**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.

**File Structure:**

1. Image.java – Interface
2. RealImage.java – Real subject
3. ProxyImage.java – Proxy class with lazy loading
4. TestProxy.java – Main test class

**Codes:**

//Image.java

public interface Image {

void display();

}

//RealImage.java

public class RealImage implements Image {

private String fileName;

public RealImage(String fileName) {

this.fileName = fileName;

loadFromRemoteServer();

}

private void loadFromRemoteServer() {

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

}

@Override

public void display() {

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

}

}

// ProxyImage.java

public class ProxyImage implements Image {

private RealImage realImage;

private String fileName;

public ProxyImage(String fileName) {

this.fileName = fileName;

}

@Override

public void display() {

if (realImage == null) {

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

}

realImage.display();

}

}

//TestProxy.java

public class TestProxy {

public static void main(String[] args) {

Image image1 = new ProxyImage("photo1.jpg");

Image image2 = new ProxyImage("photo2.jpg");

System.out.println("--- First call to image1 ---");

image1.display();

System.out.println("--- Second call to image1 ---");

image1.display();

System.out.println("--- First call to image2 ---");

image2.display();

}

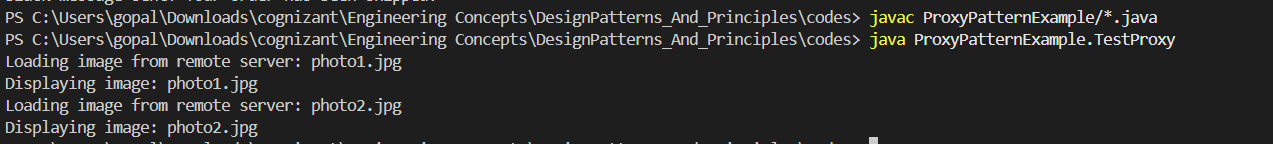
}

**Compile and Run:**

javac ProxyPatternExample/\*.java

java ProxyPatternExample.TestProxy

**Output:**

****