**JAVA**

**Inheritance**

**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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* A general example of class inheritanc.

\* \*/

class Vehicle {

public Vehicle() {

super();

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

}

}

class Car extends Vehicle {

public Car() {

super();

}

public void display() {

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

}

}

class Truck extends Vehicle {

public Truck() {

super();

}

public void display() {

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

}

}

class Bus extends Vehicle {

public Bus() {

super();

}

public void display() {

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

}

}

public class Road {

public static void main(String[] args) {

Car c = new Car();

c.display(); //displaying it's type.

System.out.println();

Truck t = new Truck();

t.display(); //displaying it's type.

System.out.println();

Bus b = new Bus();

b.display(); //displaying it's type.

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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* A program to display use of super() method.

\* \*/

class Vehicle1 {

String color;

int wheels;

String model;

public Vehicle1() {

super();

}

public Vehicle1(String color, int noOfWheels, String modelName) {

super();

this.color = color;

this.wheels = noOfWheels;

this.model = modelName;

}

}

class Bike1 extends Vehicle1 {

public Bike1() {

super("Orange",2,"KTM");

}

public void details() {

System.out.println("Bike Name : " + this.model+ " Bike Color : " + this.color+" No. of Wheels : " + this.wheels);

}

}

class Car1 extends Vehicle1 {

public Car1() {

super("Blue",4,"Verna");

}

public void details() {

System.out.println("Car Name : "+this.model+" Car Color : " +this.color+" No. of Wheels : " + this.wheels);

}

}

class Truck1 extends Vehicle1 {

public Truck1() {

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

}

public void details() {

System.out.println("Truck Name : "+this.model+" Truck Color : "+this.color+" No. of Wheels : " + this.wheels);

}

}

public class Solution\_35 {

public static void main(String[] args) {

System.out.println("");

Bike1 b = new Bike1();

b.details();

System.out.println("");

Car1 c = new Car1();

c.details();

System.out.println("");

Truck1 t = new Truck1();

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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* Program to display how protected access specifier is used.

\*

\* \*/

public class Solution\_36 {

public static void main(String[] args) {

Truck3 truck = new Truck3();

Bus3 bus = new Bus3();

Car3 car = new Car3();

Vehicle2 vehicle = new Vehicle2(truck, bus, car );

}

}

class Vehicle2 {

protected String color ;

protected int wheel;

protected String model;

public Vehicle2() {

super();

}

protected Vehicle2(Truck3 truck, Bus3 bus, Car3 car) {

// TODO Auto-generated constructor stub

truck.luggage();

bus.travel();

car.personalUse();

}

protected void display () {

}

protected String getColor() {

return color;

}

protected void setColor(String color) {

this.color = color;

}

protected int getNoOfWheels() {

return wheel;

}

protected void setNoOfWheels(int noOfWheels) {

this.wheel = noOfWheels;

}

protected String getModel() {

return model;

}

protected void setModel(String model) {

this.model = model;

}

protected void move() {

System.out.println("Every Vehicle Moves on the road");

}

}

class Truck3 extends Vehicle2 {

@Override

public String toString() {

return "Truck3 [color=" + color + ", noOfWheels=" + wheel + ", model=" + model + "]";

}

protected void luggage() {

System.out.println("Truck is for goods supply.");

Truck3 truck = new Truck3();

truck.setColor("Black");

truck.setModel("T4");

truck.setNoOfWheels(8);

System.out.println(truck);

}

}

class Bus3 extends Vehicle2 {

@Override

public String toString() {

return "Bus3 [color=" + color + ", noOfWheels=" + wheel + ", model=" + model + "]";

}

protected void travel() {

System.out.println("Bus is public transport.");

Bus3 bus = new Bus3();

bus.setColor("Grey");

bus.setModel("Volvo");

bus.setNoOfWheels(6);

System.out.println(bus);

}

}

class Car3 extends Vehicle2 {

@Override

public String toString() {

return "Car3 [color=" + color + ", noOfWheels=" + wheel + ", model=" + model + "]";

}

protected void personalUse() {

System.out.println("Car is for Personal.");

Car3 car = new Car3();

car.setColor("Black");

car.setModel("Audi Q3");

car.setNoOfWheels(4);

System.out.println(car);

}

}

**Lab Exercise No:**37

**Exercise Objective(s):***Overriding*

**Exercise:** *In the Lab Exercise 30, create a super class called Animal and make all the existing classes as*

*the sub class for Animal class. Move the method isVegetarian and canClimb to the super class*

*and implement generically. Whenever necessary change the implementation of these methods*

*in the respective subclasses.Display the characteristic of each animal.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

**Lab Exercise No:**38

**Exercise Objective(s):***final Keyword*

**Exercise:***In the Lab Exercise 37, make the Lion, Tiger, Deer, Monkey, Elephant and Giraffe classes such*

*that these classes cannot be inherited.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

**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 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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* Program to calculate the pay of workers with help of inheritance.

