|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNO | DATE | LAB NO | LAB OBJECTIVE | SIGN |
| 01 | 05-March-21 | 01 | Introduction to Java |  |
| 02 | 12-March-21 | 02 | Classes & Objects |  |
| 03 | 02-April-21 | 03 | Access Modifiers |  |
| 04 | 02-April-21 | 04 | Constructors |  |
| 05 | 09-April-21 | 05 | Static Classes & Members |  |
| 06 | 16-April-21 | 06 | Concepts of Overloading |  |
| 07 | 23-April-21 | 07 | Inheritance |  |
| 08 | 05-May-21 | 08 | Polymorphism |  |
| 09 | 20-May-21 | 09 | Association, Aggregation, Composition |  |
| 10 | 12-June-21 | 10 | GUI |  |
| 11 | 19-June-21 | 11 | Abstract Classes, Down Casting & Upcasting |  |
| 12 | 19-June-21 | 12 | Interfaces |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Bahria University,  
Karachi Campus



LAB EXPERIMENT NO.

**05**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Write a program to calculate area of rectangle by using static method. Use parameterized constructor to assign width and height to the instance. Use Output area method which uses the static method to calculate the area. |
| 02 | Write a program to display Name, Enrollment Number, University Name and, Semester of students that are from same university and semester using static fields and methods.(Hint: first set the university name and semester as follows:    Then use static variable counter to get unique roll numbers as follows: |
| 03 | Write a static method called print Times that takes an integer n and a string (in that order) as its parameters and prints the string n times. For example |
| 04 | Write a static method called insult that has two parameters, a String which represents a person’s name and an integer which represents the persons age. This method should create and **return** a String which is a personal insult based on the value of the argument age that was passed. Use the following age cutoffs (or variations of your choosing) for creating your insults: |
| 05 | Write a static method called greet Me that greets you. The method should issue a prompt asking for your name, display a polite (or not so polite) greeting message and then prompt you to enter your age. |
|  |  |

**Task No: 1 Write a program to calculate area of rectangle by using static method. Use parameterized constructor to assign width and height to the instance. Use Output area method which uses the static method to calculate the area.**

**Solution:**

**Class:**

public class Area  
{  
 static int *width*;  
 static int *height*;  
 static int *ans*;  
 Area(int h,int w)  
 {  
 *height* = h;  
 *width* =w;  
 }  
 static void rectangle()  
 {  
 *ans* = *width*\**height*;  
 }  
 void display()  
 {  
 System.*out*.println("\nArea Of Rectangle : " + *ans*);  
 }  
}

**Main:**

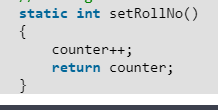
public class Main {  
  
 public static void main(String[] args) {  
 Area obj = new Area(15,30);  
 Area.*rectangle*();  
 obj.display();  
 }  
}

**Output:**

**Task No: 2 Write a program to display Name, Enrollment Number, University Name and, Semester of students that are from same university and semester using static fields and methods.(Hint: first set the university name and semester as follows:**

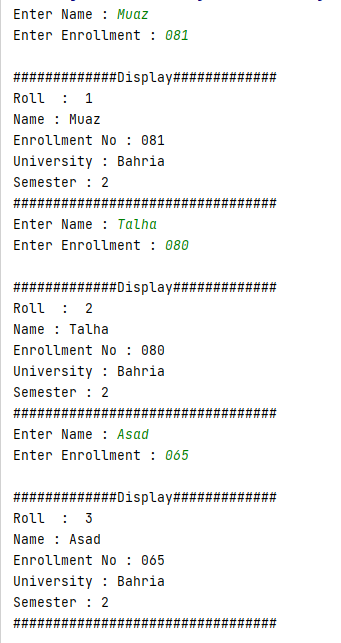
****

**Then use static variable counter to get unique roll numbers as follows:**

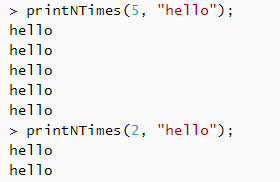


**Solution:  
Class:**package com.company;  
import java.util.Scanner;  
public class StndsRecord  
{  
 static String *Name*;  
 static String *enrollmentNo*;  
 static int *counter*;  
 static String *University* = "Bahria";  
 static String *Semester* = "2";  
 static void setdata()  
 {  
 Scanner input = new Scanner(System.*in*);  
 System.*out*.print("Enter Name : ");  
 *Name* = input.next();  
 System.*out*.print("Enter Enrollment : ");  
 *enrollmentNo* = input.next();  
  
 }  
 static int setroll()  
 {  
 *counter*++;  
 return *counter*;  
 }  
 static void Display()  
 {  
 System.*out*.println("\n#############Display#############");  
 System.*out*.println("Roll : " + *counter*);  
 System.*out*.println("Name : " + *Name*);  
 System.*out*.println("Enrollment No : " + *enrollmentNo*);  
 System.*out*.println("University : " + *University*);  
 System.*out*.println("Semester : " + *Semester*);  
 System.*out*.println("#################################");  
 }  
  
}

**Main:**public class Main {  
  
 public static void main(String[] args)  
 {  
 for (int i=0;i<3;i++)  
 {  
 StndsRecord.*setdata*();  
 StndsRecord.*setroll*();  
 StndsRecord.*Display*();  
  
 }  
 }  
}

**Output:**

**Task No: 3 Write a static method called print Times that takes an integer n and a string (in that order) as its parameters and prints the string n times. For example**



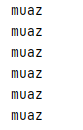
**Solution:**

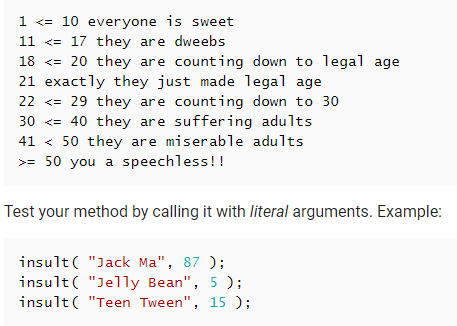
**Class:**

package com.company;  
import java.util.Scanner;  
public class Myclass  
{  
  
 static void PrintN(int a,String m)  
 {  
 Scanner input = new Scanner(System.*in*);  
 for(int j=0;j<a;j++)  
 {  
 System.*out*.println(m);  
 }  
 }  
}

**Main:**

public class Main {  
  
 public static void main(String[] args) {  
 Myclass.*PrintN*(6,"muaz");  
 }  
}

**Output:**

**Task No: 4 Write a static method called insult that has two parameters, a String which represents a person’s name and an integer which represents the persons age. This  
method should create and return a String which is a personal insult based on the value of the argument age that was passed. Use the following age cutoffs (or variations of your choosing) for creating your insults:  
**

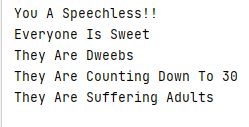
**Solution:**

**Class:**

package com.company;  
  
public class New  
{  
 static String insult(String name,int age)  
 {  
 String a;  
 if(age>=1 && age<= 10) {  
  
 a="Everyone Is Sweet";  
 }  
 else if(age>=11 && age<= 17) {  
  
 a="They Are Dweebs";  
 }  
 else if(age>=18 && age<= 20) {  
  
 a="They Are Counting Down To Legal Age";  
 }  
 else if(age==21){  
  
 a="Exactly They Just Made Legal Age";  
 }  
 else if(age>=22 &&age<= 29) {  
  
 a="They Are Counting Down To 30";  
 }  
 else if(age>=30 && age<= 40) {  
  
 a="They Are Suffering Adults";  
 }  
 else if(age>=41 && age<= 50) {  
  
 a="They Are Miserable Adults";  
 }  
 else if(age>= 50) {  
  
 a="You A Speechless!!";  
 }  
 else{  
 a="";  
 }  
 return a;  
 }  
  
}

**Main:**package com.company;  
public class Main {  
  
 public static void main(String[] args) {  
  
 System.*out*.println(New.*insult*("Asad", 87) );  
 System.*out*.println(New.*insult*("Ali",5) );  
 System.*out*.println(New.*insult*("Kashif",15) );  
 System.*out*.println(New.*insult*("Hamza",25) );  
 System.*out*.println(New.*insult*("Awais",32) );  
  
 }  
}

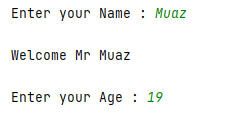
**Output:**



**Task No: 5 Write a static method called greet Me that greets you. The method should issue a prompt asking for your name, display a polite (or not so polite) greeting message and then prompt you to enter your age.**

**Solution:  
Class:**package com.company;  
  
public class Greet  
{  
 static void greet(String name)  
 {  
 System.*out*.println("\nWelcome Mr " + name);  
 }  
  
}

**Main:**package com.company;  
import java.util.Scanner;  
public class Main {  
  
 public static void main(String[] args)  
 {  
 Scanner input = new Scanner (System.*in*);  
 System.*out*.print("Enter your Name : ");  
 String name=input.nextLine();  
 Greet.*greet*(name);  
 System.*out*.print("\nEnter your Age : ");  
 int age=input.nextInt();  
  
 }  
}

**Output:**

Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

06

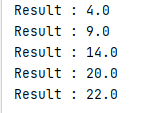
LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Write a program which contains a class ‘Calculator’ contains multiple sum method by using method overloading concept. |
| 02 | Create a class to print the area of a square and a rectangle. The class has two methods with the same name but different number of parameters. The method for printing area of rectangle has two parameters which are length and breadth respectively while the other method for printing area of square has one parameter which is side of square. |
| 03 | Create a class 'Student' with three data members which are name, age and address. The constructor of the class assigns default values name as "unknown", age as '0' and address as "not available". It has two members with the same name 'setInfo'. First method has two parameters for name and age and assigns the same whereas the second method takes has three parameters which are assigned to name, age and address respectively. Print the name, age and address of 4 students. |
| 04 | Implement the Circle class to overload the + operator so that you can add two Circle objects. Adding two Circle object should give another Circle whose radius is the sum of the radii of the two Circle objects. |
| 05 | Implement the Rectangle class to overload the + operator so that you can add two Rectangle objects. Adding two Rectangle objects should give another Rectangle object whose length is the sum of the lengths of the two Rectangle objects and whose breadth is the sum of the breadths of the two Rectangle objects. |
| 06 | Write a class Time which represents time. the class should have three fields for hours, minutes and seconds. It should have constructor to initialize the hours, minutes and seconds. A method printTime() to print the current time. Overload the following operators: plus operator (+) (add two time objects based on 24 hour clock) and < (compare two time objects) |

**Task No: 1** Write a program which contains a class ‘Calculator’ contains multiple sum method by using method overloading concept.

**Solution:  
Class:**package com.company;  
  
public class calculator  
{  
 static double *result*;  
 static void sum(int a,int b)  
 {  
 *result* = (a+b);  
 System.*out*.println("Result : " + *result*);  
 }  
 static void sum(int a,int b,int c)  
 {  
 *result* = (a+b+c);  
 System.*out*.println("Result : " + *result*);  
 }  
 static void sum(int a,int b,int c,int d)  
 {  
 *result* = (a+b+c+d);  
 System.*out*.println("Result : " + *result*);  
 }  
 static void sum(int a,int b,int c,int d,int e)  
 {  
 *result* = (a+b+c+d+e);  
 System.*out*.println("Result : " + *result*);  
 }  
 static void sum(int a,int b,int c,int d,int e,int f)  
 {  
 *result* = (a+b+c+d+e+f);  
 System.*out*.println("Result : " + *result*);  
 }  
}

**Main:**package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 calculator.*sum*(2,2);  
 calculator.*sum*(2,3,4);  
 calculator.*sum*(2,3,4,5);  
 calculator.*sum*(2,3,4,5,6);  
 calculator.*sum*(2,3,4,5,6,2);  
  
  
 }  
}

**Output:**

**Task No: 2** Create a class to print the area of a square and a rectangle. The class has two methods with the same name but different number of parameters. The method for printing area of rectangle has two parameters which are length and breadth respectively while the other method for printing area of square has one parameter which is side of square.

