**Rohitkaran Adusumalli**

**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 :**

/\*\*

\* This is a java program to understand the concept of inheritance.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

//Road class generated

public class Road {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Creating objects of various classes.

Truck obj1 = new Truck();

obj1.TruckInfo();

System.out.println();

Bus obj2 = new Bus();

obj2.BusInfo();

System.out.println();

Car obj3 = new Car();

obj3.CarInfo();

System.out.println();

}

}

//Parent class

class Vehicle {

public void NumberOfTyres(int tires){

System.out.println("This vehicle has :" + tires + " tires.");

}

public void NumberOfSeats(int seats) {

System.out.println("This vehicle has :" + seats + " seats.");

}

}

//Subclass Truck extending vehicle

class Truck extends Vehicle {

public Truck(){

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

}

//Specific method of Truck

public void TruckInfo() {

NumberOfTyres(8);

NumberOfSeats(2);

}

}

//Subclass Bus extending vehicle

class Bus extends Vehicle {

public Bus(){

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

}

//Specific method of Bus

public void BusInfo() {

NumberOfTyres(6);

NumberOfSeats(40);

}

}

//Subclass Car extending vehicle

class Car extends Vehicle {

public Car(){

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

}

//Specific method of Car

public void CarInfo() {

NumberOfTyres(4);

NumberOfSeats(5);

}

}

**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 :**

/\*\*

\* This is a java program to understand the concept of inheritance.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

//Road class generated

public class Road {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Creating objects of various classes.

Truck obj1 = new Truck();

obj1.TruckInfo();

System.out.println();

Bus obj2 = new Bus();

obj2.BusInfo();

System.out.println();

Car obj3 = new Car();

obj3.CarInfo();

System.out.println();

}

}

//Parent class

class Vehicle {

private int tires,seats;

public int getTires() {

return tires;

}

public void setTires(int tires) {

this.tires = tires;

}

public int getSeats() {

return seats;

}

public void setSeats(int seats) {

this.seats = seats;

}

public Vehicle() {

super();

}

public void NumberOfTyres(){

System.out.println("This vehicle has :" + this.getTires() + " tires.");

}

public void NumberOfSeats() {

System.out.println("This vehicle has :" + this.getSeats() + " seats.");

}

}

//Subclass Truck extending vehicle

class Truck extends Vehicle {

public Truck(){

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

}

//Specific method of Truck

public void TruckInfo() {

setTires(8);

setSeats(2);

NumberOfTyres();

NumberOfSeats();

}

}

//Subclass Bus extending vehicle

class Bus extends Vehicle {

public Bus(){

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

}

//Specific method of Bus

public void BusInfo() {

setTires(6);

setSeats(40);

NumberOfTyres();

NumberOfSeats();

}

}

//Subclass Car extending vehicle

class Car extends Vehicle {

public Car(){

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

}

//Specific method of Car

public void CarInfo() {

setTires(4);

setSeats(5);

NumberOfTyres();

NumberOfSeats();

}

}

**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 :**

/\*\*

\* This is a java program to understand the concept of inheritance.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

//Road class generated

public class City {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Creating objects of various classes.

Truck obj1 = new Truck();

obj1.display();

System.out.println();

Bus obj2 = new Bus();

obj2.display();

System.out.println();

Car obj3 = new Car();

obj3.display();

}

}

//Parent class

class Vehicle {

private int tires,seats;

protected int getTires() {

return tires;

}

protected void setTires(int tires) {

this.tires = tires;

}

protected int getSeats() {

return seats;

}

protected void setSeats(int seats) {

this.seats = seats;

}

protected Vehicle() {

super();

}

public void display() {

System.out.println("This vehicle has :" + this.getSeats() + " seats.");

System.out.println("This vehicle has :" + this.getTires() + " tires.");

}

}

//Subclass Truck extending vehicle

class Truck extends Vehicle {

protected Truck(){

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

setTires(8);

setSeats(2);

}

}

//Subclass Bus extending vehicle

class Bus extends Vehicle {

public Bus(){

setTires(6);

setSeats(40);

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

}

}

//Subclass Car extending vehicle

class Car extends Vehicle {

public Car(){

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

setTires(4);

setSeats(5);

}

}

**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 :**

/\*\*

\* This is a java program to understand the concept of polymorphism.

