**WEEK – 1**

**Design principles & Patterns**

**1.** **Implementing the Singleton Pattern**

class LazySingleton{

    private static LazySingleton instance;

 private LazySingleton(){

        System.out.println("LazySingleton Instance Created");

    }

public static LazySingleton getInstance(){

        if (instance==null){

            instance=new LazySingleton();

        }

        return instance;

    }

public void show(){

        System.out.println("Lazy singleton");

    }

}

class EagerSingleton{

    private static final EagerSingleton instance=new EagerSingleton();

 private EagerSingleton(){

        System.out.println("EagerSingleton Instance Created");

    }

public static EagerSingleton getInstance(){

        return instance;

    }

    public void show(){

        System.out.println("Eager singleton");

    }

}

public class SingletonPattern{

    public static void main(String[] args){

        LazySingleton l1=LazySingleton.getInstance();

        LazySingleton l2=LazySingleton.getInstance();

        l1.show();

        System.out.println("Lazy Singleton same instance? "+(l1==l2));

EagerSingleton e1=EagerSingleton.getInstance();

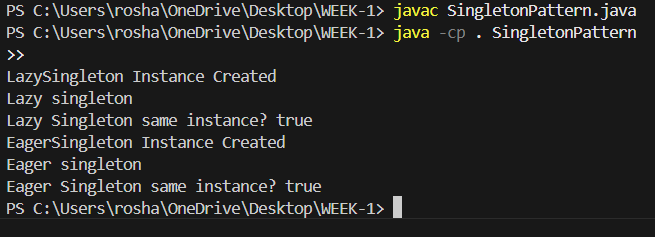
        EagerSingleton e2=EagerSingleton.getInstance();

        e1.show();

        System.out.println("Eager Singleton same instance? "+(e1==e2));

}}

**Output**



**2. Implementing the Factory Method Pattern**

interface Shape {

    void draw();

}

class Circle implements Shape{

    public void draw(){

        System.out.println("Circle");

    }

}

class Rectangle implements Shape{

    public void draw(){

        System.out.println("Rectangle");

    }

}

class Square implements Shape{

    public void draw(){

        System.out.println("Square");

    }

}

class Shapefun{

    public Shape getShape(String type){

        if (type==null) return null;

        switch(type){

            case "CIRCLE":

                return new Circle();

            case "RECTANGLE":

                return new Rectangle();

            case "SQUARE":

                return new Square();

            default:

                return null;

        }}}

public class FactoryMethodPattern{

    public static void main(String[] args){

        Shapefun obj=new Shapefun();

        Shape s1=obj.getShape("CIRCLE");

        s1.draw();

        Shape s2=obj.getShape("RECTANGLE");

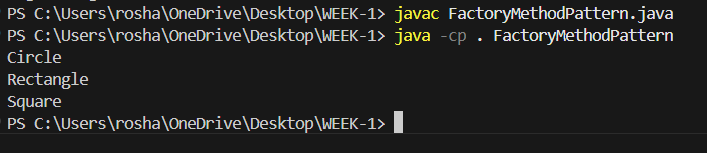
        s2.draw();

        Shape s3=obj.getShape("SQUARE");

        s3.draw();

    }}

**Output**

****