**Solution:  
Class:**package com.company;  
  
public class newclass  
{  
 static double *result*;  
 static void area(int length,int breath)  
 {  
 *result* = length\*breath;  
 System.*out*.println("Area Of Rectangle : " + *result*);  
 }  
 static void area(int sides)  
 {  
 *result* = sides\*sides;  
 System.*out*.println("Area Of Square : " + *result*);  
 }  
}

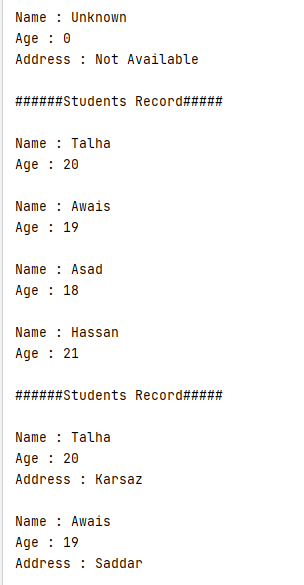
**Main:**package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 newclass.*area*(4,2);  
 newclass.*area*(4);  
 }  
}

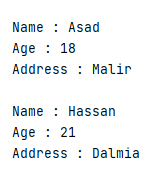
**Output:**

**Task No: 3** Create a class 'Student' with three data members which are name, age and address. The constructor of the class assigns default values name as "unknown", age as '0' and address as "not available". It has two members with the same name 'setInfo'. First method has two parameters for name and age and assigns the same whereas the second method takes has three parameters which are assigned to name, age and address respectively. Print the name, age and address of 4 students.

**Solution:  
Class:**package com.company;  
  
public class Students  
{  
 String Name;  
 int Age;  
 String Address;  
 Students()  
 {  
 Name = "Unknown";  
 Age = 0;  
 Address = "Not Available";  
 System.*out*.println("\nName : " + Name);  
 System.*out*.println("Age : " + Age);  
 System.*out*.println("Address : " + Address);  
  
 }  
 static void setinfo(String Name,int Age)  
 {  
 System.*out*.println("\nName : " + Name);  
 System.*out*.println("Age : " + Age);  
 }  
 static void setinfo(String Name,int Age,String Address)  
 {  
 System.*out*.println("\nName : " + Name);  
 System.*out*.println("Age : " + Age);  
 System.*out*.println("Address : " + Address);  
 }  
}

**Main:**package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 Students obj = new Students();  
 System.*out*.println("\n######Students Record#####");  
 Students.*setinfo*("Talha",20);  
 Students.*setinfo*("Awais",19);  
 Students.*setinfo*("Asad",18);  
 Students.*setinfo*("Hassan",21);  
 System.*out*.println("\n######Students Record#####");  
 Students.*setinfo*("Talha",20,"Karsaz");  
 Students.*setinfo*("Awais",19,"Saddar");  
 Students.*setinfo*("Asad",18,"Malir");  
 Students.*setinfo*("Hassan",21,"Dalmia");  
  
 }  
}

**Output:**



**Task No: 4** Implement the Circle class to overload the + operator so that you can add two Circle objects. Adding two Circle object should give another Circle whose radius is the sum of the radii of the two Circle objects.

**Solution:  
Class:**namespace Circleoperatoroverloading\_L6\_T4

{

class Circle

{

int diameter;

double result;

public Circle()

{

this.diameter = 0;

}

public Circle(int diameter)

{

this.diameter = diameter;

}

public double get()

{

result = diameter / 2;

return result;

}

public static Circle operator +(Circle c1,Circle c2)

{

Circle c3 = new Circle();

c3.diameter = c1.diameter + c2.diameter;

return c3;

}

}

}

**Main:**namespace Circleoperatoroverloading\_L6\_T4

{

class Program

{

static void Main(string[] args)

{

Circle obj1 = new Circle(10);

Circle obj2 = new Circle(12);

Circle obj3 = obj1 + obj2;

Console.WriteLine("\nRadius of Circle1 : " + obj1.get());

Console.WriteLine("Radius of Circle2 : " + obj2.get());

Console.WriteLine("\n------------------------------");

Console.WriteLine("\nAdd Two Circle Radius");

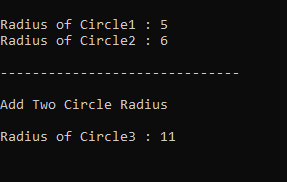
Console.WriteLine("\nRadius of Circle3 : " + obj3.get());

Console.ReadKey();

}

}

}

**Output:**

**Task No: 5** Implement the Rectangle class to overload the + operator so that you can add two Rectangle objects. Adding two Rectangle objects should give another Rectangle object whose length the sum of the lengths of the two Rectangle objects and whose breadth is is the sum of the breadths of the two Rectangle objects.

**Solution:  
Class:**namespace overloading

{

class Rec

{

public int length;

public int breath;

public double result;

public Rec()

{

this.length = 0;

this.breath = 0;

}

public Rec(int length, int breath)

{

this.length = length;

this.breath = breath;

}

public double get()

{

result = length \* breath;

return result;

}

public static Rec operator +(Rec obj1,Rec obj2)

{

Rec obj3 = new Rec();

obj3.length = obj1.length + obj2.length;

obj3.breath = obj1.breath + obj2.breath;

return obj3;

}

}

}

**Main:**namespace overloading

{

class Program

{

static void Main(string[] args)

{

Rec obj1 = new Rec(2, 4);

Rec obj2 = new Rec(4, 4);

Rec obj3 = new Rec();

obj3 = obj1 + obj2;

Console.WriteLine("\n#######RECTANGLE 1#######");

Console.WriteLine("\nLength : " + obj1.length);

Console.WriteLine("Breath : " + obj1.breath);

Console.WriteLine("Area Of Rectangle1 : " + obj1.get());

Console.WriteLine("\n#######RECTANGLE 2#######");

Console.WriteLine("\nLength : " + obj2.length);

Console.WriteLine("Breath : " + obj2.breath);

Console.WriteLine("Area Of Rectangle2 : " + obj2.get());

Console.WriteLine("\n#######RECTANGLE 3#######");

Console.WriteLine("Sum Of Rectangle1 & Rectangle2 ");

Console.WriteLine("\nLength : " + obj3.length);

Console.WriteLine("Breath : " + obj3.breath);

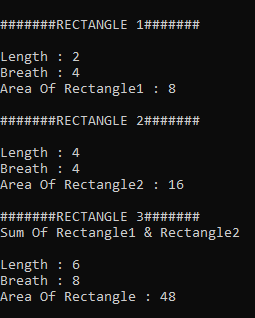
Console.WriteLine("Area Of Rectangle : " + obj3.get());

Console.ReadKey();

}

}

}

**Output:**

**Task No: 6** Write a class Time which represents time. the class should have three fields for hours, minutes and seconds. It should have constructor to initialize the hours, minutes and seconds.  
A method printTime() to print the current time.  
Overload the following operators:  
plus operator (+) (add two time objects based on 24 hour clock)  
and < (compare two time objects)

**Solution:  
Class:**namespace Time\_overloading\_L6\_T6

{

class Time

{

public int hours;

public int minutes;

public int seconds;

public Time()

{

this.hours = 0;

this.minutes = 0;

this.seconds = 0;

}

public Time(int hours,int minutes,int seconds)

{

this.hours = hours;

this.minutes = minutes;

this.seconds = seconds;

}

public static Time operator +(Time t1, Time t2)

{

Time t3 = new Time();

t3.hours = t1.hours + t2.hours;

t3.minutes = t1.minutes + t2.minutes;

t3.seconds = t1.seconds + t2.seconds;

return t3;

}

public static bool operator <(Time t1, Time t2)

{

bool ans;

if (t1.hours < t2.hours && t1.minutes < t2.minutes && t1.seconds < t2.seconds)

{

ans = true;

}

else

{

ans = false;

}

return ans;

}

public static bool operator >(Time t1, Time t2)

{

bool ans;

if (t1.hours > t2.hours && t1.minutes > t2.minutes && t1.seconds > t2.seconds)

{

ans = true;

}

else

{

ans = false;

}

return ans;

}

}

}

**Main:**namespace Time\_overloading\_L6\_T6

{

class Program

{

static void Main(string[] args)

{

Time t1 = new Time(2, 2, 2);

Time t2 = new Time(3, 3, 3);

Time t3 = new Time();

t3 = t1 + t2;

Console.WriteLine("\n#####Time 1######");

Console.WriteLine("\nHours : " + t1.hours);

Console.WriteLine("Minutes : " + t1.minutes);

Console.WriteLine("Seconds : " + t1.seconds);

Console.WriteLine("\n#####Time 2######");

Console.WriteLine("\nHours : " + t2.hours);

Console.WriteLine("Minutes : " + t2.minutes);

Console.WriteLine("Seconds : " + t2.seconds);

Console.WriteLine("\n#####Time 3######");

Console.WriteLine("ADD TIME1 & TIME2");

Console.WriteLine("\nHours : " + t3.hours);

Console.WriteLine("Minutes : " + t3.minutes);

Console.WriteLine("Seconds : " + t3.seconds);

Console.WriteLine("-------------------------------");

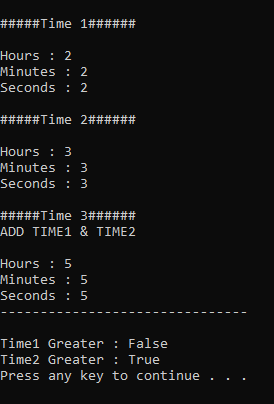
Console.WriteLine("\nTime1 Greater : {0}", t1 > t2);

Console.WriteLine("Time2 Greater : {0}", t1 < t2);

}

}

}

**Output:**

Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**07**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 1 | Write the classes below containing the given instance variables and methods, following the inherited hierarchy: |
| 2 | Write a program that inherits a class named Alien and Pirates from a parent class Human. The human class has its own features like, Human can sleep, walk, talk etc. the Alien and Pirates class inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance. |
| 3 | Write a program that inherits a class named Produce, Cosmetics, Pharmacy, electronic Item and Cloth from a parent class Item. The Item class has its own features like, name and price etc. the Child classes inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance. Chile classes like Produce, can have their own child classes i.e., Frozen and Fresh. |
| 4 | Write a program that inherits a class named Pakistani, BBQ, Chines, Fast Food and Beverages etc. from a parent class Cuisines. The Cuisines class has its own features like, name, quantity and price etc. the Child classes inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance. Child classes can have their own child classes. |
| 5 | Write code according to given guide. You must draw a class diagram first to start writing your code … |

Submitted On:

17-May-21

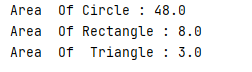
(Date:DD/MM/YY)

Task 1: **Write the classes below containing the given instance variables and methods, following the inherited hierarchy:**

**SOLUTION**

**Main:**

package com.company;  
import java.awt.\*;  
import java.util.Scanner;  
public class Main {  
  
 public static void main(String[] args)  
 {  
 Circle obj = new Circle();  
 obj.setRadius(4);  
 obj.setPi(3);  
 obj.onAreaChange();  
 Rectangle obj1 = new Rectangle();  
 obj1.setHeight(2);  
 obj1.setWidth(4);  
 obj1.getArea();  
 obj1.onAreaChange();  
 Triangle obj2 = new Triangle();  
 obj2.setBase(3);  
 obj2.setAltitude(2);  
 obj2.getArea();  
 obj2.onAreaChange();  
  
 }  
}  
  
**Shapes Class:**package com.company;  
  
class Shapes  
{  
 private double area;  
 Shapes()  
 {  
 area=0;  
 }  
 public void setArea(double area)  
 {  
 this.area = area;  
 }  
 public double getArea()  
 {  
 return area;  
 }  
 public void onAreaChange()  
 {  
 System.*out*.println("Area : " + getArea());  
 }  
}  
  
  
 **Circle:**package com.company;  
  
public class Circle extends Shapes  
{  
 private double radius;  
 private double Pi=3.1416;  
 Circle()  
 {  
  
 }  
 public void setRadius(double radius)  
 {  
 this.radius = radius;  
 }  
 public void setPi(double Pi)  
 {  
 this.Pi = Pi;  
 }  
 @Override  
 public void onAreaChange()  
 {  
 System.*out*.println("Area Of Circle : " + (Pi\*(radius\*radius)));  
 }  
}  
**Triangle:**package com.company;  
  
public class Triangle extends Shapes  
{  
 private double base;  
 private double altitude;  
 Triangle()  
 {  
  
 }  
 public void setBase(double base)  
 {  
 this.base = base;  
 }  
 public void setAltitude(double altitude)  
 {  
 this.altitude = altitude;  
 }  
 public double getArea()  
 {  
 return (base\*altitude)/2;  
 }  
 public void onAreaChange()  
 {  
 System.*out*.println("Area Of Triangle : " + getArea());  
 }  
  
}  
  
 **Rectangle:**package com.company;  
  
public class Rectangle extends Shapes{  
 private double height;  
 private double width;  
 Rectangle()  
 {  
  
 }  
 public void setHeight(double height)  
 {  
 this.height = height;  
 }  
 public void setWidth(double width)  
 {  
 this.width = width;  
 }  
 public double getArea()  
 {  
 return height\*width;  
 }  
 public void onAreaChange()  
 {  
  
