Exercise 1: Implementing the Singleton Pattern

class Logger {

private static Logger instance;

private Logger() {}

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

public void log(String message) {

System.out.println("Log: " + message);

}

}

public class SingletonPatternExample {

public static void main(String[] args) {

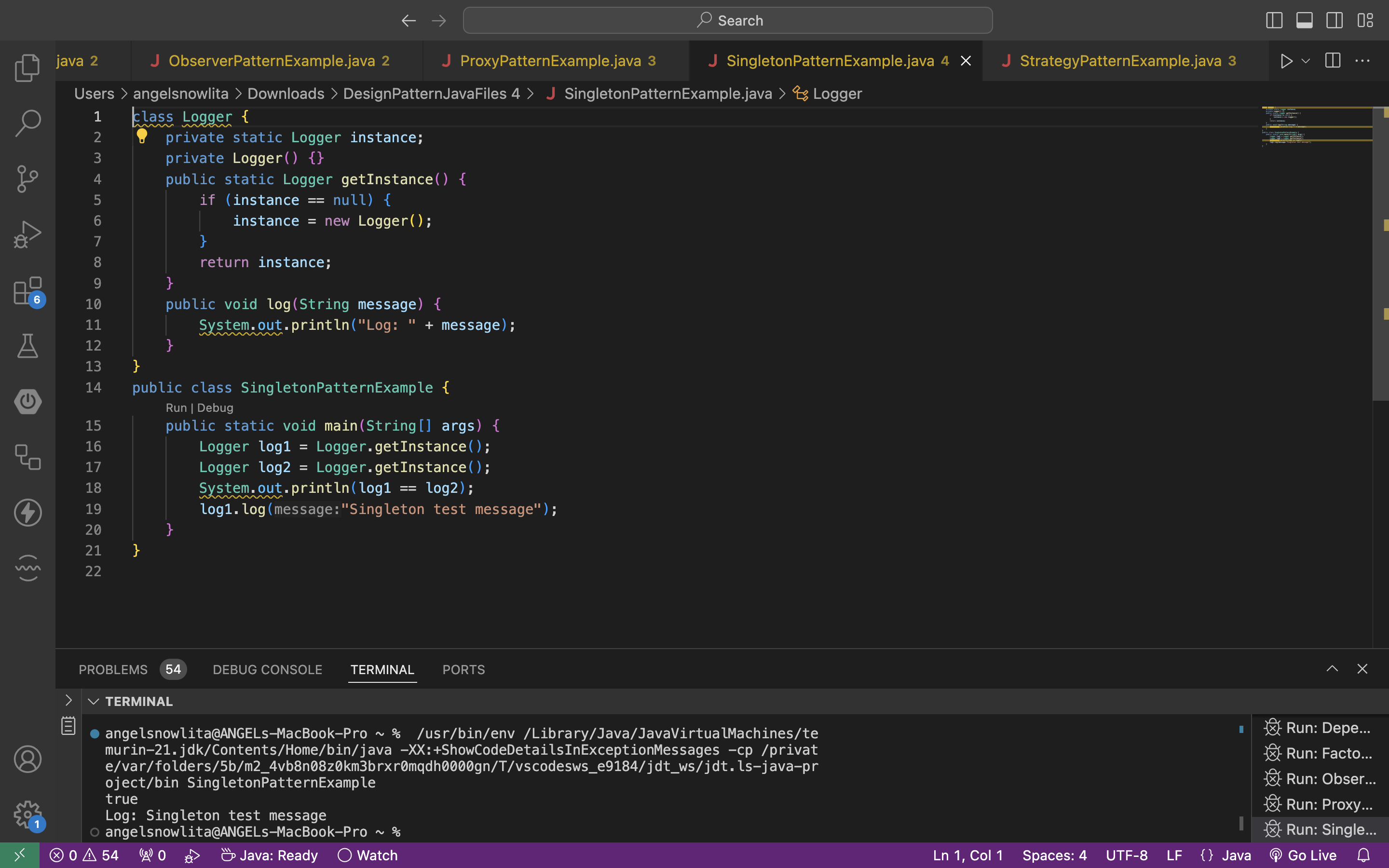
Logger log1 = Logger.getInstance();

Logger log2 = Logger.getInstance();

System.out.println(log1 == log2);

log1.log("Singleton test message");

}

}

EXERCISE 2: IMPLEMENTING THE FACTORY METHOD PATTERN

interface Document {

void open();

}

class WordDocument implements Document {

public void open() {

System.out.println("Opening Word document");

}

}

class PdfDocument implements Document {

public void open() {

System.out.println("Opening PDF document");

}

}

class ExcelDocument implements Document {

public void open() {

System.out.println("Opening Excel document");

}

}

abstract class DocumentFactory {

abstract Document createDocument();

}

class WordFactory extends DocumentFactory {

Document createDocument() {

return new WordDocument(); }

}

class PdfFactory extends DocumentFactory {

Document createDocument() {

return new PdfDocument();

}

}

class ExcelFactory extends DocumentFactory {

Document createDocument() {

return new ExcelDocument();

}

}

public class FactoryMethodPatternExample {

public static void main(String[] args) {

DocumentFactory factory = new PdfFactory();

Document doc = factory.createDocument();

doc.open();

}

}

