**202250300  
WEEK-7**

**ASSIGNMENT-6**

**Q1.Single Inheritance:**

***CODE:***

class animal{

String family="Animalia";

String name="";

public void eat(){

System.out.println("Eating.....");

}

void details(){

System.out.println("FAMILY:"+family+" Name:"+name);

}

public void sound(){}

}

class dog extends animal{

public void sound(){

System.out.println("ruff!ruff!");

}

}

public class single\_inheritance\_3003 {

public static void main(String[] args) {

System.out.println("R.Prabhakara Arjun\n2022503003\n");

dog a=new dog();

a.details();

a.name="Jimmy";

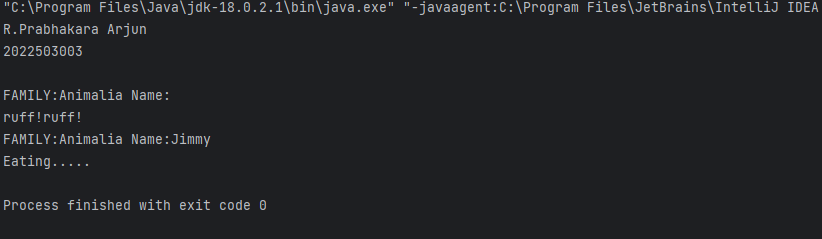
a.sound();

a.details();

a.eat();

}

}



**Q2.Multiple inheritance**

***CODE:***

class Kingdom{

String Kname="ANIMALIA";

}

class Phylum extends Kingdom{

String Pname="CHORDATA";

}

class Classs extends Phylum{

String Cname="MAMMALIA";

}

class Order extends Classs{

String Oname="CARNIVORA";

}

class Family extends Order{

String Fname="FELIDAE";

}

class Genus extends Family{

String Gname;

public void setGname(String n){

Gname=n;

}

}

class Species extends Genus{

String Sname;

public void setSname(String n){

Sname=n;

}

void Printehh(){

System.out.println(Kname+" "+Pname+" "+Cname+" "+Oname+" "+Gname+" "+Sname);

}

}

class multiple\_inheritance\_3003{

public static void main(String[] args){

System.out.println("R.Prabhakara Arjun\n2022503003\n");

Species Cat=new Species();

Cat.setGname("Felis");

Cat.setSname("Catus");

Cat.Printehh();

Species Lion=new Species();

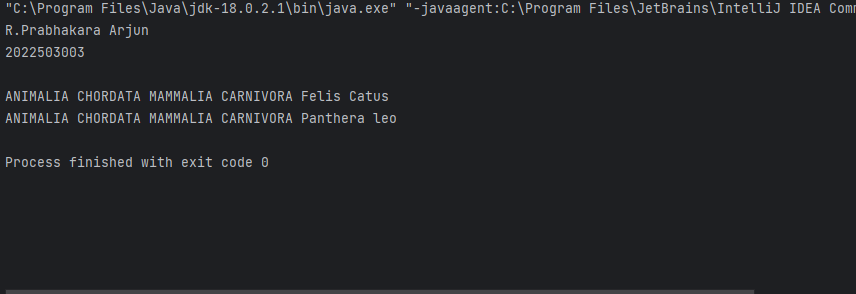
Lion.setGname("Panthera");

Lion.setSname("leo");

Lion.Printehh();

}

}



**Q3..Hierarchical Inheritance**

***CODE:***

import java.util.\*;

class animal{

String scientific\_name="";

String family="Animalia";

String name="";

public void eat(){

System.out.println("Eating.....");

}

void details(){

System.out.println("FAMILY:"+family+" Name:"+name);

}

void details(Boolean a){

if(a){

System.out.println("FAMILY:"+family+" Name:"+name+" Scientific\_name:"+scientific\_name);

}

}

void theMethod(){

details();

Scanner input=new Scanner(System.in);

System.out.println("ENTER NAME FOR THE ANIMAL:"+scientific\_name+"\nENTER NAME OF UR ANIMAL:");

name=input.nextLine();

details();

details(true);