 System.*out*.println("Area Of Rectangle : " + getArea());  
 }  
  
  
}

**OUTPUT:**

Task 2: **Write a program that inherits a class named Alien and Pirates from a parent class Human. The human class has its own features like, Human can sleep, walk, talk etc. the Alien and Pirates class inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance.**

**SOLUTION**

**Main:**package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 System.*out*.println();  
 Alien obj = new Alien();  
 Pirates obj1 = new Pirates();  
 obj.Talk();  
 obj.SuperPower();  
 obj.Eat();  
 obj1.swim();  
 obj1.Sleep();  
  
 }  
}

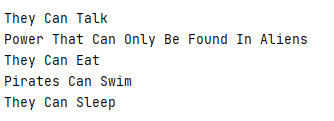
**Human Class:**package com.company;  
  
public class human  
{  
 public void Eat()  
 {  
 System.*out*.println("They Can Eat");  
 }  
 public void Walk()  
 {  
 System.*out*.println("They Can Walk");  
 }  
  
 public void Sleep()  
 {  
 System.*out*.println("They Can Sleep");  
 }  
 public void Talk()  
 {  
 System.*out*.println("They Can Talk");  
 }  
}

**Alien:**

package com.company;  
  
public class Alien extends human  
{  
 public void Fly()  
 {  
 System.*out*.println("They Can Fly");  
 }  
 public void SuperPower()  
 {  
 System.*out*.println("Power That Can Only Be Found In Aliens");  
 }  
  
  
}

**Pirates:**package com.company;  
  
public class Pirates extends human  
{  
 public void swim()  
 {  
 System.*out*.println("Pirates Can Swim");  
 }  
  
}

**OUTPUT:**



Task 3**: Write a program that inherits a class named Produce, Cosmetics, Pharmacy, electronic Item and Cloth from a parent class Item. The Item class has its own features like, name and price etc. the Child classes inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance. Chile classes like Produce, can have their own child classes i.e., Frozen and Fresh.**

**SOLUTION**

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

System.out.println("\*\*Produce\*\*");

Produce Pro = new Produce("Fruits", "Fresh", "29/04/2021");

Pro.name = "Tomatoes";

Pro.brand = "BGS Farms";

Pro.made\_in = "Pakistan";

Pro.TM\_number = 0101;

Pro.price = 1;

Pro.display();

System.out.println("\*\*Cosmetics\*\*");

Cosmetics Cos = new Cosmetics("Facial", "Red", "Liquid", "100g");

Cos.name = "Facial Glow";

Cos.brand = "Victoria's Secrete";

Cos.made\_in = "France";

Cos.TM\_number = 1111;

Cos.price = 400;

Cos.display();

System.out.println("\*\*Pharmacy\*\*");

Pharmacy Pha=new Pharmacy("Antibiotics", "25mg", "Intoxication");

Pha.name = "Aspirin";

Pha.brand = "MGF";

Pha.made\_in = "Germany";

Pha.TM\_number = 1313;

Pha.price = 2;

Pha.TM\_number = 2222;

Pha.display();

System.out.println("\*\*Electronics\*\*");

Electronics Ele = new Electronics(150, "Manual", "No");

Ele.name = "Generator";

Ele.price = 1500;

Ele.brand = "Yamamha";

Ele.made\_in = "Japan";

Ele.TM\_number = 3333;

Ele.display();

System.out.println("\*\*Cloths\*\*");

Cloths Clo = new Cloths("Shirt", "Summer", "Machine or Hand");

Clo.name = "Versache T1s";

Clo.price = 300;

Clo.brand = "Versache";

Clo.TM\_number = 4444;

Clo.made\_in = "Italy";

Clo.display();

}

}  
 **Items Class:**

package javaapplication15;

public class Items {

String name;

int price;

String brand;

int TM\_number;

String made\_in;

Items(){}

public void display() {

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

System.out.println("Price($): " + price);

System.out.println("Brand: " + brand);

System.out.println("Trade Mark: " + TM\_number);

System.out.println("Made in: " + made\_in);

}

}

**Produce Class:**

package javaapplication15;

public class Produce extends Items{

String type;

String condition;

String bestBefore;

public Produce() {}

public Produce(String type,String condition,String bestBefore) {

this.type = type;

this.condition = condition;

this.bestBefore = bestBefore;

}

public void display() {

super.display();

System.out.println("Type: " + type);

System.out.println("Condition: " + condition);

System.out.println("Exp. Date: " + bestBefore);

System.out.println();

}

}

**Cosmetics Class:**

package javaapplication15;

public class Cosmetics extends Items {

String type;

String colour;

String state;

String weight;

Cosmetics() {}

public Cosmetics(String type, String color, String state, String weight) {

this.type = type;

this.colour = color;

this.state = state;

this.weight = weight;

}

public void display() {

super.display();

System.out.println("Type: " + type);

System.out.println("Colour: " + colour);

System.out.println("State: " + state);

System.out.println("Weight: " + weight);

System.out.println();

}

}

**Pharmacy Class:**

package javaapplication15;

public class Pharmacy extends Items {

String purpose;

String mg;

String effect;

Pharmacy() {}

public Pharmacy(String purpose, String mg, String effect) {

this.purpose = purpose;

this.effect = effect;

this.mg = mg;

}

public void display() {

super.display();

System.out.println("Purpose: " + purpose);

System.out.println("Effect: " + effect);

System.out.println("mg: " + mg);

System.out.println();

}

} **Electronics Class:**

package javaapplication15;

public class Electronics extends Items {

int volts;

String type;

String waterproof;

public Electronics() {}

public Electronics(int volts, String type, String waterproof) {

this.volts = volts;

this.type = type;

this.waterproof = waterproof;

}

public void display() {

super.display();

System.out.println("Voltage: " + volts);

System.out.println("Type: " + type);

System.out.println("Water proof: " + waterproof);

System.out.println();

}

}

**Cloths Class:**

package javaapplication15;

public class Cloths extends Items {

String type;

String weather;

String hand\_Wash;

public Cloths() {}

public Cloths(String type, String weather, String wash) {

this.type = type;

this.weather = weather;

this.hand\_Wash = wash;

}

public void display() {

super.display();

System.out.println("Type: " + type);

System.out.println("Collection: " + weather);

System.out.println("Machine or Hand wash: " + hand\_Wash);

System.out.println();

}

}

**Output:**

****

Task 4: **Write a program that inherits a class named Pakistani, BBQ, Chines, Fast Food and Beverages etc. from a parent class Cuisines. The Cuisines class has its own features like, name, quantity and price etc. the Child classes inheriting these functionalities as well as they have their characteristics, thus explaining the concepts of inheritance. Child classes can have their own child classes.**

**SOLUTION**

**Main:**package com.company;  
  
public class Main {  
  
 public static void main(String[] args) {  
  
 BBQ obj1=new BBQ();  
 obj1.order();  
 obj1.item();  
 Chinese obj2=new Chinese();  
 System.*out*.println(obj2.a);  
 obj2.menu();  
 FastFood obj3=new FastFood();  
 obj3.menu();  
 Beverages obj4=new Beverages();  
 obj4.menu();  
  
 }  
}

**Cuisines Class:**package com.company;  
  
public class Cuisines  
{  
 String name;  
 int quantity;  
 double price;  
  
 void order()  
 {  
 System.*out*.println("Take Order");  
 }  
 void Take\_Away()  
 {  
 System.*out*.println("Deliver Food");  
 }  
 void Dine\_In()  
 {  
 System.*out*.println("Serve Food");  
 }  
  
}

**Beverages Class:**

package com.company;  
  
public class Beverages extends Cuisines  
{  
 String Soft\_Drinks;  
 public void menu()  
 {  
 System.*out*.println("Soft Drinks Are Available");  
 }  
  
}

**Fast Food Class:**

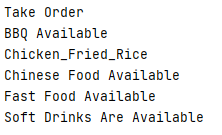
package com.company;  
  
public class FastFood extends Cuisines{  
 String Zinger\_Burger;  
 String Broast;  
 String Sandwich;  
 public void menu()  
 {  
 System.*out*.println("Fast Food Available");  
 }  
}

**BBQ Class:**

package com.company;  
  
public class BBQ extends Cuisines  
{  
 String Tikka;  
 String Chicken\_Boti;  
 String Seekh\_Kabab;  
 public void item()  
 {  
 System.*out*.println("BBQ Available");  
 }  
  
}

**Chinese Class:**

package com.company;  
  
public class Chinese extends Cuisines  
{  
 String a = "Chicken\_Fried\_Rice";  
 String Chicken\_Chilli;  
 String Chicken\_Black\_Pepper;  
  
 public void menu()  
 {  
 System.*out*.println("Chinese Food Available");  
 }  
  
}

**OUTPUT:**

Task 5: **Write code according to given guide. You must draw a class diagram first to start writing your code …**

**SOLUTION**

**Main:**

package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 System.*out*.println("Items Class");  
 Item Obj = new Item("Fan", 14999.8);  
 System.*out*.println(Obj.toString());  
 System.*out*.println("Weighted Item Class");  
 Weighted\_item Obj1 = new Weighted\_item ("Banana", 8, 1.5);  
 System.*out*.println(Obj1.toString());  
 System.*out*.println("Counted Item Class");  
 Counted\_item Obj2 = new Counted\_item("Pens", 10, 5);  
 System.*out*.println(Obj2.toString());  
  
  
 }  
}

**Item Class:**

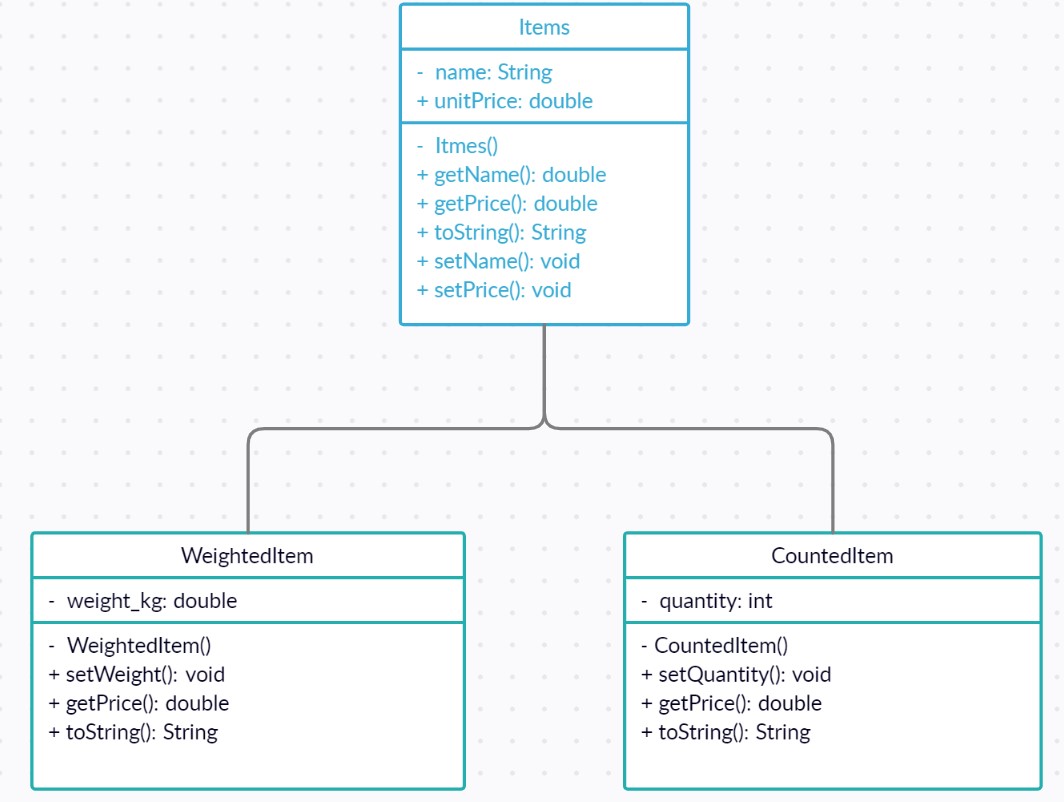
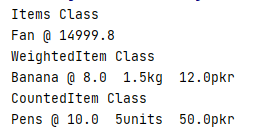
package com.company;  
  
public class Item  
{  
 private String name;  
 private double unitPrice;  
 Item() {  
 this.name = "no item";  
 this.unitPrice = 0;  
 }  
 Item(String name, double unitPrice) {  
 this.name = name;  
 this.unitPrice = unitPrice;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public void setPrice(double unitPrice) {  
 this.unitPrice = unitPrice;  
 }  
 public String getName() {  
 return this.name;  
 }  
 public double getPrice() {  
 return this.unitPrice;  
 }  
 @Override  
 public String toString() {  
 return (this.name + " @ " + this.unitPrice);  
 }  
  