\* \*/

import java.util.Scanner;

public class Worker {

String name;

double salaryRate;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getSalaryRate() {

return salaryRate;

}

public void setSalaryRate(double salaryRate) {

this.salaryRate = salaryRate;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

DailyWorker dailyWorker = new DailyWorker();

dailyWorker.pay();

SalariedWorker salariedWorker = new SalariedWorker();

salariedWorker.pay();

}

}

class DailyWorker extends Worker {

Scanner scnr = new Scanner(System.in);

void pay () {

System.out.println("Daily Worker Salary");

DailyWorker dailyWorker = new DailyWorker();

System.out.println("Enter total working days : ");

int hrs = scnr.nextInt();

System.out.println("Enter Daily Wage : ");

dailyWorker.setSalaryRate(scnr.nextDouble());

System.out.println("Total Salary is " + (hrs\*dailyWorker.getSalaryRate()));

}

}

class SalariedWorker extends Worker{

Scanner scnr = new Scanner(System.in);

void pay () {

System.out.println("Salaried Worker Salary");

SalariedWorker salariedWorker = new SalariedWorker();

System.out.println("Enter your per hour salary: ");

salariedWorker.setSalaryRate(scnr.nextDouble());

System.out.println("Total Salary is " + (40\*salariedWorker.getSalaryRate()));

}

}

**Lab Exercise No:**40

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

**Exercise:***ModifyLab Exercise 38 by creating intermediate Super classes called WildAnimals and*

*DomesticAnimals and create corresponding subclasses for the same. Create generic*

*methods in the super class and display the details of wild animals, domestic animals in*

*general separately and also display the details of each animal separately.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

**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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* Use of abstract class to find different values over different shape.

\* \*/

import java.util.Scanner;

abstract class Shape {

abstract void volume(double radius,double height);

abstract void surfaceArea(double radius,double height);

}

class Cylinder extends Shape{

void volume(double radius, double height) {

double volume = 3.14\*radius\*radius\*height;

System.out.println("Volume of cylinder is: " + volume);

}

void surfaceArea(double radius,double height) {

double surfaceArea = 2\*3.14\*radius\*(radius+height);

System.out.println("Volume of cylinder is: " + surfaceArea);

}

}

class Sphere extends Shape{

void volume(double radius, double height) {

double volume = (4\*3.14\*radius\*radius\*radius)/3;

System.out.println("Volume of sphere is: " + volume);

}

void surfaceArea(double radius,double height) {

double surfaceArea = 4\*3.14\*radius\*radius;

System.out.println("Volume of sphere is: " + surfaceArea);

}

}

class Cube extends Shape{

void volume(double radius, double height) {

double volume = height\*height\*height;

System.out.println("Volume of cube is: " + volume);

}

void surfaceArea(double radius,double height) {

double surfaceArea = 6\*height\*height;

System.out.println("Volume of cube is: " + surfaceArea);

}

}

class Area {

public static void main(String args[])

{

double h, rad;

Cylinder c = new Cylinder();

Scanner s = new Scanner(System.in);

System.out.print("Enter Base Radius & Vertical Height of Cylinder: ");

rad = s.nextDouble();

h = s.nextDouble();

c.volume(rad, h);

c.surfaceArea(rad, h);

Sphere sp = new Sphere();

System.out.print("Enter Radius of Sphere: ");

rad = s.nextDouble();

sp.volume(rad, 0);

sp.surfaceArea(rad, 0);

Cube cu = new Cube();

System.out.print("Enter Side of a Cube: ");

h = s.nextDouble();

cu.volume(0, h);

cu.surfaceArea(0, h);

}

}

**Lab Exercise No:**42

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

**Exercise:***Create the classes required to store data regarding different types of courses that employees*

*Ina company can enroll for. All courses have name and course fee. Courses are also either*

*classroom delivered or delivered online. Courses could also be full time or part time. The*

