**Exercise 6: Implementing the Proxy Pattern**

**Scenario:**

You are developing an image viewer application that loads images from a remote server. Use the Proxy Pattern to add lazy initialization and caching.

ProxyPatternExample/

├── Image.java

├── RealImage.java

├── ProxyImage.java

└── Main.java

**Image.java :-**

public interface Image {

void display();

}

**RealImage.java :-**

public class RealImage implements Image {

private final String fileName;

public RealImage(String fileName) {

this.fileName = fileName;

loadFromRemoteServer();

}

private void loadFromRemoteServer() {

System.out.println("🔄 Loading image from remote server: " + fileName);

// Simulate loading delay

try {

Thread.sleep(1000);

} catch (InterruptedException ignored) {}

}

@Override

public void display() {

System.out.println("🖼️ Displaying image: " + fileName);

}

}

**ProxyImage.java :-**

public class ProxyImage implements Image {

private final String fileName;

private RealImage realImage;

public ProxyImage(String fileName) {

this.fileName = fileName;

}

@Override

public void display() {

if (realImage == null) {

realImage = new RealImage(fileName); // Lazy initialization

}

realImage.display(); // Cached image is used after first load

}

}

**Main.java :-**

public class Main {

public static void main(String[] args) {

Image img1 = new ProxyImage("nature.jpg");

Image img2 = new ProxyImage("cityscape.png");

// First time loads from remote server

img1.display();

System.out.println("\n---\n");

// Second time uses cached image

img1.display();

System.out.println("\n---\n");

img2.display();

}

}

**Output :-**