}

**Weighted Item:**

package com.company;  
  
public class Weighted\_item extends Item{  
 private double weight\_kg;  
 Weighted\_item(String name, double unitPrice, double weight\_kg) {  
 super(name, unitPrice);  
 this.weight\_kg = weight\_kg;  
 }  
 public void setWeight(double weight\_kg) {  
 this.weight\_kg = weight\_kg;  
 }  
 @Override  
 public double getPrice() {  
 return super.getPrice() \* weight\_kg;  
 }  
 @Override  
 public String toString() {  
 return (super.getName() + " @ " + super.getPrice() + " " + weight\_kg + "kg " + getPrice() + "pkr");  
 }  
  
}

**Counted Item:**

package com.company;  
  
public class Counted\_item extends Item{  
 private int quantity;  
 Counted\_item(String name, double unitPrice, int quantity) {  
 super(name, unitPrice);  
 this.quantity = quantity;  
 }  
 public void setQuantity(int quantity) {  
 this.quantity = quantity;  
 }  
 @Override  
 public double getPrice() {  
 return super.getPrice() \* quantity;  
 }  
 @Override  
 public String toString() {  
 return (super.getName() + " @ " + super.getPrice() + " " + quantity + "units " + getPrice() + "pkr");  
 }  
  
  
  
  
}

**OUTPUT:**

Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**08**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Create a payroll system using **classes**, **inheritance,** and **polymorphism.**  Four types of employees paid weekly.   1. **Salaried employees**: fixed salary irrespective of hours 2. **Hourly employees**: 40 hours salary and overtime (> 40 hours) 3. **Commission employees**: paid by a percentage of sales. 4. **Base-plus-commission employees**: base salary and a percentage of sales   The information know about each employee is his/her first name, last name, and national identity card number. The reset depends on the type of employee. |
| 02 | You must implement the following diagram including some attributes and other functions: |

Submitted On:

17-May-21

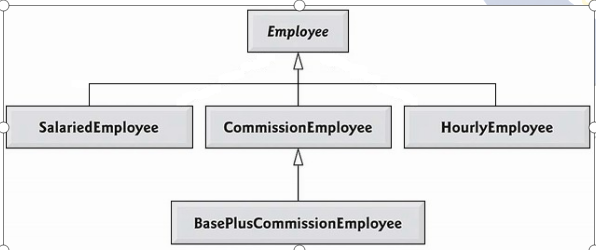
(Date: DD/MM/YY)

**Task 1:** Create a payroll system using **classes**, **inheritance,** and **polymorphism.**

Four types of employees paid weekly.

1. **Salaried employees**: fixed salary irrespective of hours
2. **Hourly employees**: 40 hours salary and overtime (> 40 hours)
3. **Commission employees**: paid by a percentage of sales.
4. **Base-plus-commission employees**: base salary and a percentage of sales

The information know about each employee is his/her first name, last name, and national identity card number. The reset depends on the type of employee.



**Solution  
Employ Class:**

package com.company;  
  
public class Employee  
{  
 private String Firstname;  
 private String Lastname;  
 private String CNIC;  
 Employee()  
 {  
 this.Firstname = "No Name";  
 this.Lastname = "No Name";  
 this.CNIC = "No CNIC";  
 }  
 Employee(String Firstname,String Lastname,String CNIC)  
 {  
 this.Firstname = Firstname;  
 this.Lastname = Lastname;  
 this.CNIC = CNIC;  
 }  
 public void setFirstname(String Firstname)  
 {  
 this.Firstname = Firstname;  
 }  
 public void setLastname(String Lastname)  
 {  
 this.Lastname = Lastname;  
 }  
 public void setCNIC(String CNIC)  
 {  
 this.CNIC = CNIC;  
 }  
 public String getFirstname()  
 {  
 return Firstname;  
 }  
 public String getLastname()  
 {  
 return Lastname;  
 }  
 public String getCNIC()  
 {  
 return CNIC;  
 }  
  
 public String toString()  
 {  
 return "\nFirstName : " + this.Firstname + "\nLastName : " + this.Lastname + "\nCNIC : " + this.CNIC;  
 }  
 public double Earning()  
 {  
 return 0.00;  
 }  
}

**Salaried Class:**

package com.company;  
  
public class Salaried extends Employee  
{  
 private double Weeklysalary;  
 Salaried()  
 {  
 super();  
 this.Weeklysalary = 0;  
 }  
 Salaried(String Firstname,String Lastname,String CNIC,double Weeklysalary)  
 {  
 super(Firstname,Lastname,CNIC);  
 this.Weeklysalary = Weeklysalary;  
 }  
 public void setWeeklysalary(double Weeklysalary)  
 {  
 if(Weeklysalary > 0);  
 this.Weeklysalary = Weeklysalary;  
 }  
 public double getWeeklysalary()  
 {  
 return Weeklysalary;  
 }  
  
  
 public String toString()  
 {  
 return "# Salaried Employee #" + super.toString();  
 }  
 public double Earning()  
 {  
 return Weeklysalary;  
 }  
}

**Hourly Employee Class:**

package com.company;  
  
public class Hourly\_Employee extends Employee  
{  
 private double hours;  
 private double wage;  
 Hourly\_Employee()  
 {  
 this.hours = 0;  
 this.wage = 0;  
 }  
 Hourly\_Employee(String Firstname,String Lastname,String CNIC,double hours,double wage)  
 {  
 super(Firstname,Lastname,CNIC);  
 this.hours = hours;  
 this.wage = wage;  
 }  
 public void setHours(double hours)  
 {  
 if(hours > 0);  
 {  
 this.hours = hours;  
 }  
  
 }  
 public void setWage(double wage)  
 {  
 if(wage > 0);  
 {  
 this.wage = wage;  
 }  
  
 }  
 public double getHours()  
 {  
 return hours;  
 }  
 public double getWage()  
 {  
 return wage;  
 }  
 public String toString()  
 {  
 return "\n# Hourly Employee #" + super.toString() + "\nWage = " + wage + "\nHours = " + hours;  
 }  
 public double Earning()  
 {  
 if(hours <=40)  
 {  
 return wage\*hours;  
 }  
 else  
 return 40\*wage+(hours-40)\*wage\*1.5;  
 }  
}

**Commission Employ Class:**

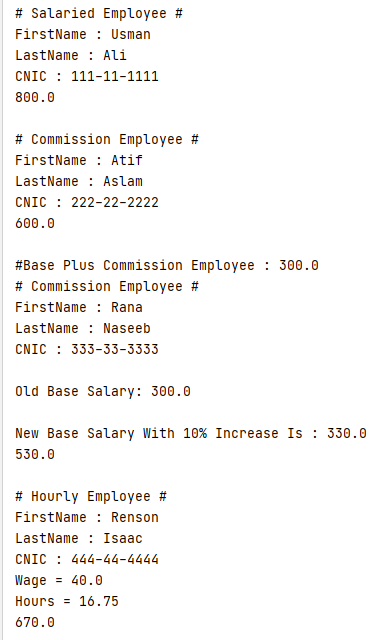
package com.company;  
  
public class Comission\_Employee extends Employee  
{  
 private double Gross\_Sales;  
 private double Commission\_Rate;  
 Comission\_Employee()  
 {  
 this.Gross\_Sales = 0;  
 this.Commission\_Rate = 0;  
 }  
 Comission\_Employee(String Firstname,String Lastname,String CNIC,double Gross\_Sales,double Commission\_Rate)  
 {  
 super(Firstname,Lastname,CNIC);  
 this.Gross\_Sales = Gross\_Sales;  
 this.Commission\_Rate = Commission\_Rate;  
 }  
 public void setGross\_Sales(double Gross\_Sales)  
 {  
 if(Gross\_Sales > 0)  
 {  
 this.Gross\_Sales = Gross\_Sales;  
 }  
 }  
 public void setCommission\_Rate(double Commission\_Rate)  
 {  
 if(Commission\_Rate > 0)  
 {  
 this.Commission\_Rate = Commission\_Rate;  
 }  
 }  
 public double getGross\_Sales()  
 {  
 return Gross\_Sales;  
 }  
 public double getCommission\_Rate()  
 {  
 return Commission\_Rate;  
 }  
  
 public String toString()  
 {  
 return "\n# Commission Employee #" + super.toString();  
 }  
 public double Earning( ) {  
 return Gross\_Sales \* Commission\_Rate;  
 }  
  
}

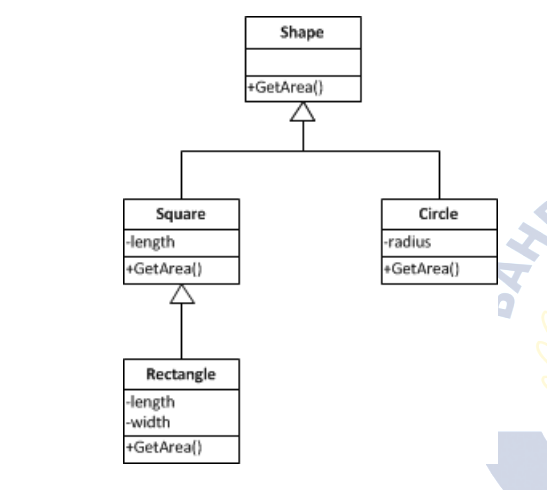
**Base Plus Commission Class:**

package com.company;  
  
public class BasePlus\_Commission extends Comission\_Employee  
{  
 private double baseSalary;  
 BasePlus\_Commission()  
 {  
 this.baseSalary = 0;  
 }  
 BasePlus\_Commission(String Firstname,String Lastname,String CNIC,double Gross\_Sales,double Commission\_Rate,double BaseSalary)  
 {  
 super(Firstname,Lastname,CNIC,Gross\_Sales,Commission\_Rate);  
 this.baseSalary = BaseSalary;  
 }  
 public void setBaseSalary(double baseSalary)  
 {  
 if(baseSalary>0)  
 {  
 this.baseSalary = baseSalary;  
 }  
 }  
 public double getBaseSalary()  
 {  
 return baseSalary;  
 }  
 @Override  
 public double Earning( ) {  
 return getBaseSalary() + super.Earning();  
 }  
 @Override  
 public String toString( )  
 {  
 return "\n#Base Plus Commission Employee : " +getBaseSalary() + super.toString();  
 }  
  
  
}

**Main:**

package com.company;  
  
public class Main {  
  
 public static void main(String[] args)  
 {  
 Employee FirstEmployee = new Salaried("Usman" ,"Ali","111-11-1111", 800.00 );  
 Employee SecondEmployee = new Comission\_Employee("Atif" ,"Aslam", "222-22-2222", 10000, 0.06 );  
 Employee ThirdEmployee = new BasePlus\_Commission("Rana", "Naseeb", "333-33-3333", 5000 , 0.04 , 300 );  
 Employee FourthEmployee = new Hourly\_Employee( "Renson" , "Isaac", "444-44-4444" , 16.75 , 40 );  
 *// polymorphism: calling toString() and earning() on Employee’s reference* System.*out*.println(FirstEmployee);  
 System.*out*.println(FirstEmployee.Earning());  
 System.*out*.println(SecondEmployee);  
 System.*out*.println(SecondEmployee.Earning());  
 System.*out*.println(ThirdEmployee);  
 *// performing downcasting to access & raise base salary* BasePlus\_Commission CurrentEmployee = (BasePlus\_Commission) ThirdEmployee;  
 double OldBaseSalary = CurrentEmployee.getBaseSalary();  
 System.*out*.println( "\nOld Base Salary: " + OldBaseSalary) ;  
 CurrentEmployee.setBaseSalary(1.10 \* OldBaseSalary);  
 System.*out*.println("\nNew Base Salary With 10% Increase Is : " + CurrentEmployee.getBaseSalary());  
 System.*out*.println(ThirdEmployee.Earning() );  
 System.*out*.println(FourthEmployee);  
 System.*out*.println(FourthEmployee.Earning());  
  
  
  
  
  
  
 }  
}

**Output:** **Task 2:** You must implement the following diagram including some attributes and other functions:



**Solution  
Shapes Class:**package com.company;  
  
public class Shapes  
{  
 private double area;  
 public void setArea(double area)  
 {  
 this.area = area;  
 }  
 public double GetArea()  
 {  
 return area;  
 }  
  
}

**Circle Class:**package com.company;  
  
public class Circle extends Shapes  
{  
 private double radius;  
 public void setRadius(double radius)  
 {  
 this.radius=radius;  
 }  
 public double getRadius() {  
 return radius;  
 }  
 public double getarea()  
 {  
 double area=3.14\*radius\*radius;  
 return area;  
 }  
 public String toString()  
 {  
 return "Area Of Circle : " + getarea();  
 }  
  