System.out.println("-----------------------------------------------");

}

public void sound(){}

}

class cat extends animal{

cat(){

scientific\_name="felis catus";

}

public void sound(){

System.out.println("meow!meow!!");

}

}

class lion extends animal{

lion(){

scientific\_name="panthera leo";

}

public void sound(){

System.out.println("ROARS!!");

}

}

public class heirarchy\_3003 {

public static void main(String[] args) {

System.out.println("R.Prabhakara Arjun\n2022503003\n");

dog a=new dog();

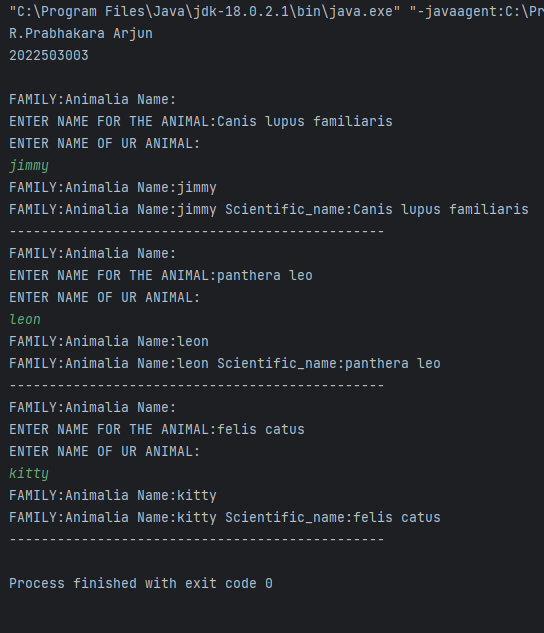
a.theMethod();

lion b=new lion();

b.theMethod();

cat c=new cat();

c.theMethod();

 }

}

**Q4.Multiple Inheritance (class? Vs Interface)**

***CODE:***

class plants{

String scientificName="";

String modeOfNutrietent="";

}

class organsism\_combo extends animal,plants{

void display(){

System.out.println("This won't even run!!");

}

}

public class multiple\_inheritance\_class\_3003{

public static void main(String[] args){

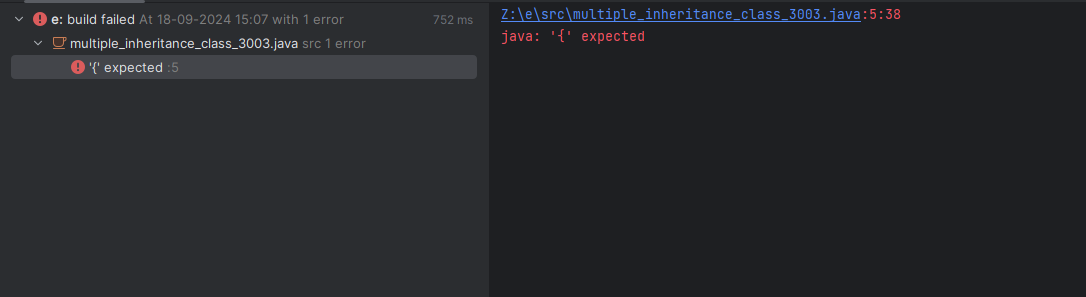
organsism\_combo org=new organsism\_combo();

org.display();

System.out.println("This won't even run!!");

}

}



**Multiple Inheritance (class? Vs *Interface)***

import java.util.Scanner;

interface animal1{

String scientific\_name="";

String family="Animalia";

String name="";

public void eat();

void details();

void details(Boolean a);

public void sound();

}

interface plant {

String scientific\_name="";

String family="plantae";

String mode\_of\_nutritent="";

}

class organisms implements plant,animal1 {//if implements it should redfine all the methods!!!

public void eat(){

System.out.println("Eating.....");

}

public void details(){}

public void details(Boolean a){//should be public

}

public void sound(){}

}

public class multiple\_inheritance\_interface\_3003 {

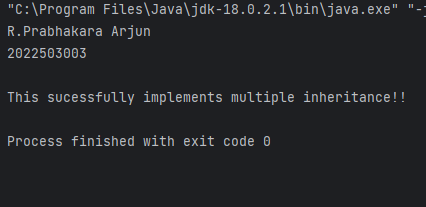
public static void main(String[] args){

System.out.println("R.Prabhakara Arjun\n2022503003\n");