\*

\* @author Rohitkaran

\*/

package com.hsbc.demo;

public class ThirtyNineSol {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Taking a salaried and daily worker

DailyWorker work1 = new DailyWorker("Rohitkaran", 100, 52);

int salary = work1.Pay(work1.getNoOfHrsWorked(),work1.getSalaryRate());

System.out.println("The payment of given daily worker is : " + salary);

SalariedWorker work2 = new SalariedWorker("Swathy", 200, 35);

salary = work2.Pay(work2.getNoOfHrsWorked(),work2.getSalaryRate());

System.out.println("The payment of given salaried worker is : " + salary);

}

}

//Parent class

class Worker{

private String name;

private int salaryRate;

private int noOfHrsWorked;

public Worker() {

super();

}

public Worker(String name, int salaryRate, int noOfHrsWorked) {

super();

this.name = name;

this.salaryRate = salaryRate;

this.noOfHrsWorked = noOfHrsWorked;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getSalaryRate() {

return salaryRate;

}

public void setSalaryRate(int salaryRate) {

this.salaryRate = salaryRate;

}

public int getNoOfHrsWorked() {

return noOfHrsWorked;

}

public void setNoOfHrsWorked(int noOfHrsWorked) {

this.noOfHrsWorked = noOfHrsWorked;

}

}

//Subclass of Worker class

class DailyWorker extends Worker{

public DailyWorker(String name, int salaryRate, int noOfHrsWorked) {

super(name,salaryRate,noOfHrsWorked);

// TODO Auto-generated constructor stub

}

//Pay method of DailyWorker class

public int Pay(int noOfHrsWorked,int salaryRate) {

int salary= noOfHrsWorked \* salaryRate;

return salary;

}

}

//Subclass of Worker class

class SalariedWorker extends Worker{

public SalariedWorker(String name, int salaryRate, int noOfHrsWorked) {

super(name,salaryRate,noOfHrsWorked);

// TODO Auto-generated constructor stub

}

//Pay method of SalariedWorker class

public int Pay(int noOfHrsWorked,int salaryRate) {

int salary= 40 \* salaryRate;

return salary;

}

}

**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 :**

/\*\*

\* This is a java program to understand the concept of abstract classes.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

import java.util.Scanner;

public class FourtyOne {

public static void main(String[] args) {

// TODO Auto-generated method stub

double height,radius;

Scanner scan = new Scanner(System.in);

Cylinder ob1 = new Cylinder();

System.out.println("Enter the radius and height of cylinder : ");

height = scan.nextDouble();

radius = scan.nextDouble();

ob1.Volume(height, radius);

ob1.SurfaceArea(height, radius);

Sphere ob2 = new Sphere();

System.out.println("Enter the radius of sphere : ");

radius = scan.nextDouble();

ob2.Volume(0, radius);

ob2.SurfaceArea(0, radius);

Cube ob3 = new Cube();

System.out.println("Enter the side length of the cube : ");

height = scan.nextDouble();

ob3.Volume(height, 0);

ob3.SurfaceArea(height, 0);

scan.close();

}

}

//Creating the abstract class

abstract class Shape3D{

abstract void Volume(double height,double radius);

abstract void SurfaceArea(double height, double radius);

}

//Subclass of abstract class Shape3D

class Cylinder extends Shape3D{

//Defining the abstract class methods

void Volume(double height, double radius) {

double volume = Math.PI \* radius \* radius \* height;

System.out.println("The volume of cyclinder is : " + volume);

