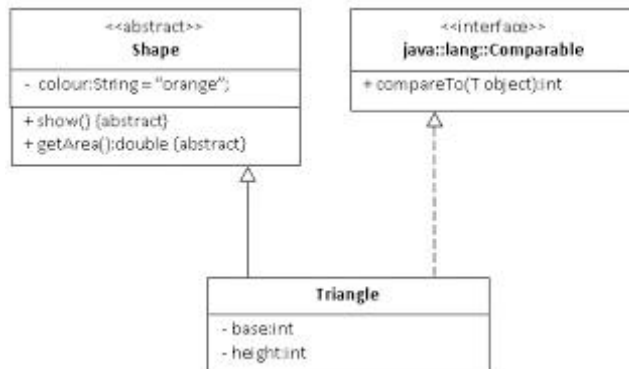


# ASSIGNMENT 6 : LAB 5

## Question 1

Provide an alternative implementation of the Triangle class based on the following specification:



Have a ShapeManager class that holds the main() loop.

### Code:

ShapeManager

```
import java.util.Scanner;
```

```
/**
```

```
 * @author Harsh
```

```
 */
```

```
public class ShapeManager {
```

```
    public static void main(String args[])
```

```
    {
```

```
        int base,base1,height,height1,ch,p; double area;
```

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

```
        do{
```

```
            System.out.println("Enter 1 for showing dimensions \nEnter 2 for calculating area\nEnter 3 for comparing areas\nEnter 4 to exit"); // accepting choice
```

```
            ch=sc.nextInt();
```

```
            switch(ch)
```

```
{
    case 1: System.out.println("Enter base and height");
        base=sc.nextInt();
        height=sc.nextInt();
        Triangle myTri=new Triangle(base,height);
        myTri.show();
        break;
    case 2: System.out.println("Enter base and height");
        base=sc.nextInt();
        height=sc.nextInt();
        Triangle yrTri=new Triangle(base,height);
        area=yrTri.getArea();
        System.out.println("Area is "+area);
        break;
    case 3: System.out.println("Enter base and height for Triangle 1");
        base=sc.nextInt();
        height=sc.nextInt();
        System.out.println("Enter base and height for Triangle 2");
        base1=sc.nextInt();
        height1=sc.nextInt();
        Triangle obj1=new Triangle(base,height);
        Triangle obj2=new Triangle(base1,height1);
        p=obj1.compareTo(obj2); // getting value to compare
        if(p>0)
            System.out.println("Triangle 1 area is more than Triangle 2");
        else if(p==0)
            System.out.println("Triangles have equal area");
        else
            System.out.println("Triangle 2 area is more than Triangle 1");
        break;
    case 4: System.out.println("Thank You!!");
```

## OBJECT ORIENTED PROGRAMMING LAB

```
        break;
        default: System.out.println("Invalid choice");
        break;
    }
} while(ch!=4);
}
}
```

## Triangle

```
import java.lang.Comparable;
/**
 * @author Harsh
 */
public class Triangle extends Shape implements Comparable {

    private int base,height;
    public Triangle(int b, int h){
        this.base=b;
        this.height=h;
    }
    public double getArea(){
        return (.5*base*height); // returning calculated area
    }

    public void show()
    {
        System.out.println("Base is "+base);
        System.out.println("Height is "+height);
    }
    public int compareTo(Object o){
        Triangle obj=(Triangle)o;
```

## OBJECT ORIENTED PROGRAMMING LAB

```

        if(this.getArea()<obj.getArea())
            return -1;
        else if(this.getArea()>obj.getArea())
            return 1;
        else
            return 0;
    }
}

