***Dt : 17/10/2022***

***Assignment:(Solution)***

***Construct BankTransaction application***

***Balance.java***

***package test;***

***public class Balance {***

***public double bal=2000;***

***public double getBalance() {***

***return bal;***

***}***

***}***

***Transaction.java***

***package test;***

***public interface Transaction {***

***public static final Balance b=new Balance();***

***public void process(int amt);***

***}***

***CheckPinNo.java***

***package test;***

***public class CheckPinNo {***

***public boolean verify(int pinNo) {***

***return switch(pinNo) {***

***case 1111 : yield true;***

***case 2222 : yield true;***

***case 3333 : yield true;***

***default : yield false;***

***};***

***}***

***}***

***WithDraw.java***

***package test;***

***public class WithDraw implements Transaction***

***{***

***public void process(int amt)***

***{***

***if(amt<=b.bal)***

***{***

***System.out.println("Amt WithDrawn:"+amt);***

***b.bal=b.bal-amt;***

***System.out.println("Balance amt:"+b.getBalance());***

***System.out.println("Transaction Completed...");***

***}//end of if***

***else***

***{***

***System.out.println("Insufficient fund...");***

***}***

***}***

***}***

***Deposit.java***

***package test;***

***public class Deposit implements Transaction{***

***public void process(int amt)***

***{***

***System.out.println("Amt deposited:"+amt);***

***b.bal=b.bal+amt;***

***System.out.println("Balance amt:"+b.getBalance());***

***System.out.println("Transaction Completed...");***

***}***

***}***

***BankMainClass.java(MainClass)***

***package maccess;***

***import test.\*;***

***import java.util.\*;***

***public class BankMainClass {***

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

***Scanner s = new Scanner(System.in);***

***int count=0;***

***abc:***

***while(true) {***

***System.out.println("Enter the PinNo:");***

***int pinNo = s.nextInt();***

***CheckPinNo cpn = new CheckPinNo();***

***boolean k = cpn.verify(pinNo);***

***if(k)***

***{***

***System.out.println("====Choice====");***

***System.out.println("1.WithDraw\n2.Deposit");***

***System.out.println("Enter the Choice:");***

***switch(s.nextInt())***

***{***

***case 1:***

***System.out.println("Enter the amt:");***

***int a1 = s.nextInt();***

***if(a1>0 && a1%100==0)***

***{***

***WithDraw wd = new WithDraw();***

***wd.process(a1);***

***}//end of if***

***else***

***{***

***System.out.println("Invalid amt...");***

***}***

***break abc;//stop the loop***

***case 2:***

***System.out.println("Enter the amt:");***

***int a2 = s.nextInt();***

***if(a2>0 && a2%100==0)***

***{***

***Deposit dp = new Deposit();***

***dp.process(a2);***

***}//end of if***

***else***

***{***

***System.out.println("Invalid amt...");***

***}***

***break abc;//stop the loop***

***default:***

***System.out.println("Invalid Choice.....");***

***break abc;***

***}//end of switch***