  
}

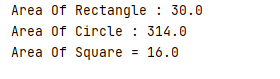
**Square Class:**

package com.company;  
  
public class Square extends Shapes  
{  
 private double length;  
 public Square(double length)  
 {  
  
 this.length=length;  
 }  
  
 public double getLength() {  
 return length;  
 }  
 public void setLength(double length) {  
 this.length=length;  
 }  
 public double getarea()  
 {  
 double area=length\*length;  
 return area;  
 }  
 public String toString()  
 {  
 return "Area Of Square = " +getarea();  
 }  
  
}

**Rectangle Class:**

package com.company;  
  
public class Rectangle extends Square  
{  
 private double width;  
 public Rectangle(double length,double width)  
 {  
  
 super(length);  
 this.width=width;  
 }  
  
 public void setWidth(int width) {  
 this.width = width;  
 }  
  
  
 public double getWidth() {  
 return width;  
 }  
 public double getarea()  
 {  
 double area=getLength()\*width;  
 return area;  
 }  
  
 @Override  
 public String toString() {  
 super.toString();  
 return "Area Of Rectangle : " + getarea();  
 }  
  
  
}

**Main:**package com.company;  
import java.util.Scanner;  
public class Main {  
  
 public static void main(String[] args)  
 {  
 Scanner input = new Scanner(System.*in*);  
 Circle obj = new Circle();  
 obj.setRadius(10);  
 Rectangle obj1 = new Rectangle(10, 3);  
 System.*out*.println(obj1.toString()) ;  
 System.*out*.println(obj.toString());  
 Square obj2 = new Square(4);  
 System.*out*.println(obj2.toString());  
  
 }  
}

**Output:**Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

09

LIST OF TASKS

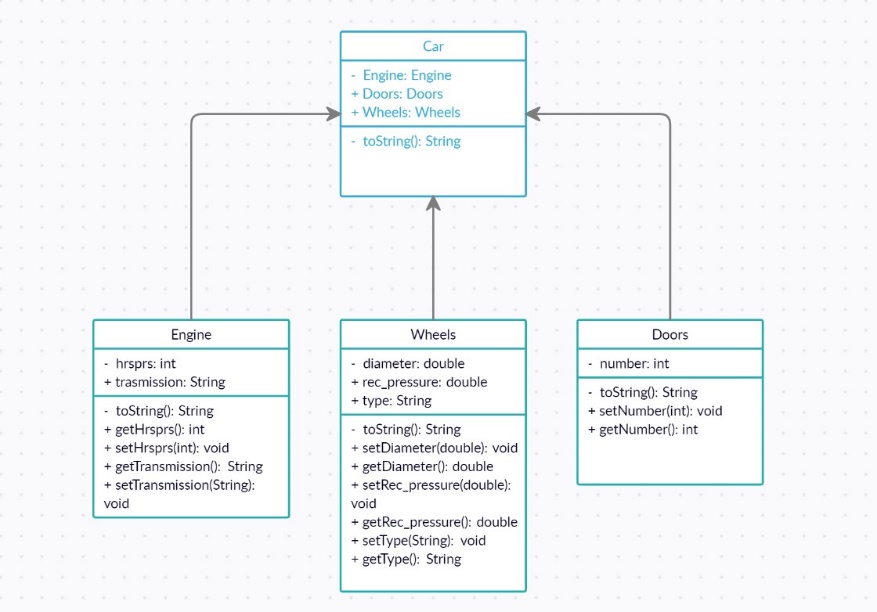
|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Write a program and create the objects of classes in class car to explain the concept of composition. Create several classes as engine, doors, capacity and wheel having their individual methods attributes. The object of these classes are created in a car class and they are set as public. The object of this car class is created in Main method and this with the help of this object we can call other classes as well and can use their functionalities and design UML class diagram. |
| 02 | Write complete program for Flight's class, Time's class and Passenger's class with the concept of association and aggregation and design UML class diagram. Functions information also been given in the table below:   |  |  | | --- | --- | | **Method** | **Description** | | addPassenger(Passenger) | This method will add Passenger's object to vector passengerList. | | printInfo() | This method will display all flight information namely ID (Flight number), destination, departure time, arrival time and number of passengers.  For Example:  Flight no : PK-303  Destination : Lahore  Departure : 8:10  Arrival : 9:00  Number of passenger :2 | | getHour() | This method will return the value of attribute **hour** | | getMinute() | This method will return the value of attribute **minute** | |
| 03 | A company manages many stores. Each Store contains many Products. Implement Product, Store and Company classes using association and aggrigation concepts and design UML class diagram. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Submitted On:

05/06/2021

(Date: DD/MM/YY)

**Task 1: Write a program and create the objects of classes in class car to explain the concept of composition. Create several classes as engine, doors, capacity and wheel having their individual methods attributes. The object of these classes are created in a car class and they are set as public. The object of this car class is created in Main method and this with the help of this object we can call other classes as well and can use their functionalities and design UML class diagram.**

**Solution** **Main**

**package** com.company;  
  
**public class** Main {  
  
 **public static void** main(String[] args)  
 {  
 Car obj = **new** Car(**"Ferrari"**);  
 obj.**engine**.engine(**"123456"**);  
 obj.**doors**.Door(**"Automatic"**);  
 obj.**capacity**.Capaciti(**"5"**);  
 obj.**wheel**.Wheels(**"Racing Wheel"**);  
 System.***out***.println(**"\nName Engine\_Num Doors Capacity Wheels"**);  
  
  
 System.***out***.println(obj.getCar\_Name()+**"\t\t"**+ obj.**engine**+**"\t\t"**+obj.**doors**+**"\t\t"**+obj.**capacity**+**"\t\t\t"**+obj.**wheel**);  
  
  
 }  
}

**Car**

**package** com.company;  
  
**public class** Car  
{  
 **public** String **Car\_Name**;  
 **public final** Engine **engine**;  
 **public final** Doors **doors**;  
 **public final** Capacity **capacity**;  
 **public final** Wheel **wheel**;  
 **public** Car(String Car\_Name)  
 {  
 **engine** = **new** Engine();  
 **doors** = **new** Doors();  
 **capacity** = **new** Capacity();  
 **wheel** = **new** Wheel();  
 **this**.**Car\_Name** = Car\_Name;  
 }  
 **public** String getCar\_Name()  
 {  
 **return Car\_Name**;  
 }  
  
  
}

**Engine**

**package** com.company;  
  
**public class** Engine  
{  
 **public** String **Engine\_num**;  
  
 **public** Engine()  
 {  
 **this**.**Engine\_num** = **"0"**;  
 }  
 **public void** engine(String Engine\_num)  
 {  
 **this**.**Engine\_num** = Engine\_num;  
 }  
 **public** String toString()  
 {  
 **return Engine\_num**;  
 }  
}

**Doors**

**package** com.company;  
  
**public class** Doors  
{  
 **public** String **Door\_Type**;  
  
 **public** Doors()  
 {  
 **this**.**Door\_Type** = **"No type"**;  
 }  
 **public void** Door(String Door\_Type)  
 {  
 **this**.**Door\_Type** = Door\_Type;  
 }  
 **public** String toString()  
 {  
 **return Door\_Type**;  
 }  
  
  
}

**Capacity**

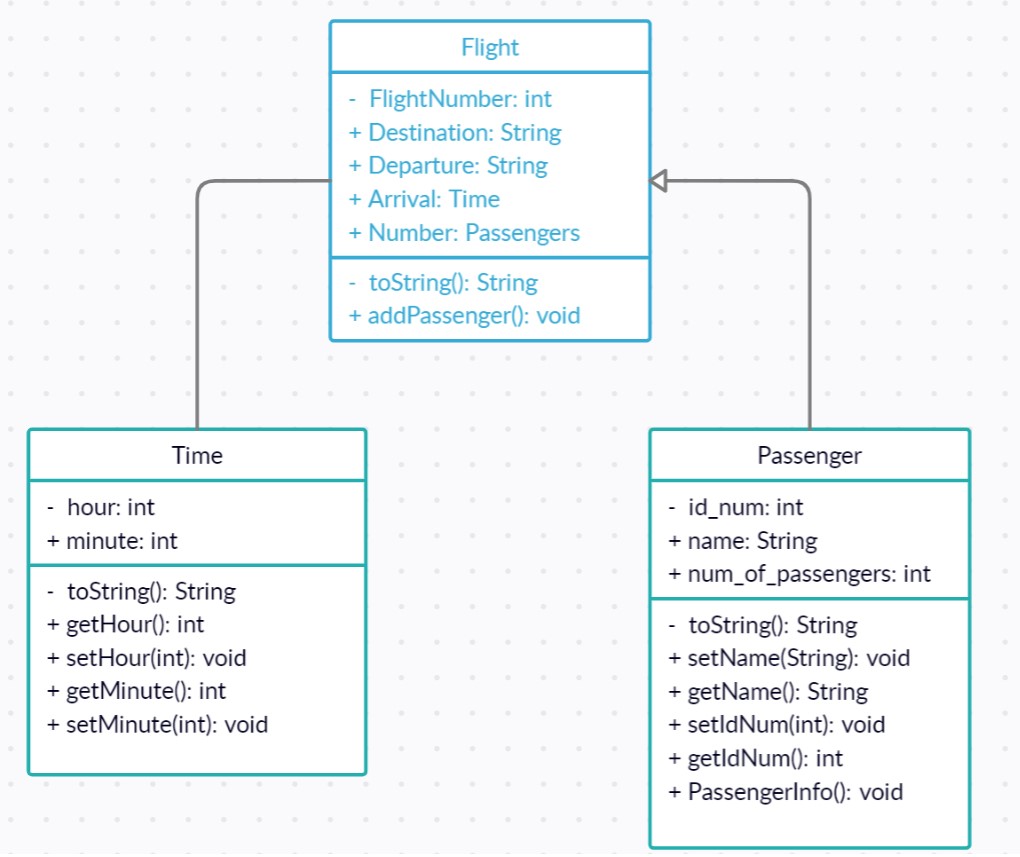
**package** com.company;  
  
**public class** Capacity  
{  
 **public** String **Car\_Capacity**;  
 **public** Capacity()  
 {  
 **this**.**Car\_Capacity** = **"No Capacity"**;  
 }  
 **public void** Capaciti(String Car\_Capacity)  
 {  
 **this**.**Car\_Capacity** = Car\_Capacity;  
 }  
 **public** String toString()  
 {  
 **return Car\_Capacity**;  
 }  
}

**Wheels**

**package** com.company;  
  
**public class** Wheel  
{  
 **public** String **Wheel\_type**;  
 **public** Wheel()  
 {  
 **this**.**Wheel\_type** = **"No Wheel"**;  
 }  
 **public void** Wheels(String Wheel\_type)  
 {  
 **this**.**Wheel\_type** = Wheel\_type;  
 }  
 **public** String toString()  
 {  
 **return Wheel\_type**;  
 }  
}

**Output** **Task 2:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Write complete program for Flight's class, Time's class and Passenger's class with the concept of association and aggregation and design UML class diagram. Functions information also been given in the table below:   |  |  | | --- | --- | | **Method** | **Description** | | addPassenger(Passenger) | This method will add Passenger's object to vector passengerList. | | printInfo() | This method will display all flight information namely ID (Flight number), destination, departure time, arrival time and number of passengers.  For Example:  Flight no : PK-303  Destination : Lahore  Departure : 8:10  Arrival : 9:00  Number of passenger :2 | | getHour() | This method will return the value of attribute **hour** | | getMinute() | This method will return the value of attribute **minute** | |

**Solution** **Main  
package** com.company;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**public class** Main {  
  
 **public static void** main(String[] args)  
 {  
 Passenger\_Class P1 = **new** Passenger\_Class(**"PIA"**,**"PK-314"**,**"021"**,**"lahore"**,12);  
 Passenger\_Class P2 = **new** Passenger\_Class(**"PIA"**,**"PK-314"**,**"022"**,**"lahore"**,12);  
 Passenger\_Class P3 = **new** Passenger\_Class(**"PIA"**,**"PK-314"**,**"023"**,**"lahore"**,12);  
 Passenger\_Class P4 = **new** Passenger\_Class(**"PIA"**,**"PK-314"**,**"025"**,**"lahore"**,12);  
 List<Passenger\_Class> Passengers = **new** ArrayList<Passenger\_Class>();  
 Passengers.add(P1);  
 Passengers.add(P2);  
 Passengers.add(P3);  
 Passengers.add(P4);  
 Flight\_Class obj = **new** Flight\_Class(Passengers);  
 Time\_Class obj1 = **new** Time\_Class(12,45);  
 obj1.getHours();  
 obj1.getMinutes();  
 System.***out***.println(**"\n####Passenger 1#####"**);  
 System.***out***.println(**"Flight No : "** + P1.getFlight\_No());  
 System.***out***.println(**"ID : "** + P1.getId());  
 System.***out***.println(**"Destination : "** + P1.getDestination());  
 System.***out***.println(**"Departure : "** + P1.getDeparture\_Time());  
 System.***out***.println(**"Arrival : "** + obj1.getHours()+**":"**+obj1.getMinutes());  
  
