**Lab 4**

**Aditya kamble**

1)-Method Overloading: Write a class Calculator with overloaded methods add(). Implement add() methods that take:

     - Two integers

     - Two double values

     - Three integers

     - A variable number of integers

-**package** lab4;

**public** **class** calciaddmethod {

**double** a;

**double** b;

**double** c;

String raju;

**public** **int** add(**int** a,**int** b)

{

**return** a+b;

}

**public** **double** add(**double** a,**double** b)

{

**return** a+b;

}

**public** **int** add(**int** a,**int** b,**int** c)

{

**return** a+b+c;

}

**public** String add(**int** a,String name)

{

**return** a+name;

}

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

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

calciaddmethod obj1=**new** calciaddmethod();

**int** result1=obj1.add(4,4);

calciaddmethod obj2=**new** calciaddmethod();

**double** result2=obj2.add(4.1,1.2);

calciaddmethod obj3=**new** calciaddmethod();

**int** result3=obj3.add(4,4,4);

calciaddmethod obj4=**new** calciaddmethod();

//String result4=obj4.add(4, "raju" );

System.***out***.println("Addition of two integers ="+result1);

System.***out***.println("Addition of two double integers ="+result2);

System.***out***.println("Addition of three integers="+result3);

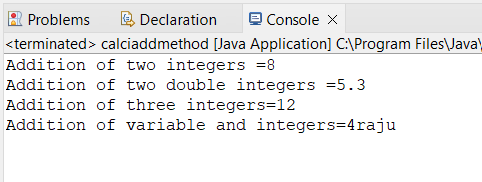
String s=obj4.add(4, "raju" );

System.***out***.println("Addition of variable and integers="+s);

}

}

Output-



**(2)** Super Keyword: Create a class Person with a constructor that accepts and sets name and age.

   - Create a subclass Student that adds a grade property and initializes name and age using the super keyword in its constructor.

   - Demonstrate the creation of Student objects and the usage of super to call the parent class constructor.

**-package** lab4;

**class** student{

**public** **void** setage(**int** age)

{

System.***out***.println("Age of the student="+age);

}

**public** **void** setname(String name)

{

System.***out***.println("Age of the student="+name);

}

}

**public** **class** personn **extends** student {

personn(){

**super**.setage(10);

**super**.setname("raju");

}

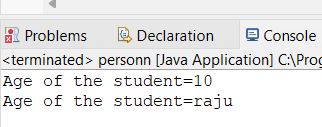
**public** **static** **void** main(String[] args) {

personn o=**new** personn();

}

}

**Output-**

****

**(3)-** Super Keyword: Create a base class Shape with a method draw() that prints "Drawing Shape".

   - Create a subclass Circle that overrides draw() to print "Drawing Circle".

   - Inside the draw() method of Circle, call the draw() method of the Shape class using super.draw().

   - Write a main method to demonstrate calling draw() on a Circle object

-

**package** lab4;

**class** circle

{ String s="draw circle";

**public** **void** draw(String s)

{

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

}

}

**public** **class** shape **extends** circle {

shape()

{

**super**.draw(s);

}

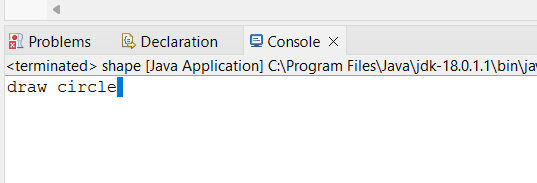
**public** **static** **void** main(String[] args) {

shape obj1=**new** shape();

}

}

Output-



**(4)-** Create a base class BankAccount with a method deposit(amount) and a constructor that sets the initial balance.

   - Create a subclass SavingsAccount that overrides deposit(amount) to add interest before depositing. Use the super keyword to call the deposit method of the base class.

   - Write a main method to demonstrate creating a SavingsAccount and depositing an amount to see the effect of interest.