System.out.println("This sucessfully implements multiple inheritance!!");

}

}



**Q5.Polymorphism**

***CODE:***

class parent {

void print() {

System.out.println("print() with nothing passed as parameter!!");

}

void print(int a) {

System.out.println("print() when parameter is passed!!");

}

}

class child{

void print(){

System.out.println("print() override!!");

}

void print(int a){

System.out.println("print(parameter) is overloaded in child!!");

}

}

public class polymorphism\_3003 {

public static void main(String[] args) {

System.out.println("R.Prabhakara Arjun\n2022503003\n");

child a=new child();

a.print();

a.print(1);

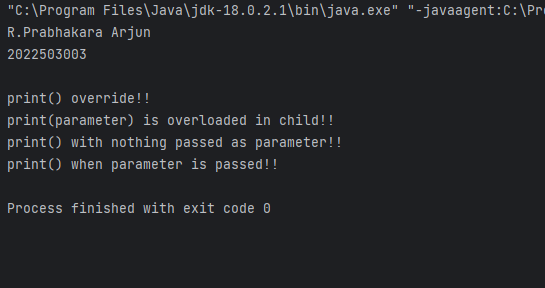
parent b=new parent();

b.print();

b.print(1);

}

}



**Q6.Polymorphism,Constructor in inheritance,package**

***CODE:***

package p1;

import p1.p2.\*;

class poly\_3003{

public static void main(String[] args){

System.out.println("R.Prabhakara Arjun\n2022503003");

animal abc=new animal();

animal xyz=new dog();

animal pqr=new cat();

abc.sound();

xyz.sound();

pqr.sound();

}

}

package p1.p2;

import java.util.\*;

public class animal{

String scientific\_name="";

String family="Animalia";

String name="";

public void eat(){

System.out.println("Eating.....");

}

void details(){

System.out.println("FAMILY:"+family+" Name:"+name);

}

public void sound(){

System.out.println("drow........");

}

void details(Boolean a){

if(a){

System.out.println("FAMILY:"+family+" Name:"+name+" Scientific\_name:"+scientific\_name);

}

}

void theMethod(){

details();

Scanner input=new Scanner(System.in);

System.out.println("ENTER NAME FOR THE ANIMAL:"+scientific\_name+"\nENTER NAME OF UR ANIMAL:");

name=input.nextLine();

details();

details(true);

System.out.println("-----------------------------------------------");

}

}

package p1.p2;

import p1.p2.\*;

public class cat extends animal{

public cat(){

scientific\_name="felis catus";

}

public void sound(){

System.out.println("meow!meow!!");

}

}package p1.p2;

import p1.p2.\*;

public class dog extends animal{

publicdog(){

scientific\_name="Canis lupus familiaris";

}

public void sound(){

System.out.println("ruff!ruff!");

}

}

package p1.p2;

import p1.p2.\*;

public class dog extends animal{

public dog(){

scientific\_name="Canis lupus familiaris";

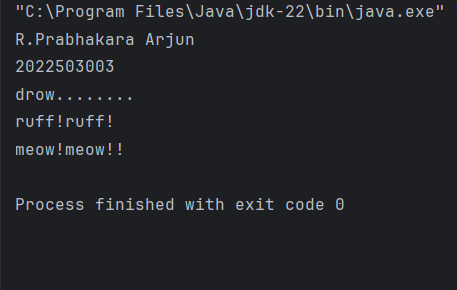
}

public void sound(){

System.out.println("ruff!ruff!");

}

}

****

**Super usage in inheritance**

package p1;

import p1.p2.\*;

class hybrid\_dog extends dog{

hybrid\_dog(){

super();

System.out.println(scientific\_name);

scientific\_name="Canis lupus ";

System.out.println(scientific\_name);