 System.***out***.println(**"\n####Passenger 2#####"**);  
 System.***out***.println(**"Flight No : "** + P2.getFlight\_No());  
 System.***out***.println(**"ID : "** + P2.getId());  
 System.***out***.println(**"Destination : "** + P2.getDestination());  
 System.***out***.println(**"Departure : "** + P2.getDeparture\_Time());  
 System.***out***.println(**"Arrival : "** + obj1.getHours()+**":"**+obj1.getMinutes());  
  
 System.***out***.println(**"\n####Passenger 3#####"**);  
 System.***out***.println(**"Flight No : "** + P3.getFlight\_No());  
 System.***out***.println(**"ID : "** + P3.getId());  
 System.***out***.println(**"Destination : "** + P3.getDestination());  
 System.***out***.println(**"Departure : "** + P3.getDeparture\_Time());  
 System.***out***.println(**"Arrival : "** + obj1.getHours()+**":"**+obj1.getMinutes());  
  
 System.***out***.println(**"\n####Passenger 4#####"**);  
 System.***out***.println(**"Flight No : "** + P4.getFlight\_No());  
 System.***out***.println(**"ID : "** + P4.getId());  
 System.***out***.println(**"Destination : "** + P4.getDestination());  
 System.***out***.println(**"Departure : "** + P4.getDeparture\_Time());  
 System.***out***.println(**"Arrival : "** + obj1.getHours()+**":"**+obj1.getMinutes());  
  
  
 System.***out***.println(**"\nNo Of Passengers In Flight : "** + 4);  
  
 }  
}

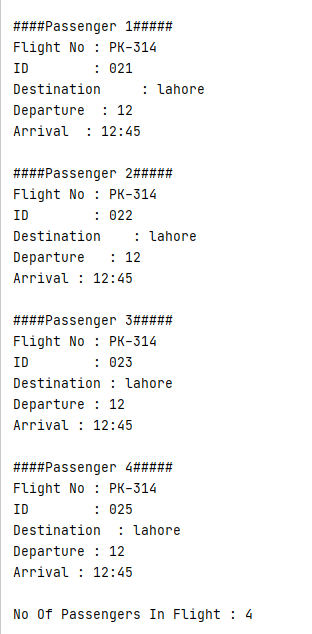
**Flight Class**

**package** com.company;  
**import** java.util.\*;  
**public class** Flight\_Class  
{  
  
 **public int No\_Of\_Passengers**;  
 **private** List<Passenger\_Class> **Passenger**;  
 **public** Flight\_Class()  
 {  
 **this**.**No\_Of\_Passengers** = 0;  
 }  
 **public** Flight\_Class(List<Passenger\_Class> Passenger)  
 {  
 **this**.**Passenger** = Passenger;  
 }  
 **public int** getNo\_Of\_Passengers()  
 {  
 **No\_Of\_Passengers** = 0;  
 List<Passenger\_Class> Passengers = **null**;  
 **for** (Passenger\_Class s : Passengers)  
 {  
 **No\_Of\_Passengers**++;  
 }  
 **return No\_Of\_Passengers**;  
 }  
}

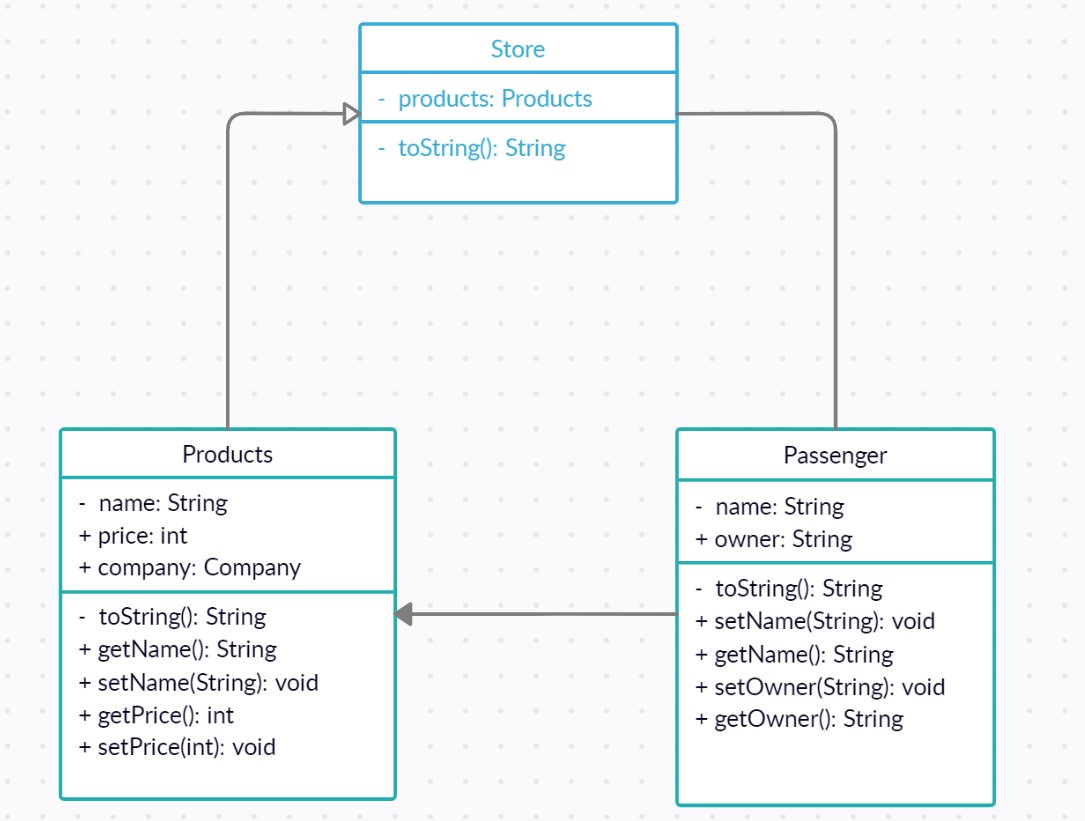
**Passenger Class**

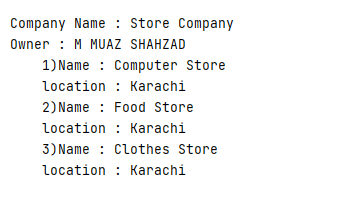
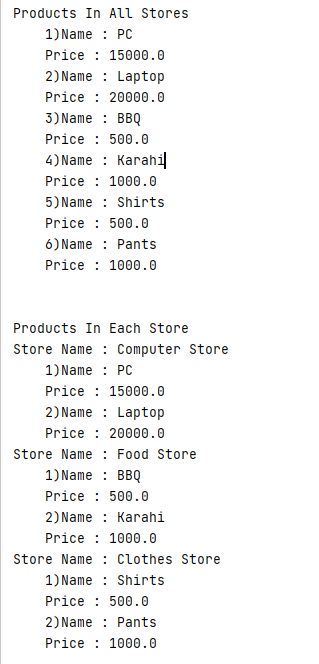
**package** com.company;  
  
  
**public class** Passenger\_Class  
{  
 **int Departure\_Time**;  
 **public** String **Flight\_Name**;  
 **public** String **Flight\_No**;  
 **public** String **Id**;  
 **public** String **Destination**;  
 **public** Passenger\_Class()  
 {  
 **this**.**Flight\_Name** = **"No Flight"**;  
 **this**.**Destination** = **"No Destination"**;  
 **this**.**Id** = **"NO Id"**;  
 **this**.**Flight\_No** = **"No Flight"**;  
 **this**.**Departure\_Time** = 0;  
 }  
 **public** Passenger\_Class(String Flight\_Name,String Fligh\_No,String Id, String Destination,**int** Departure\_Time)  
 {  
 **this**.**Flight\_Name** = Flight\_Name;  
 **this**.**Flight\_No** = Fligh\_No;  
 **this**.**Id** = Id;  
 **this**.**Destination** = Destination;  
 **this**.**Departure\_Time** = Departure\_Time;  
  
 }  
 **public int** getDeparture\_Time() {  
 **return Departure\_Time**;  
 }  
 **public** String getFlight\_Name() {  
 **return Flight\_Name**;  
 }  
  
 **public** String getFlight\_No() {  
 **return Flight\_No**;  
 }  
 **public** String getId()  
 {  
 **return Id**;  
 }  
 **public** String getDestination()  
 {  
 **return Destination**;  
 }  
  
  
}

**Time Class  
package** com.company;  
  
**public class** Time\_Class  
{  
  
 **int Hours**;  
 **int Minutes**;  
 **public** Time\_Class()  
 {  
  
 **this**.**Hours** = 0;  
 **this**.**Minutes** = 0;  
 }  
 **public** Time\_Class(**int** Hours,**int** Minutes)  
 {  
 **this**.**Hours** = Hours;  
 **this**.**Minutes** = Minutes;  
 }  
 **public int** getHours()  
 {  
 **return Hours**;  
 }  
 **public int** getMinutes() {  
 **return Minutes**;  
 }  
}

**Output**

**Task 3: A company manages many stores. Each Store contains many Products. Implement Product, Store and Company classes using association and aggrigation concepts and design UML class diagram.**

**Solution** **Main  
package** com.company;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**public class** Main {  
  
 **public static void** main(String[] args)  
 {  
 Products p1 = **new** Products(**"PC"**,15000);  
 Products p2 = **new** Products(**"Laptop"**,20000);  
 Products p3 = **new** Products(**"BBQ"**,500);  
 Products p4 = **new** Products(**"Karahi"**,1000);  
 Products p5 = **new** Products(**"Shirts"**,500);  
 Products p6 = **new** Products(**"Pants"**,1000);  
 p1.Addproductinlist(p1);  
 p2.Addproductinlist(p2);  
 p3.Addproductinlist(p3);  
 p4.Addproductinlist(p4);  
 p5.Addproductinlist(p5);  
 p5.Addproductinlist(p6);  
  
 List<Products> producttype1 = **new** ArrayList<Products>();  
 producttype1.add(p1);  
 producttype1.add(p2);  
 List<Products> producttype2 = **new** ArrayList<Products>();  
 producttype2.add(p3);  
 producttype2.add(p4);  
 List<Products> producttype3 = **new** ArrayList<Products>();  
 producttype3.add(p5);  
 producttype3.add(p6);  
 System.***out***.println();  
 Products.*PrintList*();  
 Store s1 = **new** Store(**"Computer Store"**,**"Karachi"**,producttype1);  
 s1.Addstoreinlist(s1);  
 Store s2 = **new** Store(**"Food Store"**,**"Karachi"**,producttype2);  
 s2.Addstoreinlist(s2);  
 Store s3 = **new** Store(**"Clothes Store"**,**"Karachi"**,producttype3);  
 s3.Addstoreinlist(s3);  
 System.***out***.println();  
 System.***out***.println(**"\nProducts In Each Store"**);  
 s1.Printproductlist();  
 s2.Printproductlist();  
 s3.Printproductlist();  
 Company c = **new** Company(**"Store Company"**,**"M MUAZ SHAHZAD"**,Store.*storelist*);  
 System.***out***.println();  
 c.printstorelist();  
 System.***out***.println();  
 }  
  
 **Company Class  
package** com.company;  
**import** java.util.List;  
**public class** Company  
{  
 String **Company\_Name**;  
 String **Owner**;  
 List<Store> **allstores**;  
 Company(String Company\_Name,String Owner,List<Store> stores){  
 **this**.**Company\_Name**=Company\_Name;  
 **this**.**Owner**=Owner;  
 **this**.**allstores** = stores;  
 }  
 **public void** printstorelist()  
 {  
 **int** i=1;  
 System.***out***.println(**"Company Name : "** +**Company\_Name**+ **"\nOwner : "** +**Owner**);  
 **for**(Store d: **allstores**){  
 System.***out***.println(**" "**+i+**")Name : "**+d.**Store\_Name** + **"\n\tlocation : "** +d.**location**);  
 i++;  
 }  
 }  
  