*program must be menu based input which enables the course coordinator to register*

*employees for courses, list out employees registered for specific courses, deregister employees*

*from a course.*

**Recommended duration:***30Mins*

**Solution Guidance (if applicable):***The output can be like this,*

*Enter the number of the choice you want to choose:*

1. *To register for a course*
2. *To deregister for a course*
3. *To list the courses offered*
4. *To list the employees registered for a specific course.*

**JAVA**

**Interface**

**Lab Exercise No:**43

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

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

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* Implementing inheritance with Interfaces

\* \*/

import java.util.Scanner;

interface Shape0{

void volume(double radius,double height);

void area(double radius,double height);

}

class Cylinder0 implements Shape0{

@Override

public void volume(double radius, double height) {

// TODO Auto-generated method stub

double vol = 3.14\*radius\*radius\*height;

System.out.println("Volume of Cylinder: "+vol);

}

@Override

public void area(double radius, double height) {

// TODO Auto-generated method stub

double sarea = 2\*3.14\*radius\*(radius+height);

System.out.println("SurfaceArea of Cylinder: "+sarea);

}

}

class Sphere0 implements Shape0{

@Override

public void volume(double radius, double height) {

// TODO Auto-generated method stub

double vol = (4/3)\*3.14\*radius\*radius\*radius;

System.out.println("Volume of Sphere: "+vol);

}

@Override

public void area(double radius, double height) {

// TODO Auto-generated method stub

double sarea = 4\*3.14\*radius\*radius;

System.out.println("Volume of Sphere: "+sarea);

}

}

class Cube0 implements Shape0{

@Override

public void volume(double radius, double height) {

// TODO Auto-generated method stub

double vol = height\*height\*height;

System.out.println("Volume of Cube: "+vol);

}

@Override

public void area(double radius, double height) {

// TODO Auto-generated method stub

double sarea = 6\*height\*height;

System.out.println("Volume of cube is: " + sarea);

}

}

public class Solution43 {

public static void main(String[] args) {

// TODO Auto-generated method stub

double h, rad;

Cylinder0 c = new Cylinder0();

Scanner s = new Scanner(System.in);

System.out.print("Enter Base Radius & Vertical Height of Cylinder: ");

rad = s.nextDouble();

h = s.nextDouble();

c.volume(rad, h);

c.area(rad, h);

Sphere0 sp = new Sphere0();

System.out.print("Enter Radius of Sphere: ");

rad = s.nextDouble();

sp.volume(rad, 0);

sp.area(rad, 0);

Cube0 cu = new Cube0();

System.out.print("Enter Side of a Cube: ");

h = s.nextDouble();

cu.volume(0, h);

cu.area(0, h);

}

}

**Lab Exercise No:**44

**Exercise Objective(s):***Implementingmore 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()*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

package com.hsbc.pack2;

/\*

\* Implementing more than one interface.

\*

\* \*/

public class SolutionDraw {

public static void main(String[] args) {

// TODO Auto-generated method stub

Line l = new Line();

l.drawingColor();

l.fillingColor();

l.size();

l.thickness();

System.out.println();

Circle c = new Circle();

c.drawingColor();

c.fillingColor();

c.size();

c.thickness();

System.out.println();

Square s = new Square();

s.drawingColor();

s.fillingColor();

s.size();

s.thickness();

}

}

interface IDrawable {

void drawingColor();

void thickness();

}

interface IFillable{

void fillingColor();

void size();

}

class Line implements IDrawable, IFillable {

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("FillingColor in Line");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("Size in Line");

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("DrawingColor in Line");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness in Line");

}

}

class Circle implements IDrawable, IFillable {

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("FillingColor in Circle");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("Size in Circle");

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("DrawingColor in Circle");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness in Circle");

}

}

class Square implements IDrawable, IFillable {

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("FillingColor in Square");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("Size in Square");

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("DrawingColor in Square");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness in Square");

}

}

**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() methodthat all the worker classeswillimplement.*

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

package com.hsbc.finance;

/\*

\* Implementation of an interface

\*/

interface Payable {

void getPayInfo();

}

class DailyWorker implements Payable {

String name;

int rate,dailyHour;

public DailyWorker(String name, int rate, int dailyHour) {

this.name = name;

this.rate = rate;

this.dailyHour = dailyHour;

}

@Override

public void getPayInfo() {

// TODO Auto-generated method stub

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

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

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

}

}

class SalariedWorker implements Payable {

String name;

int rate;

public SalariedWorker(String name, int rate) {

this.name = name;

this.rate = rate;