***}//end of if***

***else***

***{***

***System.out.println("Invalid pinNo...");***

***count++;***

***}***

***if(count==3)***

***{***

***System.out.println("Transaction blocked...");***

***break;//stop the loop***

***}***

***}//end of loop***

***}***

***}***

***o/p:***

***Enter the PinNo:***

***1111***

***====Choice====***

***1.WithDraw***

***2.Deposit***

***Enter the Choice:***

***2***

***Enter the amt:***

***12000***

***Amt deposited:12000***

***Balance amt:14000.0***

***Transaction Completed...***

***============================================================***

***faq:***

***wt is the diff b/w***

***(i)Class***

***(ii)AbstractClass***

***=>Class holds only concrete methods,but abstract classes will hold both***

***Concrete methods and abstract methods***

***=>Classes can be instantiated,but abstract classes cannot be instantiated.***

***==================================================================***

***faq:***

***wt is the diff b/w***

***(i)Interface***

***(ii)AbstractClass***

***=>Interface cannot hold 'blocks and constructors',but abstract class can***

***hold 'blocks and constructors'.***

***=>Variables in Interface are automatically 'static and final',but***

***Variables in abstract classes are user choice.***

***=>Programming components in Interfaces are automatically public,but in***

***abstract classes automatically default.***

***=>"abstract" keyword is not manditory to declare abstract methods in***

***Interfaces,but "abstract" keyword is manditory to define abstract methods***

***in abstract classes.***

***=================================================================***

***faq:***

***wt is the diff***

***(i)Class***

***(ii)Interface***

***=>Class can be instantiated,but Interface cannot be instantiated.***

***=>Class can hold 'blocks and constructors',but Interface cannot hold***

***'blocks and constructors'***

***=>Class can hold only concrete methods,but Interface can hold both***

***abstract methods and concrete methods***

***=>Variables in classes are user choice,but Variables in Interfaces are***

***automatically 'static and final'***

***=>Programming components in classes are automatically "default",but***

***in interfaces automatically "public".***

***==================================================================***

***faq:***

***wt is the diff b/w***

***(i)HAS-A relation***

***(ii)IS-A relation***

***(i)HAS-A relation:***

***=>references concept is known as HAS-A relation,because in references***

***one object HAS-A reference of another object.***

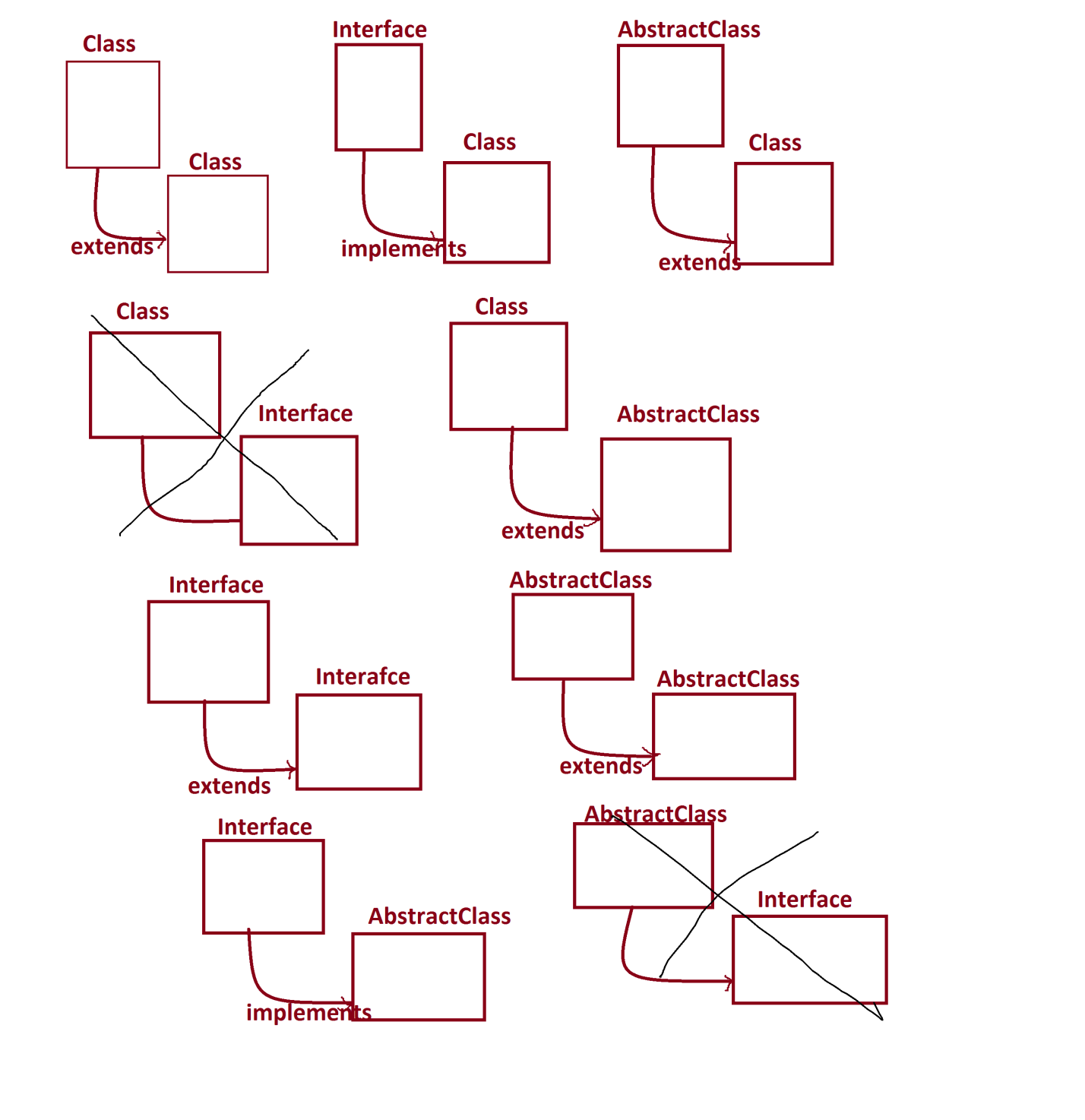
***(ii)IS-A relation:***

***=>Inheritance concept is known as IS-A relation,because CClass is***

***referred by PClass.***

***===================================================================***

***Summary of Single Inheritance Models:***

******

***========================================================================***