**DAY - 3 Solutions**

**Lab Exercise No:**34

**Exercise Objective(s):***The concept of inheritance*

**Exercise:***Create a class called Vehicle. Create subclasses like Truck, Bus, Car etc. Add common methods in the base class and specific methods in the corresponding class. Create a class called Road and create objects for the Truck, Car, Bus etc and display the appropriate message.*

**Solution:**

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* Assignment No. 34

\* Create a class called Vehicle. Create subclasses like Truck, Bus, Car etc. Add common methods

in the base class and specific methods in the corresponding class. Create a class called Road

and create objects for the Truck, Car, Bus etc and display the appropriate message.

\*/

class Vehicle {

public Vehicle() {

super();

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

}

}

class Car extends Vehicle {

public Car() {

super();

}

public void printDesc() {

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

}

}

class Truck extends Vehicle {

public Truck() {

super();

}

public void printDesc() {

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

}

}

class Bus extends Vehicle {

public Bus() {

super();

}

public void printDesc() {

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

}

}

public class Road {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

Car c = new Car();

c.printDesc();

System.out.println();

Truck t = new Truck();

t.printDesc();

System.out.println();

Bus b = new Bus();

b.printDesc();

System.out.println();

}

}

***Lab Exercise No:****35*

***Exercise Objective(s):****super keyword*

***Exercise:****In the Lab Exercise 34, in the Vehicle class constructor initialize few variables like color, no of wheels, model etc. Give appropriate values for these variables from the invoking subclass.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* The code contains the Vehicle class constructor initializing few variables like color, no of

wheels, model etc. Give appropriate values for these variables from the invoking subclass.

\*/

class Vehicle {

String color;

int noOfWheels;

String modelName;

public Vehicle() {

super();

}

public Vehicle(String color, int noOfWheels, String modelName) {

super();

this.color = color;

this.noOfWheels = noOfWheels;

this.modelName = modelName;

}

}

class Bike extends Vehicle {

public Bike() {

super("Red",2,"Ducati V4S");

}

public void details() {

System.out.println("Bike Name : " + this.modelName + "\n" + "Bike Color : " + this.color + "\n" + "No. of Wheels : " + this.noOfWheels);

}

}

class Car extends Vehicle {

public Car() {

super("Blue",4,"Buggati Chiron");

}

public void details() {

System.out.println("Car Name : " + this.modelName + "\n" + "Car Color : " + this.color + "\n" + "No. of Wheels : " + this.noOfWheels);

}

}

class Truck extends Vehicle {

public Truck() {

super("Black",2,"Tesla Cybertruck");

}

public void details() {

System.out.println("Truck Name : " + this.modelName + "\n" + "Truck Color : " + this.color + "\n" + "No. of Wheels : " + this.noOfWheels);

}

}

public class Solution35 {

public static void main(String[] args) {

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

Bike b = new Bike();

b.details();

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

Car c = new Car();

c.details();

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

Truck t = new Truck();

t.details();

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

}

}

**Lab Exercise No:**36

**Exercise Objective(s):***protected access specifier*

**Exercise:***In the Lab Exercise 35, create another class called City which creates an object for the Car,Truck and Bus class and displays the details through a display () method in the Vehicle class.The other methods and data members should not be accessible by the City class.*

**Solution:**

/\*\*

\* @author jayesh

\* Create another class called City which creates an object for the Car,

Truck and Bus class and displays the details through a display () method in the Vehicle class.

The other methods and data members should not be accessible by the City class.

\*/

class Vehicle {

private String color;

private int noOfWheels;

private String modelName;

public Vehicle() {

super();

}

public Vehicle(String color, int noOfWheels, String modelName) {

super();

this.color = color;

this.noOfWheels = noOfWheels;

this.modelName = modelName;

}

protected void display() {

System.out.println("Vehicle Type : " + this.getClass().getName() + "\n" +"Vehicle Model" + this.modelName + "\n" + "Vehicle color : " + this.color + "\n" + "Vehicle no. of wheels : " + this.noOfWheels);

}

}

class Bike extends Vehicle {

public Bike() {

super("Red",2,"Ducati V4S");

}

}

class Car extends Vehicle {

public Car() {

super("Blue",4,"Buggati Chiron");

}

}

class Truck extends Vehicle {

public Truck() {

super("Black",2,"Tesla Cybertruck");

}

}

class City extends Vehicle {

public static void main(String[] args) {

Bike b = new Bike();

b.display();