}

@Override

public void getPayInfo() {

// TODO Auto-generated method stub

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

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

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

}

}

public class MonthlyMain {

/\*\*

\* Main Class

\*/

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("......................................");

}

}

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

**Lab Exercise No:**46

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

**Exercise:***Createa package called bank with the following Interfaces.*

*<Interface> Account*

*<Interface>DepositAcc <Interface>LoanAcc*

*<Interface> Interest*

*<Interface>CreditInterest <Interface>DebitInterest*

1. *<Interface> Account*

*Data members: Four String variables to hold the account type “Savings, Fixed,PersonalLoan,*

*HousingLoan”*

*Methods: createAcc()*

1. *<Interface>DepositAcc*

*Methods: withdraw (), deposit(),getBalance()*

1. *<Interface>LoanAcc*

*Methods: repayPrincipal (),payInterest (),payPartialPrincipal ()*

1. *<Interface>Interest*

*Data members: Four double variables to hold the interest percentage of Savings account, Fixed*

*deposit account,PersonalLoan account and HousingLoan account.*

*Methods: calcInt()*

1. *<Interface>CreditInterest*

*Methods: addMonthlyInt(),addHalfYrlyInt(),addAnnualInt()*

1. *<Interface>DebitInterest*

*Methods: deductMonthlyInt(),deductHalfYrlyInt(),deductAnnualInt()*

*Create a package called BankImpl and create the following classes in it.*

1. *SavingsAcc which implements DepositAcc and CreditInterest*
2. *FDAcc which implements DepositAcc and CreditInterest*
3. *PersonalLoanAcc which implements LoanAcc and DebitInterest*
4. *HousingLoanAcc which implements LoanAcc and DebitInterest*

*Now create a class called MyAccount and create instances of all the accounts and generate appropriate output.*

**Recommended duration:***40Mins*

**Solution Guidance (if applicable):** *NA*

**JAVA**

**Inner class**

**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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *Interest calculation for a month*

*(Current balance \* 12%)/12*

package com.hsbc.pack2;

/\*

\* Program to give example ofinner class.

\* \*/

public class BankAccount47 {

float balance;

// Inner class

public class InterestAdder{

// for returning total amount

public float interestAdder()

{ //adding interest to the balance amount

return (balance+(balance\*12/100));

}

}

public float getBalance() {

InterestAdder in = new InterestAdder();

return in.interestAdder();

}

public void setBalance(float balance) {

this.balance = balance;

}

public void deposit(float amount){

this.balance += amount;

}

public void withdraw(float amount){

this.balance -= amount;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

BankAccount47 obj = new BankAccount47();

obj.deposit(500);

obj.withdraw(200);

obj.deposit(700);

// To get total amount = balance+interest amount

System.out.println(obj.getBalance());

}

}

**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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *Interest calculation for a month*

*(Current balance \* 12%)/12*

package com.hsbc.pack2;

/\*

\* Local inner classe.

\* \*/

public class BankAccount48 {

float balance;

public void setBalance(float balance){

this.balance = balance;

}

public float getBalance(){

class Inner{ //defining local inner class to the method getBalance.

public float interestAdder()

{ //adding interest to the balance amount

return (balance+(balance\*12/100));

}

}

Inner inn = new Inner();

this.balance = inn.interestAdder();

return balance;

}

public void deposit(float amount){

this.balance += amount;

}

public void withdraw(float amount){

this.balance -= amount;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

BankAccount48 obj = new BankAccount48();

obj.deposit(500);

System.out.println(obj.getBalance());

}

}

**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.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *Interest calculation for a month*

*(Current balance \* 12%)/12*

package com.hsbc.pack2;

/\*

\* Program containing - Anonymous inner class

\* \*/

abstract class Calculate{

abstract int interest();

}

class BankAccount{

public int balance, interest;

void deposit(int x) {

// adding in balance

balance = balance+x;

}

void withdraw(int x) {

//withdrawing from the balance

balance = balance-x;

}

int getBalance() {

// to get sum of deposit and interest over it

Calculate p=new Calculate(){

int interest()

{

return(((balance)\*12)/(100));

}

};

balance=balance+p.interest();

return balance;

}

public static void main(String[] args)

{

BankAccount b=new BankAccount();

b.deposit(200);

System.out.println(b.getBalance());

}

}