```

## Shape

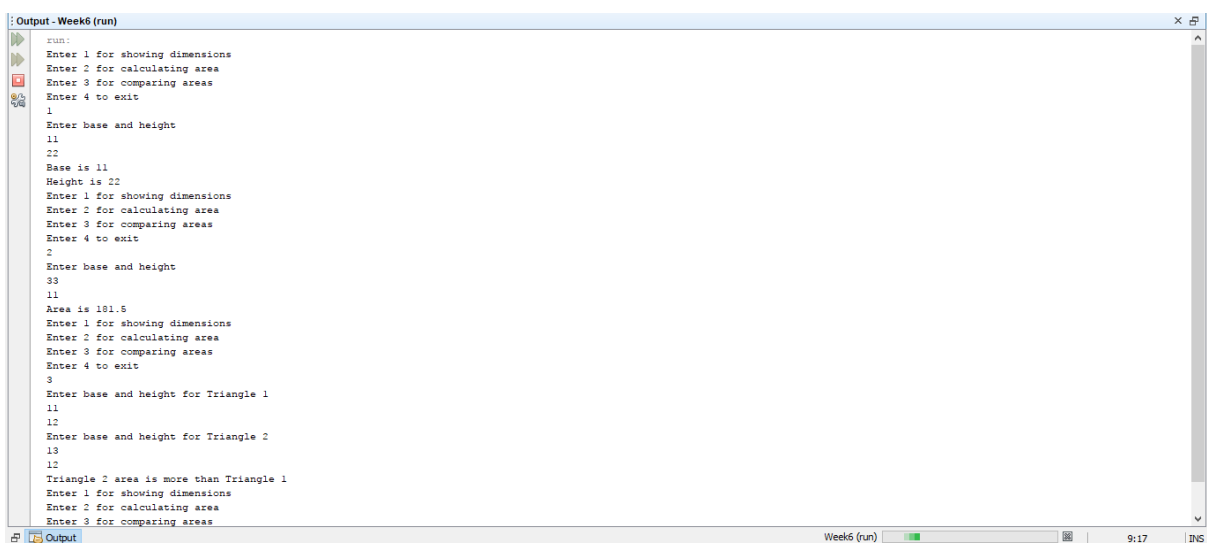
```

/**
 * @author Harsh
 */
public abstract class Shape {

    public abstract double getArea();
    public abstract void show();
}

```

## OUTPUT :



```

run:
Enter 1 for showing dimensions
Enter 2 for calculating area
Enter 3 for comparing areas
Enter 4 to exit
1
Enter base and height
11
22
Base is 11
Height is 22
Enter 1 for showing dimensions
Enter 2 for calculating area
Enter 3 for comparing areas
Enter 4 to exit
2
Enter base and height
33
11
Area is 181.5
Enter 1 for showing dimensions
Enter 2 for calculating area
Enter 3 for comparing areas
Enter 4 to exit
3
Enter base and height for Triangle 1
11
12
Enter base and height for Triangle 2
13
12
Triangle 2 area is more than Triangle 1
Enter 1 for showing dimensions
Enter 2 for calculating area
Enter 3 for comparing areas

```

(a) Which of the methods of the Triangle class did you have to change? Why?

Answer: We had to change the method for comparing. We imported the Comparable interface from java.lang package. Hence we are changing the statements in that method.

(b) Which is a better design - Lab#5 Question 1 or Lab#3 Question 1? Why?

Answer: Lab #5's design is a better one since it has an abstract class which provides the Back bone of the functions to be implemented. It also has an interface which is imported from the lang package, which provides a common implementation to unrelated classes.

## OBJECT ORIENTED PROGRAMMING LAB

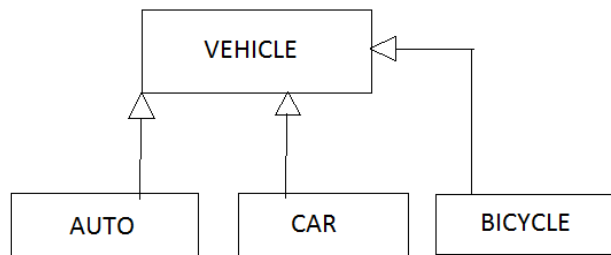
**Question 2:** Create a Virtual World (game!) where the following entities exist:

>Person, Student, CSEStudent

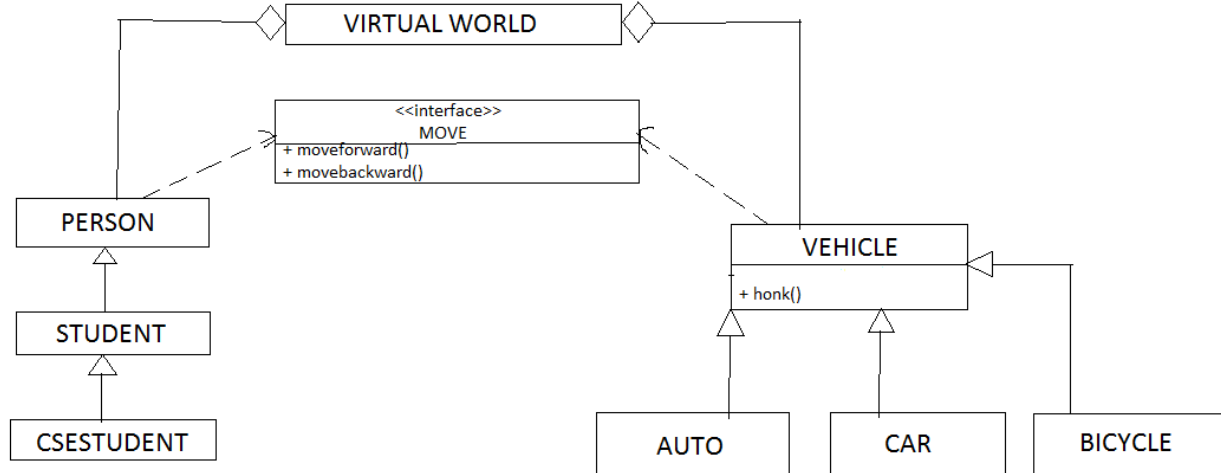
>Auto, Car, Bicycle.