}

//Defining the abstract class methods

void SurfaceArea(double height, double radius) {

double surfaceArea = 2 \* Math.PI \* radius \* ( radius \* height);

System.out.println("The surface area of cyclinder is : " + surfaceArea);

}

}

//Subclass of abstract class Shape3D

class Sphere extends Shape3D{

//Defining the abstract class methods

void Volume(double height, double radius) {

double volume = (4 \* Math.PI \* radius \* radius \* radius)/3;

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

}

//Defining the abstract class methods

void SurfaceArea(double height, double radius) {

double surfaceArea = 4 \* Math.PI \* radius \* radius;

System.out.println("The surface area of sphere is : " + surfaceArea);

}

}

//Subclass of abstract class Shape3D

class Cube extends Shape3D{

//Defining the abstract class methods

void Volume(double height, double radius) {

double volume = height \* height \* height;

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

}

//Defining the abstract class methods

void SurfaceArea(double height, double radius) {

double surfaceArea = 6 \* height \* height;

System.out.println("The surface area of cube is : " + surfaceArea);

}

}

**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 :**

/\*\*

\* This is a java program to create a menu for a course coordinator with following options

\* To register for a course

\* To de-register for a course

\* To list the courses offered

\* To list the employees registered for a specific course.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

import java.util.HashMap;

import java.util.HashSet;

import java.util.Scanner;

import java.util.Set;

public class FourtyTwoSol {

static Scanner sc = new Scanner(System.in);

public static void main(String[] args) {

//List of the courses available

Course course1 = new OfflineCourse(1, "AWS", false);

Course course2 = new OfflineCourse(2, "Azure", true);

Course course3 = new OnlineCourse(3, "Angular JS", true);

Course course4 = new OnlineCourse(4, "Vue JS", false);

Course []courses = {course1, course2, course3, course4};

//Employee database in a hashmap

HashMap<Integer, String> employees = new HashMap<Integer, String>();

employees.put(1,"Rohitkaran");

employees.put(2,"Swathy");

employees.put(3,"Saaketh");

boolean exit = false;

while(!exit) {

System.out.println("Enter the number of the choice you want to choose:");

System.out.println("1. To register for a course");

System.out.println("2. To deregister for a course");

System.out.println("3. To list the courses offered");

System.out.println("4. To list employees for specific course\n");

int option = sc.nextInt();

switch(option) {

case 1:

register(courses, employees);

break;

case 2:

deRegister(courses, employees);

break;

case 3:

listCourses(courses);

break;

case 4:

listCourseSpecificEmp(courses, employees);

break;

default:

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

System.out.println();

exit = true;

break;

}

}

}

public static void register(Course courses[], HashMap<Integer, String> employees) {

System.out.println("Enter your Employee ID: ");

int empId = sc.nextInt();

System.out.println();

if(!employees.containsKey(empId)) {

System.out.println("Employee ID does not exist");

System.out.println();

return;

}

System.out.println("Course Available: ");

listCourses(courses);

System.out.println("Enter single Course ID to enroll: ");

int courseId = sc.nextInt();

if(courseId <= 0 || courseId > courses.length) {

System.out.println("Invalid CourseId");

System.out.println();

return;

}

String response = courses[courseId - 1].registerEmployee(empId, employees.get(empId));

System.out.println(response);

}

public static void deRegister(Course courses[], HashMap<Integer, String> employees) {

System.out.println("Enter your Employee ID: ");

int empId = sc.nextInt();

System.out.println();

if(!employees.containsKey(empId)) {

System.out.println("Employee ID does not exist");

System.out.println();

return;

}

System.out.println("Course Available: ");

listCourses(courses);

System.out.println("Enter Course ID to de-register: ");

int courseId = sc.nextInt();

if(courseId <= 0 || courseId > courses.length) {

System.out.println("Invalid CourseId");

System.out.println();

return;

}

String response = courses[courseId - 1].deRegisterEmployee(empId, employees.get(empId));

System.out.println(response);