Car c = new Car();

c.display();

Truck t = new Truck();

t.display();

}

}

**Lab Exercise No:**39

**Exercise Objective(s):***Polymorphism*

**Exercise:***Create a class called Worker. Write classes DailyWorker and SalariedWorker that inherit from Worker.Every worker has a name and a salary rate. Write method Pay (int hours) to compute the week pay of every worker. A Daily worker is paid on the basis of the number of days she/he works.The salaried worker gets paid the wage for 40 hours a week no matter what the actual hours are. Test this program to calculate the pay of workers.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\*Create a class called Worker. Write classes DailyWorker and SalariedWorker that inherit from

Worker.Every worker has a name and a salaryrate. Write method Pay (int hours) to compute

the week pay of every worker. A Daily worker is paid on the basis of the number of days

she/he works.The salaried worker gets paid the wage for 40 hours a week no matter what the

actual hours are. Test this program to calculate the pay of workers.

\*/

class Worker {

String wName;

public Worker(String wName) {

this.wName = wName;

}

}

class DailyWorker extends Worker {

int rate;

public DailyWorker(String wName,int rate) {

super(wName);

this.rate = rate;

}

public void pay(int hours) {

System.out.println("DailyWorker Name : " + this.wName + "\n" + "DailyWorker Salary : " + this.rate\*hours);

}

}

class SalariedWorker extends Worker {

int rate;

public SalariedWorker(String wName, int rate) {

super(wName);

this.rate = rate;

}

public void pay() {

System.out.println("SalariedWorker Name : " + this.wName + "\n" + "SalariedWorker Salary : " + rate\*40);

}

}

public class Main {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

DailyWorker d = new DailyWorker("Jayesh",30);

d.pay(52);

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

SalariedWorker s = new SalariedWorker("Saurabh",40);

s.pay();

}

}

**Lab Exercise No:**41

**Exercise Objective(s):***abstract classes*

**Exercise:***Create a class called Shape3D with the following method signatures alone, volume () and surfaceArea (). Then create subclasses like Cylinder, Sphere, and Cubeetc and implement these methods.*

**Solution:**

/\*\*

\* @author jayesh

\* Create a class called Shape3D with the following method signatures alone, volume () and

surfaceArea (). Then create subclasses like Cylinder, Sphere, and Cube,etc and implement

these methods.

\*/

package com.hsbc.pack;

abstract class Shape3D {

public abstract void volume();

public abstract void surfaceArea();

}

class Cylinder extends Shape3D {

int r,h;

public Cylinder(int r,int h) {

this.r = r;

this.h = h;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of cylinder is : " + Math.PI\*this.r\*this.r\*this.h);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cylinder is : " + 2\*Math.PI\*this.r\*(this.r+this.h));

}

}

class Sphere extends Shape3D {

int r;

public Sphere(int r) {

this.r = r;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of sphere is : " + (double)(4\*Math.PI\*this.r\*this.r\*this.r)/3);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cylinder is : " + 4\*Math.PI\*this.r\*this.r);

}

}

class Cube extends Shape3D {

int side;

public Cube(int side) {

this.side = side;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of cube is : " + this.side\*this.side\*this.side);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cube is : " + 6\*this.side\*this.side);

}

}

public class Shape3DCalculator {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

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

Cylinder cy = new Cylinder(4,5);

cy.volume();

cy.surfaceArea();

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

Sphere s = new Sphere(7);

s.volume();

s.surfaceArea();

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

Cube cu = new Cube(6);

cu.volume();

cu.surfaceArea();

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

}

}

**Lab Exercise No:**43

**Exercise Objective(s):***Implementation of an interface*

**Exercise:***Implement Lab Exercise 41using Interfaces.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\*Create a interface called Shape3D with the following method signatures alone, volume () and

surfaceArea (). Then create classes like Cylinder, Sphere, and Cubeetc and implement

these methods.