All vehicles have the attribute numWheels and will honk(). All entities in the Virtual World will need to move forward and backward.

- (a) Design the Vehicle hierarchy, explaining the use of abstract class / concrete class / interface.



- b) Draw the class diagram for the Virtual World, consisting of the Person hierarchy and Vehicle hierarchy. Explain how movement is incorporated in your design.



Movement is incorporated using an interface. We are using an interface because we have unrelated classes here, namely Person and Vehicle which have common attributes, i.e. moveforward() and movebackward().

- (c) Implement your design from part (b) with a VirtualWorldManager.

**CODE:**

**Person.java**

## OBJECT ORIENTED PROGRAMMING LAB

```
/**
 *
 * @author Harsh
 */
import java.util.Scanner;
public class Person implements Move//implements interface
{
    public void show()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Press 1 to move Forward, 2 to move Backward");
        int m=sc.nextInt();
        System.out.println("I am a Person and I walk!");
        if(m==1)
            moveforward();
        else
            movebackward();
    }
    public void moveforward()//To move forward
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the distance to travel");
        int d=sc.nextInt();
        System.out.println("I moved forward by "+d+" kms");
    }
    public void movebackward()//To move backward
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the distance to travel");
        int d=sc.nextInt();
        System.out.println("I moved backward by "+d+" kms");
    }
}
```

**Student.java**

```
import java.util.Scanner;
public class Student extends Person
{
    public void show()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Press 1 to move Forward, 2 to move Backward");
        int m=sc.nextInt();
        System.out.println("I am a Student and I walk!");
        if(m==1)
            super.moveforward();
    }
}
```

## OBJECT ORIENTED PROGRAMMING LAB

```

        else
            super.movebackward();
    }
}

```

**CSEStudent.java**

```

import java.util.Scanner;
public class CSEStudent extends Student
{
    public void show()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Press 1 to move Forward, 2 to move Backward");
        int m=sc.nextInt();
        System.out.println("I am a CSE Student and I walk!");
        if(m==1)
            super.moveforward();
        else
            super.movebackward();
    }
}

```

**Vehicle.java**

```

import java.util.Scanner;
public class Vehicle implements Move//implements interface
{
    public void show()
    {
        System.out.println("I am a Vehicle");
    }
    public String honk()
    {
        return "HONK!";
    }
    public void moveforward()//To move forward
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the distance to travel");
        int d=sc.nextInt();
        System.out.println("I moved forward by "+d+" kms");
    }
    public void movebackward()//To move backward
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the distance to travel");
        int d=sc.nextInt();
        System.out.println("I moved backward by "+d+" kms");
    }
}

```



## OBJECT ORIENTED PROGRAMMING LAB

```
}  
}
```

**Auto.java**

```
import java.util.Scanner;  
public class Auto extends Vehicle  
{  
    int numWheels=3;  
    public void show()  
    {  
        Scanner sc=new Scanner(System.in);  
        System.out.println("Press 1 to move Forward, 2 to move Backward");  
        int m=sc.nextInt();  
        if(m==1)  
            super.moveforward();  
        else  
            super.movebackward();  
        System.out.print("I am a Auto and I am travelling on ");  
    }  
    public void disp()  
    {  
        System.out.println(super.honk());  
    }  
    public void numofWheels()  
    {  
        System.out.println(numWheels+" wheels");  
    }  
}
```

**Car.java**

```
import java.util.Scanner;  
public class Car extends Vehicle  
{  
    int numWheels=4;  
    public void show()  
    {  
        Scanner sc=new Scanner(System.in);  
        System.out.println("Press 1 to move Forward, 2 to move Backward");  
        int m=sc.nextInt();  
        if(m==1)  
            super.moveforward();  
        else  
            super.movebackward();  
        System.out.print("I am a Car and I am driving on ");  
    }  
    public void disp()  
    {  
        System.out.println(super.honk());  
    }  
}
```

## OBJECT ORIENTED PROGRAMMING LAB

```

    public void numofWheels()
    {
        System.out.println(numWheels+" wheels");
    }
}