}

public void sound(){

System.out.println("dog's sound:");

super.sound();

}

}

class poly\_3003{

public static void main(String[] args){

System.out.println("R.Prabhakara Arjun\n2022503003");

animal abc=new animal();

animal xyz=new dog();

animal pqr=new cat();

/\*abc.sound();

xyz.sound();

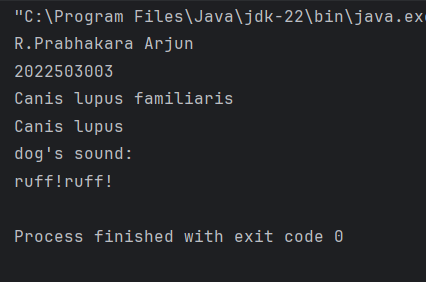
pqr.sound();\*/

hybrid\_dog diddy=new hybrid\_dog();

diddy.sound();

}

}

****

**Q7.Shallow Vs Deep Copy**

***CODE:***

import java.lang.\*;

class dog implements Cloneable{

String name="";

String breed="";

User user\_thing=new User();

dog(String name, String breed){

this.name=name;

this.breed=breed;

}

protected Object clone() throws CloneNotSupportedException{

return super.clone();

}

void displayDetails(){

System.out.println("Name of the "+name+".Breed is "+breed+".THe owner "+user\_thing.name+".His number is "+user\_thing.number);

}

}

class User implements Cloneable{

String name;

int number;

protected Object clone() throws CloneNotSupportedException{

return super.clone();

}

}

public class shallow\_deep\_copy\_3003 {

public static void main(String[] args){

System.out.println("R.Prabahakara Arjun\n2022503003\n");

try {

dog d1 = new dog("Jimmy","great dan" );

dog d2 = (dog) d1.clone();

d1.displayDetails();

d2.displayDetails();

System.out.println("comparing obj and clone:"+(d1==d2)+".comapring obj user and clone:"+(d1.user\_thing==d2.user\_thing));

d2.name="Wippy";d2.breed="german sheepard";

d1.displayDetails();

d2.displayDetails();

System.out.println("comparing obj and clone:"+(d1==d2)+".comapring obj user and clone:"+(d1.user\_thing==d2.user\_thing));

System.out.println("SHALLOW COPY:\ncomparing obj and clone:"+(d1==d2)+".comapring obj user and clone:"+(d1.user\_thing==d2.user\_thing));

d2.user\_thing=(User) d2.user\_thing.clone();

System.out.println("DEEP COPY:\ncomparing obj and clone:"+(d1==d2)+".comapring obj user and clone:"+(d1.user\_thing==d2.user\_thing));

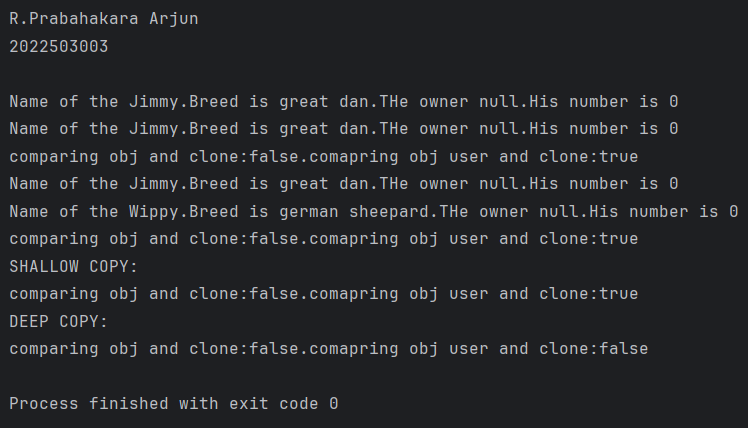
} catch (CloneNotSupportedException e) {

e.printStackTrace();

}

}

}

****

**Q8.FINALIZE**

***CODE:***

/\*

protected void finalize() throws Throwable {

try {

System.out.println("Object is being finalized.");

} finally {

// Ensure the base class finalize is also called

super.finalize();

}

}\*/

Finalize is deprecated!!!