}

public static void listCourses(Course courses[]) {

for(Course course : courses)

System.out.println(course.getCourseDetails());

System.out.println();

}

public static void listCourseSpecificEmp(Course courses[], HashMap<Integer, String> employees) {

System.out.println("Input the course ID :");

listCourses(courses);

int courseId = sc.nextInt();

if(courseId <= 0 || courseId > courses.length)

return;

Set<Integer> employeesId = courses[courseId - 1].getEmployees();

if (employees.size() == 0) {

System.out.println("No registrations yet!");

return;

}

for(int empId : employeesId) {

System.out.println("Employee[ ID:" + empId + ", " + employees.get(empId) + " ]");

}

System.out.println();

}

}

abstract class Course{

public abstract String getCourseDetails();

public abstract String registerEmployee(int empId, String empName);

public abstract String deRegisterEmployee(int empId, String empName);

public abstract Set<Integer> getEmployees();

}

class OfflineCourse extends Course{

int courseId;

String courseName;

boolean isFullTime;

Set<Integer> employees;

public OfflineCourse(int courseId, String courseName, boolean isFullTime) {

super();

this.courseId = courseId;

this.courseName = courseName;

this.isFullTime = isFullTime;

employees = new HashSet<Integer>();

}

public String getCourseDetails() {

String type = (this.isFullTime) ? "Full Time" : "Part Time";

return "Course[ Offline, " + courseId + ", " + courseName + ", " + type + "]";

}

@Override

public String registerEmployee(int empId, String empName) {

this.employees.add(empId);

return "Employee[" + empId + ", " + empName +"] successfully registered";

}

@Override

public String deRegisterEmployee(int empId, String empName) {

if(!this.employees.contains(empId))

return "Employee should register for the course";

this.employees.remove(empId);

return "Employee[" + empId + ", " + empName +"] successfully de-registered";

}

@Override

public Set<Integer> getEmployees() {

return this.employees;

}

}

class OnlineCourse extends Course{

int courseId;

String courseName;

boolean isFullTime;

Set<Integer> employees;

public OnlineCourse(int courseId, String courseName, boolean isFullTime) {

super();

this.courseId = courseId;

this.courseName = courseName;

this.isFullTime = isFullTime;

employees = new HashSet<Integer>();

}

public String getCourseDetails() {

String type = (this.isFullTime) ? "Full Time" : "Part Time";

return "Course[ Online, " + courseId + ", " + courseName + ", " + type + "]";

}

@Override

public String registerEmployee(int empId, String empName) {

this.employees.add(empId);

return "Employee with Id" + empId + empName +" has been successfully registered";

}

@Override

public String deRegisterEmployee(int empId, String empName) {

if(!this.employees.contains(empId))

return "No such employee found";

this.employees.remove(empId);

return "Employee with Id" + empId + empName +" has been successfully de-registered";

}

@Override

public Set<Integer> getEmployees() {

return this.employees;

}

}

class Employee{

private int empId;

private String empName;

public Employee() {

super();

}

public Employee(int empId, String empName) {

super();

this.empId = empId;

this.empName = empName;

}

public int getEmpId() {

return empId;

}

public String getEmpName() {

return empName;

}

}

**JAVA**

**Interface**

**Lab Exercise No:**43

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

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

**Recommended duration:** *20 Mins*

**Solution :**

/\*\*

\* This is a java program to understand the concept of interfaces.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

import java.util.Scanner;

public class FourtyOne {

public static void main(String[] args) {

// TODO Auto-generated method stub

double height,radius;

Scanner scan = new Scanner(System.in);

Cylinder ob1 = new Cylinder();

System.out.println("Enter the radius and height of cylinder : ");

height = scan.nextDouble();

radius = scan.nextDouble();

ob1.Volume(height, radius);

ob1.SurfaceArea(height, radius);

System.out.println();

Sphere ob2 = new Sphere();