 }

**Store Class  
package** com.company;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**public class** Store  
{  
 String **Store\_Name**;  
 String **location**;  
 List<Products> **productinstore**;  
 **static** List<Store> *storelist*=**new** ArrayList<Store>();  
 Store(String Store\_Name,String location,List<Products> productinstore){  
 **this**.**Store\_Name**=Store\_Name;  
 **this**.**location**=location;  
 **this**.**productinstore** = productinstore;  
 }  
 **public void** Addstoreinlist(Store s){  
 *storelist*.add(s);  
 }  
 **public void** Printproductlist() {  
 **int** i = 1;  
 System.***out***.println(**"Store Name : "** + **Store\_Name**);  
 **for** (Products d : **productinstore**) {  
 System.***out***.println(**" "** + i + **")Name : "** + d.**Product\_Name** + **"\n\tPrice : "** + d.**price**);  
 i++;  
 }  
 }  
} **Products Class  
package** com.company;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**public class** Products  
{  
 String **Product\_Name**;  
 **double price**;  
 **static** List<Products> *products*=**new** ArrayList<Products>();  
 Products(String Product\_Name,**double** price){  
 **this**.**Product\_Name**=Product\_Name;  
 **this**.**price**=price;  
 }  
 **public void** Addproductinlist(Products p){  
 *products*.add(p);  
 }  
 **public static void** PrintList() {  
 **int** i = 1;  
 System.***out***.println(**"Products In All Stores"**);  
 **for** (Products d : *products*) {  
 System.***out***.println(**" "** + i + **")Name : "** + d.**Product\_Name** + **"\n\tPrice : "** + d.**price**);  
 i++;  
 }  
 }  
  
}  
  
 **Output**

Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**10**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Create a Java GUI application that provides the user usage of Ohm’s Law formula. In electrical circuits, Ohm's law states that the current through a conductor between two points is directly proportional to the potential difference or voltage across the two points, and inversely proportional to the resistance between them, provided that the temperature remains constant.  The mathematical equation that describes this relationship is:    where V is the potential difference measured across the resistance in units of volts; I is the current through the resistance in units of amperes and R is the resistance of the conductor in units of ohms. More specifically, Ohm's law states that the R in this relation is constant, independent of the current.  The law was named after the German physicist Georg Ohm, who, in a treatise published in 1827, described measurements of applied voltage and current through simple electrical circuits containing various lengths of wire. He presented a slightly more complex equation than the one above to explain his experimental results. The above equation is the modern  form of Ohm's law. |
| 02 | Create a GUI application that prints the multiplication table of any given number. |

Submitted On:

18/06/2021

(Date: DD/MM/YY)  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 **Task 1:** Create a Java GUI application that provides the user usage of Ohm’s Law formula. In electrical circuits, Ohm's law states that the current through a conductor between two points is directly proportional to the potential difference or voltage across the two points, and inversely proportional to the resistance between them, provided that the temperature remains constant.

The mathematical equation that describes this relationship is:



where V is the potential difference measured across the resistance in units of volts; I is the current through the resistance in units of amperes and R is the resistance of the conductor in units of ohms. More specifically, Ohm's law states that the R in this relation is constant, independent of the current.

The law was named after the German physicist Georg Ohm, who, in a treatise published in 1827, described measurements of applied voltage and current through simple electrical circuits containing various lengths of wire. He presented a slightly more complex equation than the one above to explain his experimental results. The above equation is the modern

form of Ohm's law.

**Solution:**import javax.swing.JTextArea;

public class NewJFrame extends javax.swing.JFrame {

public NewJFrame() {

initComponents();

}

@SuppressWarnings("unchecked")

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jPanel1 = new javax.swing.JPanel();

jTextField1 = new javax.swing.JTextField();

jTextField2 = new javax.swing.JTextField();

jTextField3 = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jLabel7 = new javax.swing.JLabel();

jLabel8 = new javax.swing.JLabel();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jButton4 = new javax.swing.JButton();

jLabel1.setText("jLabel1");

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jTextField1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField1ActionPerformed(evt);

}

});

jTextField2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jTextField2ActionPerformed(evt);

}

});

jLabel2.setFont(new java.awt.Font("Ebrima", 1, 18)); // NOI18N

jLabel2.setText("Ohms Law : I = V/R");

jLabel3.setText("OHMS");

jLabel4.setText("VOLTS");

jLabel5.setText("Voltage");

jLabel6.setText("AMPERES");

jLabel7.setText("Resistance");

jLabel8.setText("Current");

jButton1.setText("Calculate");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Print");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Clear");

jButton3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

jButton4.setText("Quit");

jButton4.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton4ActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jLabel7, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jLabel8, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(0, 25, Short.MAX\_VALUE))

.addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jTextField1)

.addComponent(jTextField2)

.addComponent(jTextField3)

.addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(jButton2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButton4, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addGap(36, 36, 36)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 43, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 48, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jLabel6)

.addGap(5, 5, 5)))

.addGap(119, 119, 119))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(7, 7, 7)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 53, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(1, 1, 1)

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 27, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED\_SIZE, 32, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1)

.addComponent(jButton3))

.addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton2)

.addComponent(jButton4))

.addContainerGap(31, Short.MAX\_VALUE))

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap(41, Short.MAX\_VALUE)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

);

pack();

}// </editor-fold>

private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

float current,voltage,resistance;

voltage = Float.parseFloat(jTextField1.getText());

resistance = Float.parseFloat(jTextField2.getText());

current = voltage/resistance;

jTextField3.setText(Float.toString(current));

}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

System.exit(0);

}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

jTextField1.requestFocus();

}

public static void main(String args[])

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new NewJFrame().setVisible(true);

}

});

}

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JLabel jLabel7;

private javax.swing.JLabel jLabel8;

private javax.swing.JPanel jPanel1;

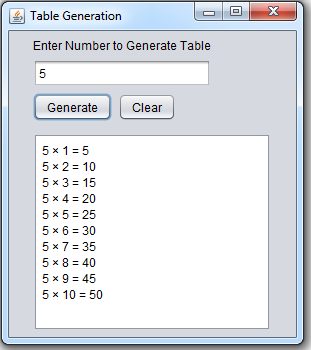
private javax.swing.JTextField jTextField1;

private javax.swing.JTextField jTextField2;

private javax.swing.JTextField jTextField3;

}

**Output:**  
  
 **Task 2:** **Create a GUI application that prints the multiplication table of any given number.**



**Solution:**import static java.lang.Integer.parseInt;

public class NewJFrame extends javax.swing.JFrame {

/\*\*

\* Creates new form NewJFrame

\*/

public NewJFrame() {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jTextField1 = new javax.swing.JTextField();

jButton1 = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jButton1.setText("Calculate");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jLabel1.setText("NUMBER : ");

jLabel2.setFont(new java.awt.Font("Ebrima", 0, 24)); // NOI18N

jLabel2.setText(" TABLE :");

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jScrollPane1.setViewportView(jTextArea1);

jButton2.setText("Clear");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(32, 32, 32)

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 68, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 92, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 131, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jButton1)

.addGap(18, 18, 18)

.addComponent(jButton2, javax.swing.GroupLayout.PREFERRED\_SIZE, 67, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(layout.createSequentialGroup()

.addGap(89, 89, 89)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 325, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(141, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(55, 55, 55)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, 25, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButton1)

.addComponent(jLabel1)

.addComponent(jButton2))

.addGap(27, 27, 27)

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 260, Short.MAX\_VALUE)

.addContainerGap())

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

int index ;

index = parseInt(jTextField1.getText());

for (int i =1 ;i<=10;i++)

{

int cal = index \* i;

jTextArea1.append(index+" \* "+i+" = "+cal+"\n");

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

jTextArea1.setText("");

jTextField1.setText("");

jTextField1.requestFocus();

}   
  
**MAIN:**

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(NewJFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new NewJFrame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JScrollPane jScrollPane1;

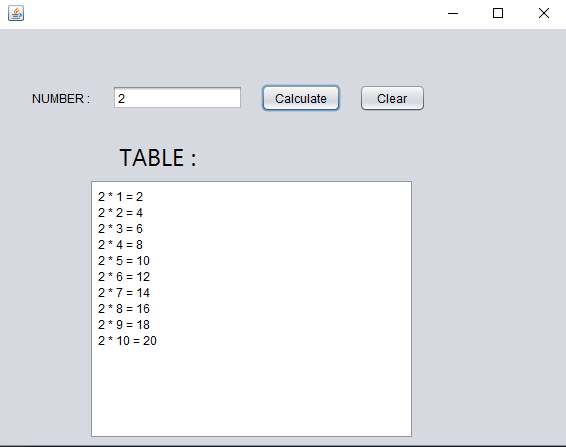
private javax.swing.JTextArea jTextArea1;

private javax.swing.JTextField jTextField1;

// End of variables declaration

}

**Output:**



Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**11**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | Write a program which implements a interface of Banking System by having all standard functionalities and will be implemented by branches.  **Hint:(Interface Methods)**  CreateAccount()  Search Account details()  Update CustInfo()  Cash Withdraw()  Cash Deosit() |
| 02 | By looking at the formulae for an ellipse, provide the missing code for all of the methods in the class Ellipse including the toString() method. Test your program using the TestShapes.java class. Your output should look as follows (for an ellipse with *a* = 10 and *b* = 7) (values are randomly generated).   1. Square 2. Area=100.0 3. Perimeter=40.0 4. Ellipse 5. Area=219.9114857512855 6. Perimeter=53.8212680240788 7. Eccentricity=0.714142842854285 8. Press any key to continue...   How about the following class Circle. Since a Circle is a special case of an Ellipse, will the output of TestShapes.java be affected if the following class is used instead of the class Circle used previously: *Circle.java* public class Circle extends Ellipse {  public Circle(double radius){  super(radius, radius);  }  }  With this modification, the class diagram would look as follows: |

Submitted On:

**19-06-2021**

(Date: DD/MM/YY)

Task 1: **Write a program which implements a interface of Banking System by having all standard functionalities and will be implemented by branches.**

**Hint:(Interface Methods)**

CreateAccount()

Search Account details()

Update CustInfo()

Cash Withdraw()

Cash Deosit()

Solution:  
Interface Banking:  
package javaapplication22;

import java.util.Scanner;

interface BankingSystem

{

public void CreateAccount();

public void SearchAccountdetails();

public void UpdateCustInfo();

public void CashWithdraw();

public void CashDeosit();

}

class branch1 implements BankingSystem

{

Scanner input = new Scanner(System.in);

float balance,accnum,index=1;

String name;

public void CreateAccount()

{

System.out.println("Enter Your Name: ");

name=input.next();

System.out.println("An bank account has been opened");

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

accnum= 4344343+index;

System.out.println("accnum: "+accnum);

index += 1;

}

public void SearchAccountdetails()

{

System.out.println("Enter account number: ");

float account = input.nextFloat();

if(account == accnum)

{

System.out.println("Account Found.");

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

System.out.println("Account Number: "+accnum);

System.out.println("Balance: "+balance);

}

}

public void UpdateCustInfo()

{

System.out.println("Enter Your Name: ");

name=input.next();

System.out.println("Your name linked with your bank account is updated");

}

public void CashWithdraw()

{

System.out.println("Enter amount: ");

float amount = input.nextFloat();

if(balance>=amount)

{

System.out.println(amount+" Rupees has been withdrawned from your account");

balance -= amount;

}

}

public void CashDeosit()

{

System.out.println("Enter amount: ");

float amount = input.nextFloat();

System.out.println(amount+" Rupees has been deposited to your account");

balance += amount;

}

}

**Main:**public class JavaApplication22 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

branch1 c1 = new branch1();

String choice;

String ans="yes";

while (ans.equals("yes") || ans.equals("Yes"))

{

System.out.println("Enter 'A' to create account\nEnter 'B' to search account\nEnter 'C' to deposit money\nEnter 'D' to withdraw money\nEnter 'E' to update account info\n");

choice = input.next();

switch(choice)

{

case "A":

case "a":

c1.CreateAccount();

break;

case "B":

case "b":

c1.SearchAccountdetails();

break;

case "C":

case "c":

c1.CashDeosit();

break;

case "D":

case "d":

c1.CashWithdraw();

break;

case "E":

case "e":

c1.UpdateCustInfo();

break;

default:

System.out.println("Sorry invalid option selected");

}

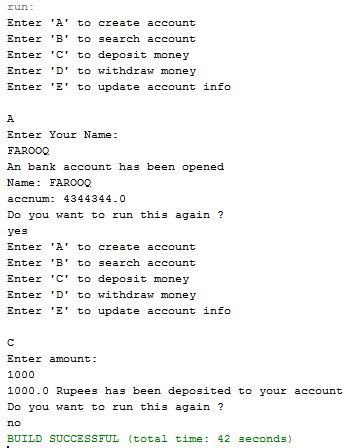
System.out.println("Do you want to run this again ?");

ans=input.next();

}

}

}

Output:  
******  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
Task 2: **By looking at the formulae for an ellipse, provide the missing code for all of the methods in the class Ellipse including the toString() method. Test your program using the TestShapes.java class. Your output should look as follows (for an ellipse with *a* = 10 and *b* = 7) (values are randomly generated).**

1. Square
2. Area=100.0
3. Perimeter=40.0
4. Ellipse
5. Area=219.9114857512855
6. Perimeter=53.8212680240788
7. Eccentricity=0.714142842854285
8. Press any key to continue...

