**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*

**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*

**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*

**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*

**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*

**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*

**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*

**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 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*

**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*

**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*