System.out.println("Enter the radius of sphere : ");

radius = scan.nextDouble();

ob2.Volume(0, radius);

ob2.SurfaceArea(0, radius);

System.out.println();

Cube ob3 = new Cube();

System.out.println("Enter the side length of the cube : ");

height = scan.nextDouble();

ob3.Volume(height, 0);

ob3.SurfaceArea(height, 0);

scan.close();

}

}

//Creating the abstract class

interface Shape3D{

void Volume(double height,double radius);

void SurfaceArea(double height, double radius);

}

//Class implementing the interface

class Cylinder implements Shape3D{

//Defining the implemented class methods

@Override

public void Volume(double height, double radius) {

double volume = Math.PI \* radius \* radius \* height;

System.out.println("The volume of cyclinder is : " + volume);

}

//Defining the implemented class methods

@Override

public void SurfaceArea(double height, double radius) {

double surfaceArea = 2 \* Math.PI \* radius \* ( radius \* height);

System.out.println("The surface area of cyclinder is : " + surfaceArea);

}

}

//Class implementing the interface

class Sphere implements Shape3D{

//Defining the implemented class methods

@Override

public void Volume(double height, double radius) {

double volume = (4 \* Math.PI \* radius \* radius \* radius)/3;

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

}

//Defining the implemented class method

@Override

public void SurfaceArea(double height, double radius) {

double surfaceArea = 4 \* Math.PI \* radius \* radius;

System.out.println("The surface area of sphere is : " + surfaceArea);

}

}

//Class implementing the interface

class Cube implements Shape3D{

//Defining the implemented class methods

@Override

public void Volume(double height, double radius) {

double volume = height \* height \* height;

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

}

//Defining the implemented class methods

@Override

public void SurfaceArea(double height, double radius) {

double surfaceArea = 6 \* height \* height;

System.out.println("The surface area of cube is : " + surfaceArea);

}

}

**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()*

**Recommended duration:** *20 Mins*

**Solution :**

/\*\*

\* This is a java program to implement more than one interface.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

public class FourtyFour {

public static void main(String[] args) {

// TODO Auto-generated method stub

Line l1 = new Line();

l1.drawingColor();

l1.thickness();

System.out.println();

Circle c1 = new Circle();

c1.drawingColor();

c1.fillingColor();

System.out.println();

Square s1 = new Square();

s1.drawingColor();

s1.fillingColor();

s1.size();

}

}

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

}

@Override

public void size() {

// TODO Auto-generated method stub

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("drawing color of line");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

System.out.println("Thickness of line");

}

}

class Circle implements IDrawable,IFillable{

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("Filling color of Circle");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("Size of circle");

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("Drawing color of circle");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

}

}

class Square implements IDrawable,IFillable{

@Override

public void fillingColor() {

// TODO Auto-generated method stub

System.out.println("Filling color of square");

}

@Override

public void size() {

// TODO Auto-generated method stub

System.out.println("SIze of square");

}

@Override

public void drawingColor() {

// TODO Auto-generated method stub

System.out.println("drawing color of square");

}

@Override

public void thickness() {

// TODO Auto-generated method stub

}

}

**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.*

**Recommended duration:** *20 Mins*

**Solution :**

//Finance package

package finance;

public interface Payable {

int getPayInfo();

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/\*\*

\* This is a java program to implement and extend an interface and class simultaneously.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

import finance.Payable;

public class ThirtyNineSol {

public static void main(String[] args) {

// TODO Auto-generated method stub

//Taking a salaried and daily worker

DailyWorker work1 = new DailyWorker("Rohitkaran", 100, 52);

int salary = work1.getPayInfo();

System.out.println("The payment of given daily worker is : " + salary);

SalariedWorker work2 = new SalariedWorker("Swathy", 200, 35);

salary = work2.getPayInfo();

System.out.println("The payment of given salaried worker is : " + salary);

}

}

//Parent class

class Worker{

private String name;

private int salaryRate;

private int noOfHrsWorked;

public Worker() {

super();