How about the following class Circle. Since a Circle is a special case of an Ellipse, will the output of TestShapes.java be affected if the following class is used instead of the class Circle used previously:

##### *Circle.java*

public class Circle extends Ellipse {

public Circle(double radius){

super(radius, radius);

}

}

With this modification, the class diagram would look as follows:

Solution  
  
Interface Eccentric  
/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication20;

import java.util.Random;

interface Eccentric {

double eccentricity();

}

abstract class Shape{

public String name(){

return getClass().getName();

}

public abstract double area();

public abstract double perimeter();

public String toString() {

return "\n" +name() +"\n Area=" +area() +"\nPerimeter=" +perimeter();

}

}  
  
Circle Class:  
class Circle extends Shape {

private double radius;

public Circle(double r){

radius = r;

}

public double area(){

return Math.PI \* (radius \* radius);

}

public double perimeter(){

return 2.0 \* Math.PI \* radius;

}

public double getRadius(){

return radius;

}

}

Class Rectangle:  
class Rectangle extends Shape {

private double length;

private double width;

public Rectangle(double length, double width){

this.length = length;

this.width = width;

}

public double area(){

return length \* width;

}

public double perimeter(){

return 2\*(length+width);

}

public double getLength(){

return length;

}

public double getWidth(){

return width;

}

}

Class Square:  
class Square extends Rectangle{

public Square(double length){

super(length, length);

}

}

Class Equilateral Triangle:  
class EquilateralTriangle extends Shape

{

private double side;

public EquilateralTriangle(double side)

{

this.side=side;

}

public double getSide()

{

return side;

}

public double area(){

return ((0.25)\*(side\*side));

}

public double perimeter(){

return 3\*side;

}

}

Class Ellipse:  
class Ellipse extends Shape implements Eccentric

{

double a, b;

public Ellipse(double s1, double s2){

if(s1 < s2) {

a = s2;

b = s1;

}

else {

a = s1;

b = s2;

}

}

public double perimeter(){

if (a == b)

{

double Perimeter = 2\*3.14\*a;

return Perimeter;

}

else

{

double Perimeter = 3.14\*(Math.sqrt(2\*(a\*a + b\*b) - (a\*a + b\*b-2\*a\*b)/2));

return Perimeter;

}

}

public double area(){

double Area = 3.14\*a\*b;

return Area;

}

public double eccentricity(){

double Eccentricity=0;

if(a>b)

{

Eccentricity = (Math.sqrt(a\*a-b\*b))/a;

}

else if(b>a)

{

Eccentricity = (Math.sqrt(b\*b-a\*a))/b;

}

return Eccentricity;

}

}

Class Test Shapes:  
public class TestShapes {

public static Shape[] createShape() {

final int SIZE = 5;

final double DIMENSION = 100;

final int NUMBEROFSHAPES = 5;

Random generator = new Random();

//create an array having b/w 1 and SIZE entries

Shape[] randomShapes = new Shape[generator.nextInt(SIZE) + 1];

for(int i = 0; i < randomShapes.length; i++)

{

//randomly generate values b/w 0 and NUMBEROFSHAPES - 1

int assigner = generator.nextInt(NUMBEROFSHAPES);

switch(assigner) {

case 0: randomShapes[i] =

new Rectangle(generator.nextDouble()\*DIMENSION,generator.nextDouble()\*DIMENSION);

break;

case 1: randomShapes[i] = new Circle(generator.nextDouble()\*DIMENSION);

break;

case 2: randomShapes[i] = new Square(generator.nextDouble()\*DIMENSION);

break;

case 3: randomShapes[i] = new EquilateralTriangle(generator.nextDouble()\*DIMENSION);

break;

case 4: randomShapes[i] = new Ellipse (generator.nextDouble()\*DIMENSION,generator.nextDouble()\*DIMENSION);

break;

}

}

return randomShapes;

}

public static void main(String[] args){

Shape[] randomShapes = TestShapes.createShape();

for(int i = 0; i < randomShapes.length; i++){

System.out.println(randomShapes[i]);

if(randomShapes[i] instanceof Circle)

{

System.out.println("Radius= " + ((Circle) randomShapes[i]).getRadius());

}

else if(randomShapes[i] instanceof Square)

{

System.out.println("Length= " +

((Square) randomShapes[i]).getLength());

}

else if(randomShapes[i] instanceof Rectangle)

{

System.out.println("Length= " +

((Rectangle) randomShapes[i]).getLength()

+ "\nWidth= " +

((Rectangle) randomShapes[i]).getWidth());

}

else if(randomShapes[i] instanceof Ellipse)

{

System.out.println("Eccentricity = "+

((Ellipse) randomShapes[i]).eccentricity());

}

else if(randomShapes[i] instanceof EquilateralTriangle)

{

System.out.println("Each Side= " +

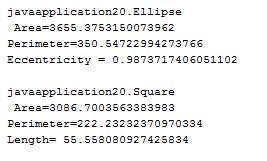
((EquilateralTriangle) randomShapes[i]).getSide());

}

}

}

}

Output:  
******

Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**12**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | **Write a program for exam department which provide abstract class and method of Exam type which contains general methods related to exams and can be used by different department for conducting exams.** |
| 02 | **You have to implement the UML diagram given below.Also Design and implement a subclass “EquilateralTriangle” having a double variable side denoting the three sides of the equilateral triangle [Note that since all the 3 sides are equal, the constructor will have only one parameter]. The area and perimeter of the equilateral triangle are given as follows:**  **Area = ¼\*\*(*side*)2**  **Perimeter = 3\**side***  **Provide accessor methods for the sides. Test your class using the TestShapes and DownCastingShapes classes.** |

Submitted On:

**19-06-2021**

**Task 1:** **Write a program for exam department which provide abstract class and method of Exam type which contains general methods related to exams and can be used by different department for conducting exams.**

**Solution  
Class Exam Department:**package javaapplication21;

abstract class ExamsDepartment {

protected int std\_present;

protected int std\_absent;

ExamsDepartment(int std\_present,int std\_absent)

{

this.std\_present=std\_present;

this.std\_absent=std\_present;

}

public void Start ()

{

System.out.println("Exam has been started");

}

public void Attendence ()

{

System.out.println("Present Student: "+std\_present);

System.out.println("Present Student: "+std\_absent);

}

public void End ()

{

System.out.println("Exam is ended");

}

public void check\_copies()

{

System.out.println("Copies has been checked");

}

public abstract void Calculate\_Gpa();

public abstract void Collect\_Copies ();

}

**Class Department 1:**class Department1 extends ExamsDepartment{

public Department1(int std\_present, int std\_absent) {

super(std\_present, std\_absent);

}

public void Calculate\_Gpa()

{

System.out.println("Gpa Calculated");

}

public void Collect\_Copies ()

{

System.out.println("Copies Collected");

}

}

**Main:**public class JavaApplication21 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

ExamsDepartment e1 = new Department1(25,5);

e1.Attendence();

e1.Start();

e1.End();

if (e1 instanceof Department1)

{

Department1 e2 = (Department1)e1;

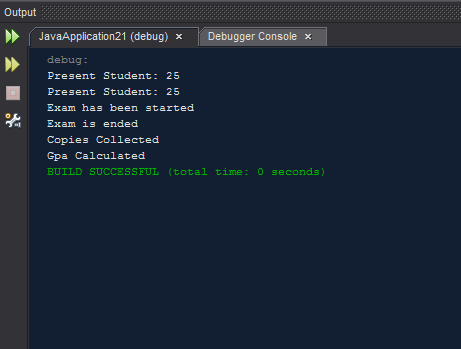
e2.Collect\_Copies();

e2.Calculate\_Gpa();

}

}

}

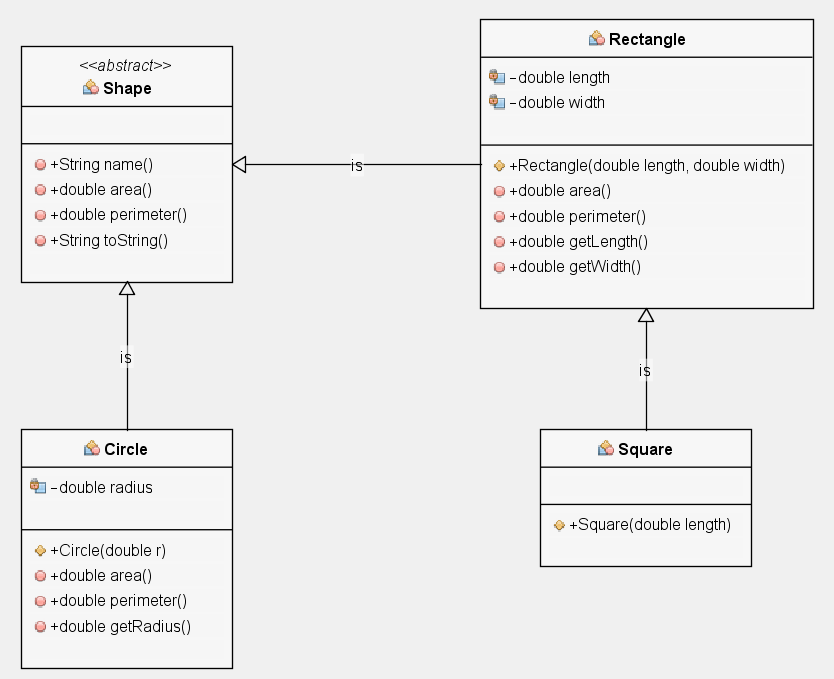
**Output:**

**Task 2: You have to implement the UML diagram given below. Also Design and implement a subclass “Equilateral Triangle” having a double variable side denoting the three sides of the equilateral triangle [Note that since all the 3 sides are equal, the constructor will have only one parameter]. The area and perimeter of the equilateral triangle are given as follows:**

**Area = ¼\*\*(*side*)2**

**Perimeter = 3\**side***

**Provide accessor methods for the sides. Test your class using the TestShapes and DownCastingShapes classes.**

  
  
  
Solution  
**Class Shapes:**  
/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication20;

import java.util.Random;

abstract class Shape{

public String name(){

return getClass().getName();

}

public abstract double area();

public abstract double perimeter();

public String toString() {

return "\n" +name() +"\n Area=" +area() +"\nPerimeter=" +perimeter();

}

}

Class Circle:  
class Circle extends Shape {

private double radius;

public Circle(double r){

radius = r;

}

public double area(){

return Math.PI \* (radius \* radius);

}

public double perimeter(){

return 2.0 \* Math.PI \* radius;

}

public double getRadius(){

return radius;

}

}

Class Rectangle:  
class Rectangle extends Shape {

private double length;

private double width;

public Rectangle(double length, double width){

this.length = length;

this.width = width;

}

public double area(){

return length \* width;

}

public double perimeter(){

return 2\*(length+width);

}

public double getLength(){

return length;

}

public double getWidth(){

return width;

}

}

Class Square:  
class Square extends Rectangle{

public Square(double length){

super(length, length);

}

}

Class EquilateralTrinagle:  
class EquilateralTriangle extends Shape

{

private double side;

public EquilateralTriangle(double side)

{

this.side=side;

}

public double getSide()

{

return side;

}

public double area(){

return ((0.25)\*(side\*side));

}

public double perimeter(){

return 3\*side;

}

}

Class Test shapes:  
public class TestShapes {

public static Shape[] createShape() {

final int SIZE = 5;

final double DIMENSION = 100;

final int NUMBEROFSHAPES = 4;

Random generator = new Random();

//create an array having b/w 1 and SIZE entries

Shape[] randomShapes = new Shape[generator.nextInt(SIZE) + 1];

for(int i = 0; i < randomShapes.length; i++)

{

//randomly generate values b/w 0 and NUMBEROFSHAPES - 1

int assigner = generator.nextInt(NUMBEROFSHAPES);

switch(assigner) {

case 0: randomShapes[i] =

new Rectangle(generator.nextDouble()\*DIMENSION,generator.nextDouble()\*DIMENSION);

break;

case 1: randomShapes[i] = new Circle(generator.nextDouble()\*DIMENSION);

break;

case 2: randomShapes[i] = new Square(generator.nextDouble()\*DIMENSION);

break;

case 3: randomShapes[i] = new EquilateralTriangle(generator.nextDouble()\*DIMENSION);

break;

}

}

return randomShapes;

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

Shape[] randomShapes = TestShapes.createShape();

for(int i = 0; i < randomShapes.length; i++)

System.out.println(randomShapes[i].toString());

}

}

Output:  
  