-

**package** lab4;

**class** savingaccount

{

**int** intrest;

**public** **void** deposite(**int** intrest)

{

System.***out***.println("intrest="+intrest);

**this**.intrest=intrest;

}

**int** deposites;

**public** **void** deposites(**int** deposites)

{

**int** total=deposites+intrest;

System.***out***.println("total(depoiste+intrest)="+total);

**this**.deposites=deposites;

}

}

**public** **class** bankaccount **extends** savingaccount{

bankaccount()

{

**super**.deposite(1000);

**super**.deposites(5000);

}

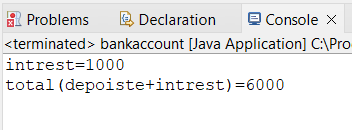
**public** **static** **void** main(String[] args) {

bankaccount obj1=**new** bankaccount();

}

}

 OUTPUT-



**(5)-** Define a class Employee with properties name and salary and a method displayDetails().

   - Create a subclass Manager that adds a property department and overrides displayDetails() to include department details. Use the super keyword to call the displayDetails() method of Employee within Manager.

   - In the main method, create objects of Employee and Manager and call displayDetails() to show the details.

-

**package** lab4;

**class** manager

{

String name;

**void** displaydetails(String name)

{

System.***out***.println("emp name="+"raju");

**this**.name=name;

}

**int** salary;

**void** displaydetails(**int** salary)

{

System.***out***.println("emp salary="+50000);

**this**.salary=salary;

}

}

**public** **class** emp **extends** manager{

emp()

{

**super**.displaydetails(name);

**super**.displaydetails(salary);

}

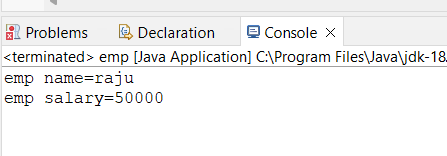
**public** **static** **void** main(String[] args) {

emp obj=**new** emp();

manager obj1=**new** manager();

}

}

Output-

(6)Write the same programme for the class ImmutableExample, to achieve object value ‘Hi’.

-

**package** lab4;

**public** **class** ImmutableExample{

**public** **final** String value;

**public** ImmutableExample(String value)

{

**this**.value=value;

}

**public** String getvalue()

{

**return** value;

}

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

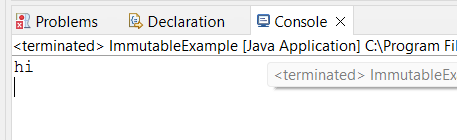
ImmutableExample obj=**new** ImmutableExample("hi");

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

}

}

Output-



**(7)-** Write the same programme for the class MutableExample, to output the object values ‘hello 2’ and ‘hello3’.

-

**package** lab4;

**public** **class** mutableExample

{

**private** String s;

mutableExample(String s) {

**this**.s = s;

}

**public** String getName() {

**return** s;

}

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

mutableExample obj = **new** mutableExample("HELLO 2");

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

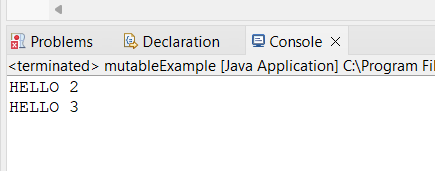
mutableExample obj1 = **new** mutableExample("HELLO 3");

System.***out***.println(obj1.getName());

}

}

Output-



(4)Write a java class to implement any 10 string methods:

● replace ● contains ● replaceAll ● indexOf ● substring ● Equals ● lastIndexOf ● startsWith

● endsWith ● EqualsIgnoreCase ● toLowerCase ● toUpperCase ● isEmpty ● Length ● split-

-

**public** **class** Stringexamples {

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

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

String s="Aditya is playing";

String newString=s.replace("is","was");

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

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

//---------------------------------------------------------

String sampleString = "Hello, World!";

**boolean** containsWorld = sampleString.contains("World");

**boolean** containsJava = sampleString.contains("Java");

// Print the results

System.***out***.println("Does the string contain 'World'? " + containsWorld);

System.***out***.println("Does the string contain 'Java'? " + containsJava);

//--------------------------------------------------------------

String p="Aditya is playing and is played well";

String newString1=p.replaceAll("is","was");

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

System.***out***.println(newString1);

//--------------------------------------------------------------

String t = "Hello, World!";

**int** index1 = t.indexOf('W');

**int** index2 = t.indexOf("World");

System.***out***.println("Index of 'W': " + index1);

System.***out***.println("Index of 'World': " + index2);

}

}

-output-

