**SWE3005**

Principles of Design Patterns

**Submitted by**

**ANKAM SRINIVAS**

**(21MIS7009)**

***To***

***DR. B.V.GOKULNATH***



**VIT-AP UNIVERSITY , AMARAVATHI**

**ANDHRA PRADESH**

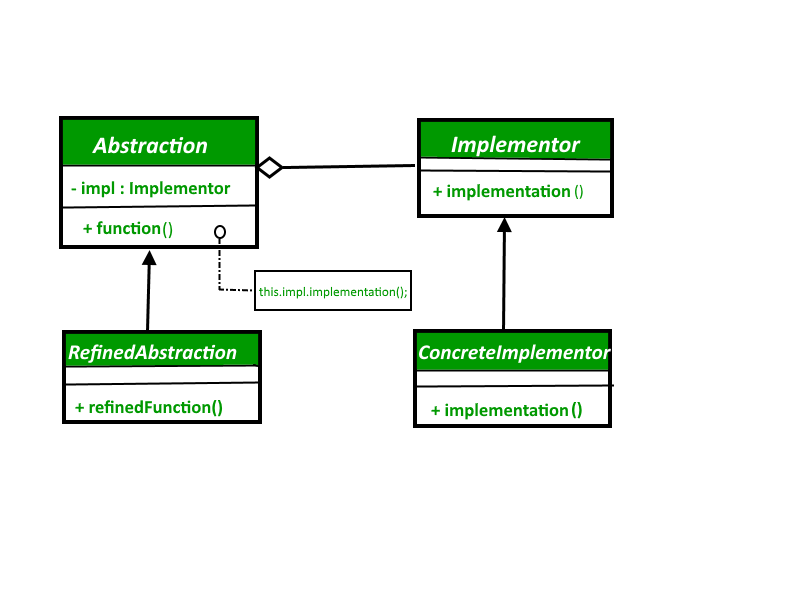
**LAB – 07**

**Question :**

1. **Java code to demonstrate Bridge design pattern .**

**Solution :-**

**UML Diagram of Bridge Design Pattern :**



**Code:**

// Java code to demonstrate bridge design pattern

abstract class Vehicle {

    protected Workshop workShop1;

    protected Workshop workShop2;

    protected Vehicle(Workshop workShop1, Workshop workShop2)

    {

        this.workShop1 = workShop1;

        this.workShop2 = workShop2;

    }

    abstract public void manufacture();

}

class Car extends Vehicle {

    public Car(Workshop workShop1, Workshop workShop2)

    {

        super(workShop1, workShop2);

    }

    @Override

    public void manufacture()

    {

        System.out.print("Car ");

        workShop1.work();

        workShop2.work();

    }

}

class Bike extends Vehicle {

    public Bike(Workshop workShop1, Workshop workShop2)

    {

        super(workShop1, workShop2);

    }

    @Override

    public void manufacture()

    {

        System.out.print("Bike ");

        workShop1.work();

        workShop2.work();

    }

}

interface Workshop

{

    abstract public void work();

}

class Produce implements Workshop {

    @Override

    public void work()

    {

        System.out.print("Produced");

    }

}

class Assemble implements Workshop {

    @Override

    public void work()

    {

        System.out.print(" And");

        System.out.println(" Assembled.");

    }

}

class Bridge Pattern {

    public static void main(String[] args)

    {

  System.out.println(" NAME   : Ankam Srinivas ");

        System.out.println(" REG.NO : 21MIS7009 ");

        Vehicle vehicle1 = new Car(new Produce(), new Assemble());

        vehicle1.manufacture();

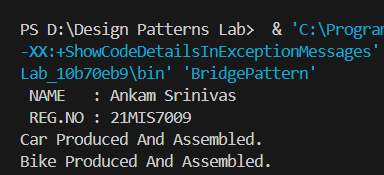
        Vehicle vehicle2 = new Bike(new Produce(), new Assemble());

        vehicle2.manufacture();