\*/

interface Shape3D {

public void volume();

public void surfaceArea();

}

class Cylinder implements Shape3D {

int r,h;

public Cylinder(int r,int h) {

this.r = r;

this.h = h;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of cylinder is : " + Math.PI\*this.r\*this.r\*this.h);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cylinder is : " + 2\*Math.PI\*this.r\*(this.r+this.h));

}

}

class Sphere implements Shape3D {

int r;

public Sphere(int r) {

this.r = r;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of sphere is : " + (double)(4\*Math.PI\*this.r\*this.r\*this.r)/3);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cylinder is : " + 4\*Math.PI\*this.r\*this.r);

}

}

class Cube implements Shape3D {

int side;

public Cube(int side) {

this.side = side;

}

@Override

public void volume() {

// TODO Auto-generated method stub

System.out.println("Volume of cube is : " + this.side\*this.side\*this.side);

}

@Override

public void surfaceArea() {

// TODO Auto-generated method stub

System.out.println("Surface area of cube is : " + 6\*this.side\*this.side);

}

}

public class IShape3DCalculator {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

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

Cylinder cy = new Cylinder(4,5);

cy.volume();

cy.surfaceArea();

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

Sphere s = new Sphere(7);

s.volume();

s.surfaceArea();

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

Cube cu = new Cube(6);

cu.volume();

cu.surfaceArea();

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

}

}

**Lab Exercise No:**44

**Exercise Objective(s):***Implementing more than one interface.*

**Exercise:***Create two interfaces namely Drawable and Fillable. Create class called Line, Circle, Square and implement following methods through interface.*

*<I>Drawable ------ drawingColor(), thickness()*

*<I>Fillable ---------- fillingColor(), size()*

**Solution :**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* Create two interfaces namely Drawable and Fillable. Create class called Line, Circle, Square and

implement following methods through interface.

<I>Drawable ------ drawingColor(), thickness()

<I>Fillable ---------- fillingColor(), size()

\*/

interface Drawable {

public void drawingColor();

public void thickness();

}

interface Fillable {

public void fillingColor();

public void size();

}

class Line implements Drawable, Fillable {

int len,thick;

String dCol;

public Line(int len, String dCol, int thick) {

this.len = len;

this.dCol = dCol;

this.thick = thick;

}

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("No filling color possible for line");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("The length of line is " +this.len);

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("The color of line is " + this.dCol);

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("The thickness of line is " + this.thick);

}

}

class Circle implements Drawable, Fillable {

int rad, thick;

String fCol,dCol;

public Circle(int rad, int thick, String fCol, String dCol) {

super();

this.rad = rad;

this.thick = thick;

this.fCol = fCol;

this.dCol = dCol;

}

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("Fill color of circle is " + this.fCol);

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("The size or area of circle is " + Math.PI\*this.rad\*this.rad);

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("Draw color of circle is " + this.dCol);

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness of circle is " + this.thick);

}

}

class Square implements Drawable, Fillable {

int side, thick;

String fCol,dCol;

public Square(int side, int thick, String fCol, String dCol) {

super();

this.side = side;

this.thick = thick;

this.fCol = fCol;

this.dCol = dCol;

}

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("Fill color of square is " + this.fCol);

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("the size or area of square is " + this.side\*this.side);

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("Draw color of square is " + this.dCol);

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness of square is " + this.thick);

}

}

public class Solution44 {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

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

Line l = new Line(5,"Blue",3);

l.drawingColor();

l.fillingColor();

l.thickness();

l.size();

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

Circle c = new Circle(6,4,"Red","Black");

c.drawingColor();

c.fillingColor();

c.thickness();

c.size();

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

Square s = new Square(8,2,"Yellow","Green");

s.drawingColor();

s.fillingColor();

s.thickness();

s.size();

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

}

}

**Lab Exercise No:**45

**Exercise Objective(s):***Implementation of an interface*

**Exercise:***In Lab Exercise 39, create a package called finance and within it create an interface called Payable.It should define the getPayInfo() method that all the worker classes will implement. Now display the details of the monthly pay of the workers.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.finance;

/\*\*

\* @author jayesh

\* create a package called finance and within it create an interface called Payable.

\* It should define the getPayInfo() method that all the worker classes will implement.

\* Now display the details of the monthly pay of the workers

\*/

interface Payable {

void getPayInfo();

}

class DailyWorker implements Payable {

String wName;

int rate,dailyHrs;

public DailyWorker(String wName, int rate, int dailyHrs) {

this.wName = wName;

this.rate = rate;

this.dailyHrs = dailyHrs;

}

@Override

public void getPayInfo() {

// TODO Auto-generated method stub

System.out.println("Payroll Information of Daily Worker(28-day schedule):");

System.out.println("Name : " + this.wName);

System.out.println("Monthly Salary : " + this.rate\*this.dailyHrs\*28);