}

public Worker(String name, int salaryRate, int noOfHrsWorked) {

super();

this.name = name;

this.salaryRate = salaryRate;

this.noOfHrsWorked = noOfHrsWorked;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getSalaryRate() {

return salaryRate;

}

public void setSalaryRate(int salaryRate) {

this.salaryRate = salaryRate;

}

public int getNoOfHrsWorked() {

return noOfHrsWorked;

}

public void setNoOfHrsWorked(int noOfHrsWorked) {

this.noOfHrsWorked = noOfHrsWorked;

}

}

//Subclass of Worker class

class DailyWorker extends Worker implements Payable{

//Overriding getPayInfo

@Override

public int getPayInfo() {

return getNoOfHrsWorked() \* getSalaryRate();

}

public DailyWorker(String name, int salaryRate, int noOfHrsWorked) {

super(name,salaryRate,noOfHrsWorked);

// TODO Auto-generated constructor stub

}

}

//Subclass of Worker class

class SalariedWorker extends Worker implements Payable{

public SalariedWorker(String name, int salaryRate, int noOfHrsWorked) {

super(name,salaryRate,noOfHrsWorked);

// TODO Auto-generated constructor stub

}

//Overriding getPayInfo

@Override

public int getPayInfo() {

return 40 \* getSalaryRate();

}

}

**Lab Exercise No:**46

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

**Exercise:***Create a 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 :**

/\*\*

\* This is a java program to understand the concept of inheritance.

\* *MyAccount has instances of all the accounts and generate appropriate output.*

\*

\* @author Rohitkaran

\*/

Intefaces :

package bank;

interface Account {

String savings = new String();

String fixed = new String();

String housingLoan = new String();

String personalLoan = new String();

void createAcc();

}

interface DepositAcc extends Account {

void deposit();

void withdraw();

void getBalance();

}

interface LoanAcc extends Account {

void repayPrincipal();

void payInterest();

void payPartialPrincipal();

}

interface Interest{

double interestSavAcc = 0;

double interestFixed = 0;

double interestPerLoan = 0;

double interestHousing = 0;

void calcInt();

}

interface CreditInterest extends Interest{

void addMonthlyInt();

void addHalfYrInt();

void addAnnualInt();

}

interface DebitInterest extends Interest{

void deductMonthlyInt();

void deductHalfYrInt();

void deductAnnualInt();

}

class SavingsAcc implements DepositAcc, CreditInterest {

@Override

public void addMonthlyInt() {

// TODO Auto-generated method stub

System.out.println("Monthly Interest added for " + this.savIR + "% is $20000");

}

@Override

public void addHalfYrlyInt() {

// TODO Auto-generated method stub

System.out.println("Half Yearly Interest added for " + this.savIR + "% is $120000");

}

@Override

public void addAnnualInt() {

// TODO Auto-generated method stub

System.out.println("Annually Interest added for " + this.savIR + "% is $240000");

}

@Override

public void withdraw(int amount) {

// TODO Auto-generated method stub

System.out.println(amount + " is withdrawn");

}

@Override

public void deposit(int amount) {

// TODO Auto-generated method stub

System.out.println(amount + " is deposited");

}

@Override

public void getBalance() {

// TODO Auto-generated method stub

System.out.println("The balance of employee is $281000");

}

@Override

public void calcInt() {

// TODO Auto-generated method stub

System.out.println("The interest for " + this.savIR + "% is $350000");

}

@Override

public void createAcc(String name, int id, int bal) {

// TODO Auto-generated method stub

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

System.out.println("Employee ID : " + id);

System.out.println("Employee Balance : " + bal);

System.out.println("Account created successfully");

}

}

class FDAcc implements DepositAcc, CreditInterest {

@Override

public void addMonthlyInt() {

// TODO Auto-generated method stub

System.out.println("Monthly Interest added for " + this.fdIR + "% is $300000");