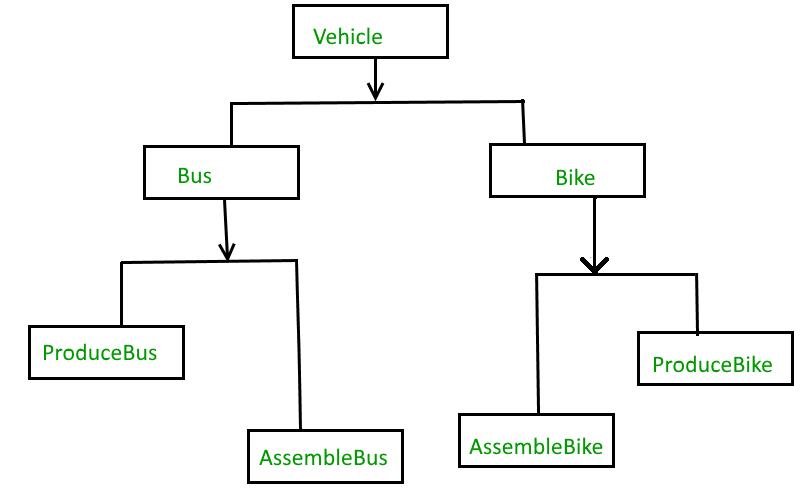
    }

}

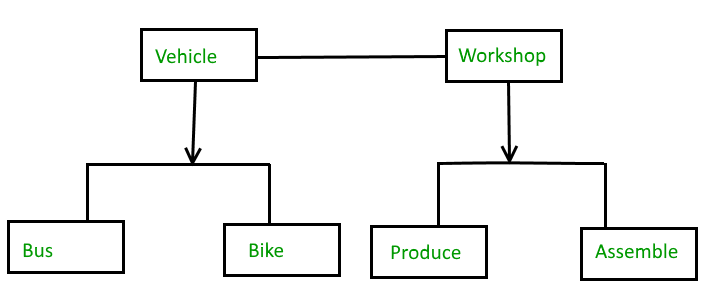
**Output:**



**Without Bridge Design Pattern** :

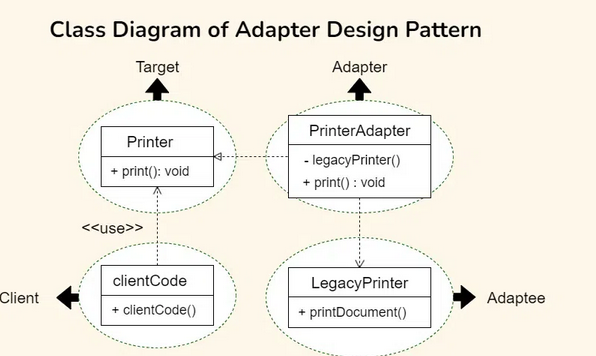


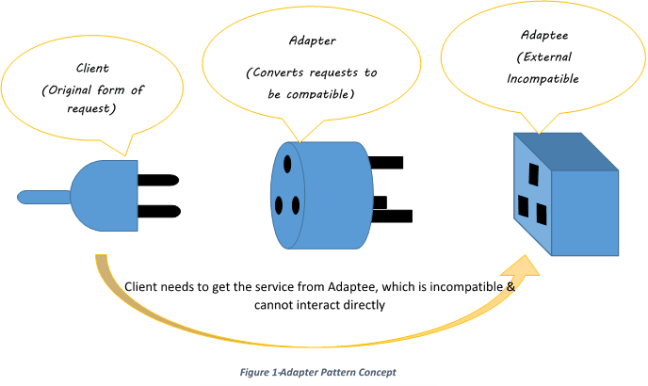
**With Bridge Design Pattern :**



1. **Java code to demonstrate Adapter design pattern .**

**UML Diagram of Adapter design pattern :**

****



**Code:**

// Target Interface

interface Printer {

    void print();

}

// Adaptee

class LegacyPrinter {

    void printDocument() {

        System.out.println("Legacy Printer is printing a document.");

    }

}

// Adapter

class PrinterAdapter implements Printer {

    private LegacyPrinter legacyPrinter = new LegacyPrinter();

    @Override

    public void print() {

        legacyPrinter.printDocument();

    }

}

// Client Code

class Client {

    static void clientCode(Printer printer) {

        printer.print();

    }

    public static void main(String[] args) {

        // Using the Adapter

  System.out.println(" NAME   : Ankam Srinivas ");

        System.out.println(" REG.NO : 21MIS7009 ");

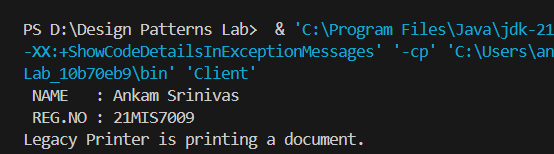
        PrinterAdapter adapter = new PrinterAdapter();

        clientCode(adapter);

    }

}

**Output:**

****

******