}

}

class Overriding{

public static void main(String[] args)

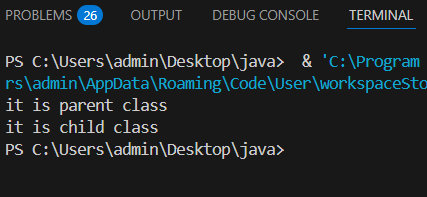
{

Child obj = new Child(); obj.show();

}

}

## Output :

****

class Parent {

protected void show()

{

System.out.println("show parent class");

}

public void display()

{

System.out.println("display parent class");

}

}

class Child extends Parent{ @Override public void show()

{

System.out.println("show child class");

}

@Override public void display()

{

System.out.println("display child class");

}

}

class Sample{

public static void main(String[] args){ Parent p = new Parent();

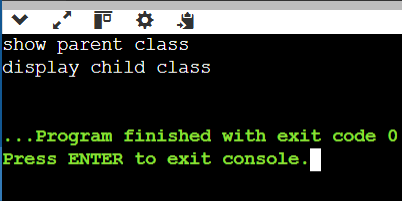
p.show();

Child c = new Child(); c.display();

}

}

## Output :



**Multiple Inheritance**

interface F\_base

{

int add(int a , int b);

}

interface S\_base

{

int sub(int a , int b);

int mul(int a , int b);

}

class Sample implements F\_base,S\_base

{

public int add(int a , int b)

{

return a+b;

}

public int sub(int a , int b)

{

return a-b;

}

public int mul(int a , int b)

{

return a\*b;

}

}

class Multinheritance

{

public static void main(String [] args)

{

Sample s = new Sample();

int c = s.add(4,6);

int d =s.sub(9,4);

int r=s.mul(4,2);

System.out.println("Addition is :"+c);

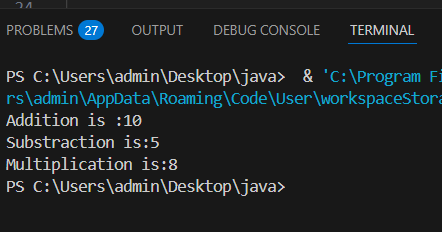
System.out.println("Substraction is:"+d);

System.out.println("Multiplication is:"+r);

}

}

## Output :



**Hybrid Inheritance**

class Operation

{

void add(int a, int b)

{

System.out.println("Addition: " + (a + b));

}

}

class Sub extends Operation

{

void subtract(int a, int b)

{

System.out.println("Subtraction: " + (a - b));

}

}

class Mul extends Sub

{

void multiply(int a, int b)

{

System.out.println("Multiplication: " + (a \* b));

}

}

public class HybridInheritance

{

public static void main(String[] args)

{

Mul m = new Mul();

m.add(10, 5);

m.subtract(10, 5);

m.multiply(10, 5);

}

}

## Output :