}

@Override

public void addHalfYrlyInt() {

// TODO Auto-generated method stub

System.out.println("Half Yearly Interest added for " + this.fdIR + "% is $180000");

}

@Override

public void addAnnualInt() {

// TODO Auto-generated method stub

System.out.println("Annually Interest added for " + this.fdIR + "% is $350000");

}

@Override

public void withdraw(int amount) {

// TODO Auto-generated method stub

System.out.println(amount + " is withdrawn");

}

@Override

public void deposit(int amount) {

// TODO Auto-generated method stub

System.out.println(amount + " is deposited");

}

@Override

public void getBalance() {

// TODO Auto-generated method stub

System.out.println("The balance of employee is $303000");

}

@Override

public void calcInt() {

// TODO Auto-generated method stub

System.out.println("The interest for " + this.fdIR + "% is $350000");

}

@Override

public void createAcc(String name, int id, int bal) {

// TODO Auto-generated method stub

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

System.out.println("Employee ID : " + id);

System.out.println("Employee Balance : " + bal);

System.out.println("Account created successfully");

}

}

**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 :**

/\*\*

\* This is a java program to create an inner class.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

public class FourtySevenSol {

public static void main(String[] args) {

// TODO Auto-generated method stub

double finBal;

BankAccount ob1 = new BankAccount();

ob1.deposit(10000);

ob1.withdraw(1000);

finBal = ob1.getBalance();

System.out.println("The final balance is :" + finBal);

}

}

//Outer class

class BankAccount {

static double balance;

//Depositing the amt

void deposit(int x) {

balance+=x;

}

//Withdrawing the amt

void withdraw(int x) {

balance-=x;

}

double getBalance() {

//Inner class object created to use its method

InterestAdder ob = new InterestAdder();

balance = balance + ob.InterestCalc();

return balance;

}

//Inner Class

class InterestAdder{

double InterestCalc(){

double interest = (balance \* 12) / (12 \* 100);

return interest;

}

}

}

**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 :**

/\*\*

\* This is a java program to create an inner class.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

public class FourtySevenSol {

public static void main(String[] args) {

// TODO Auto-generated method stub

double finBal;

BankAccount ob1 = new BankAccount();

ob1.deposit(10000);

ob1.withdraw(1000);

ob1.deposit(10000);

finBal = ob1.getBalance();

System.out.println("The final balance is :" + finBal);

}

}

//Outer class

class BankAccount {

static double balance;

//Depositing the amt

void deposit(int x) {

balance+=x;

}

//Withdrawing the amt

void withdraw(int x) {

balance-=x;

}

double getBalance() {

//Inner Class inside the method of outer class

class InterestAdder{

double InterestCalc(){

double interest = (balance \* 12) / (12 \* 100);

return interest;

}

}

//Inner class object created to use its method

InterestAdder ob = new InterestAdder();

balance = balance + ob.InterestCalc();

return balance;

}

}

**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 :**

/\*\*

\* This is a java program to create an inner class.

\*

\* @author Rohitkaran

\*/

package com.hsbc.day3;

public class FourtySevenSol {

public static void main(String[] args) {

// TODO Auto-generated method stub

double finBal;

BankAccount ob1 = new BankAccount();

ob1.deposit(10000);

ob1.withdraw(1000);

ob1.deposit(10000);

finBal = ob1.getBalance();

System.out.println("The final balance is :" + finBal);

}

}

//Inner Class inside the method of outer class

abstract class InterestAdder{

abstract double InterestCalc();

}

//Outer class

class BankAccount {

static double balance;

//Depositing the amt

void deposit(int x) {

balance+=x;

}

//Withdrawing the amt

void withdraw(int x) {

balance-=x;

}

double getBalance() {

//Creating an anonymous inner class object

InterestAdder ob = new InterestAdder() {

double InterestCalc() {

double interest = (balance \* 12) / (12 \* 100);

return interest;

}

};

balance = balance + ob.InterestCalc();

return balance;

}

}