```

**Bicycle.java**

```

import java.util.Scanner;
public class Bicycle extends Vehicle
{
    int numWheels=2;
    public void show()
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("Press 1 to move Forward, 2 to move Backward");
        int m=sc.nextInt();
        if(m==1)
            super.moveforward();
        else
            super.movebackward();
        System.out.print("I am a Bicycle and I am pedalling on ");
    }
    public void disp()
    {
        System.out.println(super.honk());
    }
    public void numofWheels()
    {
        System.out.println(numWheels+" wheels");
    }
}

```

**Move.java <<interface>>**

```

public interface Move//interface
{
    public void moveforward();
    public void movebackward();
}

```

## OBJECT ORIENTED PROGRAMMING LAB

**VirtualWorldManager.java**

```

import java.util.Scanner;
public class VirtualWorldManager
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        int flag=0,ch1,ch2,ch3;
        while(flag==0)
        {
            System.out.println("Select the entity you want to work upon");
            System.out.println("Press 1 for Human Being, 2 for Vehicle, 3 to Exit!");
            ch1=sc.nextInt();
            switch(ch1)
            {
                case 1://For Human Being
                    System.out.println("Press 1 for Person, 2 for Student, 3 for CSE Student");
                    ch2=sc.nextInt();
                    switch(ch2)
                    {
                        case 1:
                            Person p=new Person();
                            p.show();
                            break;
                        case 2:
                            Student s=new Student();
                            s.show();
                            break;
                        case 3:
                            CSEStudent c=new CSEStudent();
                            c.show();
                            break;
                        default://Invalid Inputs
                            System.out.println("Invalid Input");
                    }
                    break;
                case 2://For Vehicle
                    System.out.println("Press 1 for Auto, 2 for Car, 3 for Bicycle");
                    ch2=sc.nextInt();
                    switch(ch2)
                    {
                        case 1:
                            Auto a=new Auto();
                            System.out.println("Press 1 to move, 2 to Honk");
                            ch3=sc.nextInt();
                            if(ch3==1)
                            {
                                a.show();
                                a.numofWheels();
                            }
                    }
            }
        }
    }
}

```

## OBJECT ORIENTED PROGRAMMING LAB

```

        }
        else
            a.disp();
        break;
    case 2:
        Car cr=new Car();
        System.out.println("Press 1 to move, 2 to Honk");
        ch3=sc.nextInt();
        if(ch3==1)
        {
            cr.show();
            cr.numofWheels();
        }
        else
            cr.disp();
        break;
    case 3:
        Bicycle b=new Bicycle();
        System.out.println("Press 1 to move, 2 to Honk");
        ch3=sc.nextInt();
        if(ch3==1)
        {
            b.show();
            b.numofWheels();
        }
        else
            b.disp();
        break;
    default: //Invalid Inputs
        System.out.println("Invalid Input");
    }
    break;
case 3://For Termination
    flag=1;
    break;
default://Invalid Inputs
    System.out.println("Invalid Input");
}
}
}
}

```

## OBJECT ORIENTED PROGRAMMING LAB

**OUTPUT :**

```

run:
Select the entity you want to work upon
Press 1 for Human Being, 2 for Vehicle, 3 to Exit!
1
Press 1 for Person, 2 for Student, 3 for CSE Student
2
Press 1 to move Forward, 2 to move Backward
1
I am a Student and I walk!
Enter the distance to travel
54
I moved forward by 54 kms
Select the entity you want to work upon
Press 1 for Human Being, 2 for Vehicle, 3 to Exit!
2
Press 1 for Auto, 2 for Car, 3 for Bicycle
3
Press 1 to move, 2 to Honk
2
HONK!
Select the entity you want to work upon
Press 1 for Human Being, 2 for Vehicle, 3 to Exit!
1
Press 1 for Person, 2 for Student, 3 for CSE Student
1
Press 1 to move Forward, 2 to move Backward
2
I am a Person and I walk!
Enter the distance to travel
3
I moved backward by 3 kms
Select the entity you want to work upon
Press 1 for Human Being, 2 for Vehicle, 3 to Exit!

```

1. What happens when the method introduce() is declared as “final” in the Person class? Explain.

**Answer:** We get compile-time error. This is because the introduce() function in the other two classes are non-final and hence they cannot override the introduce() function in Person class.

2. What happens when the String constant HOBBY is declared as “final” in the Person class? Explain.

**Answer:** Since the variable hobby is declared final inside the Person class, its value won't get changed inside that class. But it will be changed in the other two child class. Hence the program gives the same output, without any errors.

3. If the attribute name were declared with package accessibility (#) in the Person class, how can you access it from the child-classes. Can you do the same when name is declared as private? Which is the better way? Explain.

**Answer:** If the name was in a different package, we had to import that package from the child classes where the name attribute was used. But if name is declared private and present in a different package, then it cannot be accessed by other classes outside that package or even outside that class. So, the better way is to declare it in a different package.