**[Digital-Nurture-4.0-JavaFSE](https://github.com/seshadrimr/Digital-Nurture-4.0-JavaFSE/tree/main) Week 1 Exercise:**

Design patterns and principles:

**1.Implementing the singleton pattern**

**Code:**

public class Singleton {

private Singleton() {

System.out.println("Singleton instance created");

}

private static class Holder {

private static final Singleton INSTANCE = new Singleton();

}

public static Singleton getInstance() {

return Holder.INSTANCE;

}

public static void main(String[] args) {

Singleton obj1 = Singleton.getInstance();

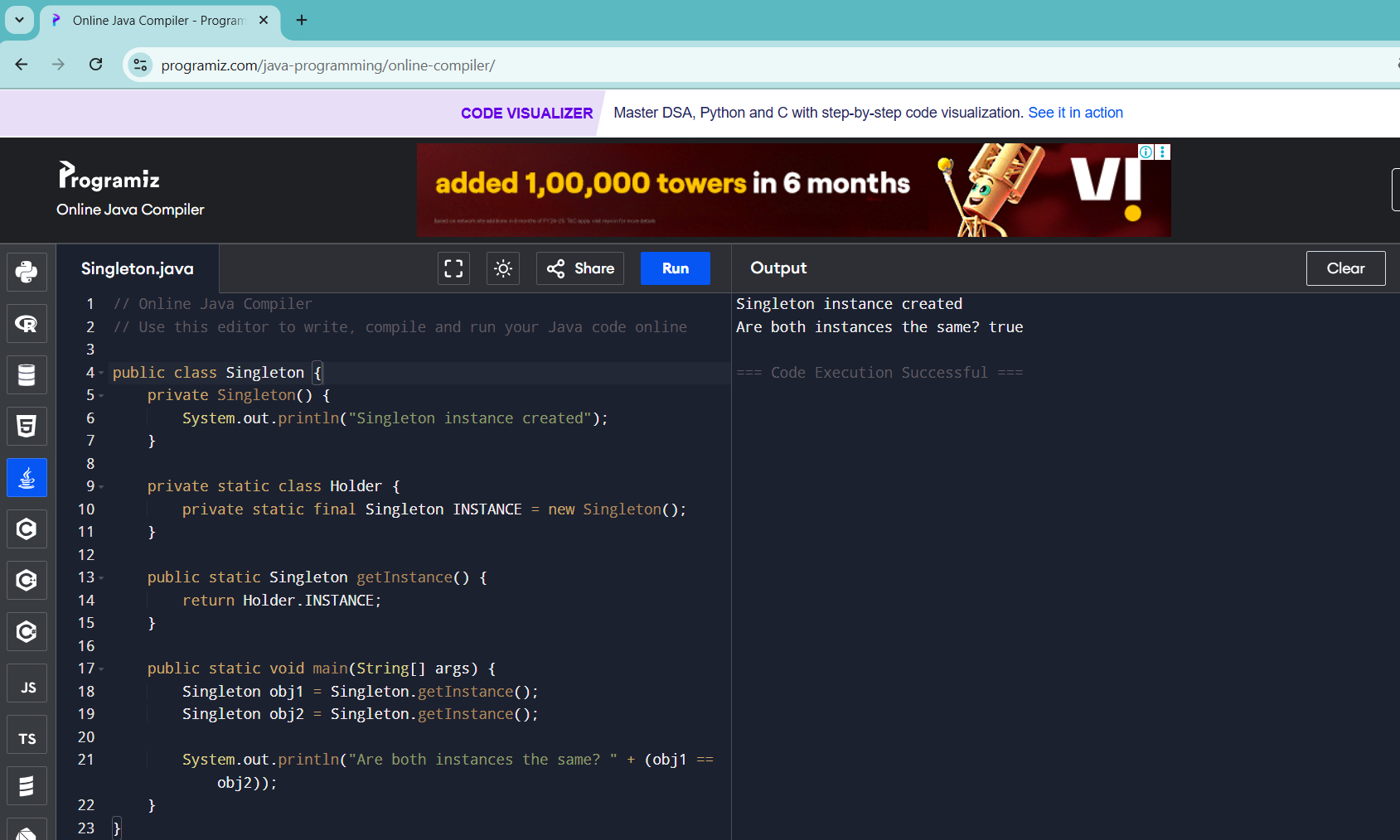
Singleton obj2 = Singleton.getInstance();

System.out.println("Are both instances the same? " + (obj1 == obj2));

}

}

**Output:**

****

**2.Implementing the factory method pattern:**

**Code:**

interface Shape {

void draw();

}

// Concrete classes

class Circle implements Shape {

public void draw() {

System.out.println("Drawing a Circle");

}

}

class Square implements Shape {

public void draw() {

System.out.println("Drawing a Square");

}

}

// Factory class

class ShapeFactory {

public Shape getShape(String type) {

if (type == null) return null;

if (type.equalsIgnoreCase("CIRCLE")) return new Circle();

if (type.equalsIgnoreCase("SQUARE")) return new Square();

return null;

}

}

// Public class with main method

public class ShapeFactoryDemo {

public static void main(String[] args) {

ShapeFactory factory = new ShapeFactory();

Shape shape1 = factory.getShape("circle");

shape1.draw(); // Output: Drawing a Circle

Shape shape2 = factory.getShape("square");

shape2.draw(); // Output: Drawing a Square

}

}

**Output:**

