3/11/2020

Develop a Java program to create an abstract class named Shape that contains two integers & an empty method named printArea(). provide three classes named Rectangle, Triangle & circle such that each one of these classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given Shape.

```
import java.util.* ;

abstract class Shape {
 int x;

public Shape (int x) {
  this.x = x;
}
 public astract void printArea();
}
class Rectangle extends Shape {
  private int y;

 public Rectangle() {
   this (1, 1);
 }

 public Rectangle(int x, int y) {
   Super (x);
   this.y = y;
 }
 public int getwidth() {
   return Super.x
 }
```

```
public Void SetWidth() {
 Scanner s= new Scanner (System.in);
 int width;

 System.out.println ("Enter width:");
  width = s.nextInt();
  Super.x = width;
 }

 public int getHeight() {
  return this.y;
 }

 public Void Set Height() {
  Scanner s= new Scanner(system.in);
  int height;

  System.out.println("Enter height:");
  height= s.nextInt();
  this.y = height;
 }

 @Override
 public Void printArea() {
  SetWidth().
  setHeight();
  System.out.println("Area: "+(getWidth() *
                                    getHeight()));
 }
}

class circle extends Shape {
 private int y;
```

```
public Circle() {
   this(1);
}

public circle(int x) {
 Super(x);
}

public int getRadius() {
  return Super.x;
}

public void SetRadius() {
  Scanner s= new Scanner(system.in);
  int radius;

  System.out.println("Enter radius.");
  radius = s.nextInt();
  Super.x = radius;
}

@override
public void printArea() {
 SetRadius();
 System.out.println("Area= "+(Math.PI *
              Math.pow(getRadius(), 2)));
}
}

class Triangle extends Shape {
private int y;

public Triangle() {
  this(1,1);
}
```

```
public Triangle(int x, inty) {
 Super (x);
 this.y = y;
}

public int getBase() {
 return Super.x;
}

public void Set Base() {
  scanner S = newScanner(System.in);
  int base;

System.out.println("Enter base)
 base = S.nextInt();
 Super.x = base;
}

public int getHeight() {
  return this.y;
 }

public Void Set Height() {
 Scanner S = new Scanner(System.in);
 int height;

System.out.println("Enter height.");
height = S.nextInt();
this.y = height;
  }

@override
public void PrintArea() {
  SetBase();
```

```
    SetHeight ();
    System.out.println("Area = " +
        ((0.5) * getBase() * getHeight ()));
  }
}

class ShapeArea{
  public static Void main(String[] args){
    shape rect = new Rectangle();
    Shape circle = new Circle();
    Shape Triangle = new Triangle();
    int opt = 0;
    Scanner s= new Scanner(System.in);

   while(opt != 4) {
    System.out.println("Your option are:
     \n1. Rectangle \n2. circle \n3.
     Triangle.");
    opt = s.nextInt();

   If (op== 1)
    rect.printArea();
   else if (opt ==2)
    circle.printArea();
   else if (opt ==3).
    triangle.printArea();
   elseif (opt ==4)
   System.out.println("Terminating
                    Session.");

   else
   System.out.println('Invalid value
                    Passed.");
  }
}
```

2) Develop a Java program to create a class Bank that maintains two kinds of acco for its customers, one called Savings acc & the other current account. The Saving account provides compound interest & withd facilities but no cheque book facility. Th current account provides cheque book facil but no interest. Current account holder should also maintain minimum balance & if it falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number & type of account. From this derive the classes Curr-acct & Sav-acct to make them more specifi to their requirements. Include the necessary methods in order to achieve th following task: Accept deposit from customer & update the balance.

- Display the Balance
- Compute & deposit interest
- permit withdrawal & update the balance
- ~~Display the balance~~ check for minimum balance
- ~~Compute & Deposit interest~~ Impose penalty if necessary & update the balance.

```
import java.util.Scanner;
import java.lang.Math;

class Account
{ String name, type, accno;
 double balance;

 Void deposit()
```

```
{Scanner get=new Scanner(system.in);
 double depo;
System.out.println ("Enter the deposit:");
  depo = get.nextdouble();
  balance= balance + depo;
}

void withdraw()
{scanner get=new Scanner(System.in);
 double withdraw;
   System.out.println ("Enter the amount
   to withdraw: ("<"+balance+")");
withdraw = get.nextDouble();
balance = balance + withdraw;
System.out.println("Balance: "+balance);
}
Void Create()
 {Scanner get= new Scanner(System.in);
  System.out.println("Name:");
   name= get.next();
 type= "Current";
    System.out.println("Account No:");
    accno= get.next();
    System.out.println ("Balance:");
    balance= get.nextDouble();
  }
 Void check()
 { System.out.println("\n Minimum Balance:"
                              + 5000);
 if (balance < 5000)
   {System.out.println ("penalty is imposed
   please deposit minimum" +(5000 -balance
   +200) +"Rs \n Rs 200 Service charge");
   deposit();
```

```
    balance= balance-200;
   }
   else
   { System.out.println("Balance:"+ balance +
     "Safe");}
   }
}

  class sav_acct extends Account
  { double intr=7;
    boolean cheque=false;
    void dispblnc()
    {System.out.println("Balance: "+ balance);
    }
    void create()
    { Scanner get=new Scanner(Sytem.in);
     System.out.println("name : ");
      name= get.next();
     type= "Savings";
     System.out.println("Balance: ");
     balance= get.nextDouble();
   }

  Void calcint()
  { double interest;
   Scanner get= new Scanner(system.in);
  System.out.println("Enter time:");
   int time;
  time= get.nextInt();
  interest= balance * Math.pow(1+ intr/100, time)-balan
   System.out.println("Interest: "+ interest);
   balance= balance + interest;
  System.out.println("Balance: "+ balance);
 }
}
```

```
class Bank
{ public static void main(String args[])
{ Scanner get= new Scanner(System.in);
 String type;
 Sav_acct  accs= new  Sav_acct().
  curr_acct  accr= new  curr_acct();
 System.out.println ("Enter type of account:
        (Current / Saving)");
 type= get.next();
 if (type.equals ("savings"))
  accs.create();
 elseif (type.equals ("current"))
  accr.create();

  int ch;
   do
 {System.out.println("\n1.Deposit \n2. Display
  Balance \n3. Deposit Interest \n4.withdrawl
  \n5. Check \n6. cheque Book (under development)
  \n7.Exit ");
  ch= get.nextInt();
  Switch (ch)
  { case 1:  if  (type.equals("Savings"))
      accs.deposit();
       else
        accr.deposit();
        break.
    case 2: If (type.equals ("saving"))
      accs.dispblnc();
      else
       accr.dispblnc();
        break.
     case 3: if (type.equals ("Savings"))
      accs.calcint();
else
  System.out.println("This account doesn
   have this provision");
  break;
case 4: if(type.equals("Savings"))
  accs.withdraw();
 else
   accr.withdraw();
   break;
case 5: if (type.equals("Savings"))
  System.out.println("This account doesnot
          have this provision");
 else
   accr.check();
    break;
case 6: if (type.equals("Savings"))
  System.out.println("This account doesnot
       have this provision");
 else
  System.out.println("This account does
       have this provision");
  break;
 default: if (ch!7)
   System.out.println("Enter valid option")
 }
} while (ch!=7);
}
}
```