}

}

class SalariedWorker implements Payable {

String wName;

int rate;

public SalariedWorker(String wName, int rate) {

this.wName = wName;

this.rate = rate;

}

@Override

public void getPayInfo() {

// TODO Auto-generated method stub

System.out.println("Payroll Information of Salaried Worker(28-day schedule):");

System.out.println("Name : " + this.wName);

System.out.println("Monthly Salary : " + this.rate\*40\*4);

}

}

public class MonthlyMain {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

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

DailyWorker d = new DailyWorker("Jayesh",200,7);

d.getPayInfo();

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

SalariedWorker s = new SalariedWorker("Ashish",300);

s.getPayInfo();

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

}

}

**Lab Exercise No:**47

**Exercise Objective(s):***Inner classes*

**Exercise:***Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.Create an inner class called InterestAdder and implement the interest calculations and add the interest to the current balance.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.

\* Create an inner class called InterestAdder and implement the interest calculations and add

\* the interest to the current balance.

\*/

class BankAccount {

String accName;

float balance;

/\*\*

\* @param accName

\* @param balance

\*/

public BankAccount(String accName, float balance) {

super();

this.accName = accName;

this.balance = balance;

}

public void getBalance() {

System.out.println("Updated Balance of user " + this.accName + " : " + this.balance);

}

public void withdraw(int amount) {

this.balance = this.balance - amount;

}

public void deposit(int amount) {

this.balance = this.balance + amount;

}

class InterestAdder {

void interest(int months) {

float i = (float)((months \* balance \* 0.12)/12);

balance += i;

System.out.println("The balance after " + months + " months is " + balance);

}

}

}

public class InterestCalculator {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

BankAccount b = new BankAccount("Jayesh", 12000);

BankAccount.InterestAdder ia = b.new InterestAdder();

ia.interest(42);

}

}

**Lab Exercise No:**48

**Exercise Objective(s):***Local inner classes*

**Exercise:***Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.Create a local inner class inside the getBalance() method called InterestAdder and implement the interest calculations and add the interest to the current balance.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.

\* Create an inner class called InterestAdder and implement the interest calculations and add

\* the interest to the current balance.

\*/

class BankAccount {

String accName;

float balance;

/\*\*

\* @param accName

\* @param balance

\*/

public BankAccount(String accName, float balance) {

super();

this.accName = accName;

this.balance = balance;

}

public void getBalance() {

System.out.println("Updated Balance of user " + this.accName + " : " + this.balance);

}

public void withdraw(int amount) {

this.balance = this.balance - amount;

}

public void deposit(int amount) {

this.balance = this.balance + amount;

}

public void interestCalc(int m) {

class InterestAdder {

void interest(int months) {

float i = (float)((months \* balance \* 0.12)/12);

balance += i;

System.out.println("The balance after " + months + " months is " + balance);

}

}

InterestAdder i = new InterestAdder();

i.interest(m);

}

}

public class InterestCalculator {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

BankAccount b = new BankAccount("Jayesh", 12000);

b.interestCalc(42);

}

}

**Lab Exercise No:**49

**Exercise Objective(s):***Anonymous inner classes*

**Exercise:***Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.Create an anonymous inner class in the getBalance() method to do the interest calculations and add the interest to the current balance.*

**Solution:**

/\*\*

\*

\*/

package com.hsbc.pack;

/\*\*

\* @author jayesh

\* Create a class called BankAccount with deposit (), withdraw () and getBalance () methods.

\* Create an inner class called InterestAdder and implement the interest calculations and add

\* the interest to the current balance.

\*/

interface InterestAdder {

public void interest(int months);

}

class BankAccount {

String accName;

float balance;

/\*\*

\* @param accName

\* @param balance

\*/

public BankAccount(String accName, float balance) {

super();

this.accName = accName;

this.balance = balance;

}

public void getBalance() {

System.out.println("Updated Balance of user " + this.accName + " : " + this.balance);

InterestAdder ia = new InterestAdder() {

@Override

public void interest(int months) {

// TODO Auto-generated method stub

float i = (float)((months \* balance \* 0.12)/12);

balance += i;

System.out.println("The balance after " + months + " months is " + balance);

}

};

ia.interest(42);

}

public void withdraw(int amount) {

this.balance = this.balance - amount;

}

public void deposit(int amount) {

this.balance = this.balance + amount;

}

}

public class InterestCalculator {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

BankAccount b = new BankAccount("Jayesh", 12000);

b.getBalance();

}

}