**Design Patterns and Principles - Hands-On**

**Exercise 1: Singleton Pattern**

***→ Main file (MainSingleton.java)***

public class MainSingleton {  
 public static void main(String[] args) {  
 Logger a = Logger.get();  
 Logger b = Logger.get();  
 a.log("x");  
 b.log("y");  
 System.out.println(a == b ? "ok" : "no");  
 }  
}

***→ Singleton class (Logger.java)***

public class Logger {  
 private static Logger l;  
 private Logger() {  
 System.out.println("init");  
 }  
 public static Logger get() {  
 if (l == null) l = new Logger();  
 return l;  
 }  
 public void log(String m) {  
 System.out.println(m);  
 }  
}

**Exercise 2: Factory Method Pattern**

***→ Factory main file (MainFactory.java)***

public class MainFactory {  
 public static void main(String[] args) {  
 Make m = new MakePdf();  
 Doc d = m.get();  
 d.open();  
 }  
}

***→ Document interface (Doc.java)***

interface Doc {  
 void open();  
}

***→ PDF file class (Pdf.java)***

public class Pdf implements Doc {  
 public void open() {  
 System.out.println("pdf");  
 }  
}

***→ Word file class (Word.java)***

public class Word implements Doc {  
 public void open() {  
 System.out.println("word");  
 }  
}

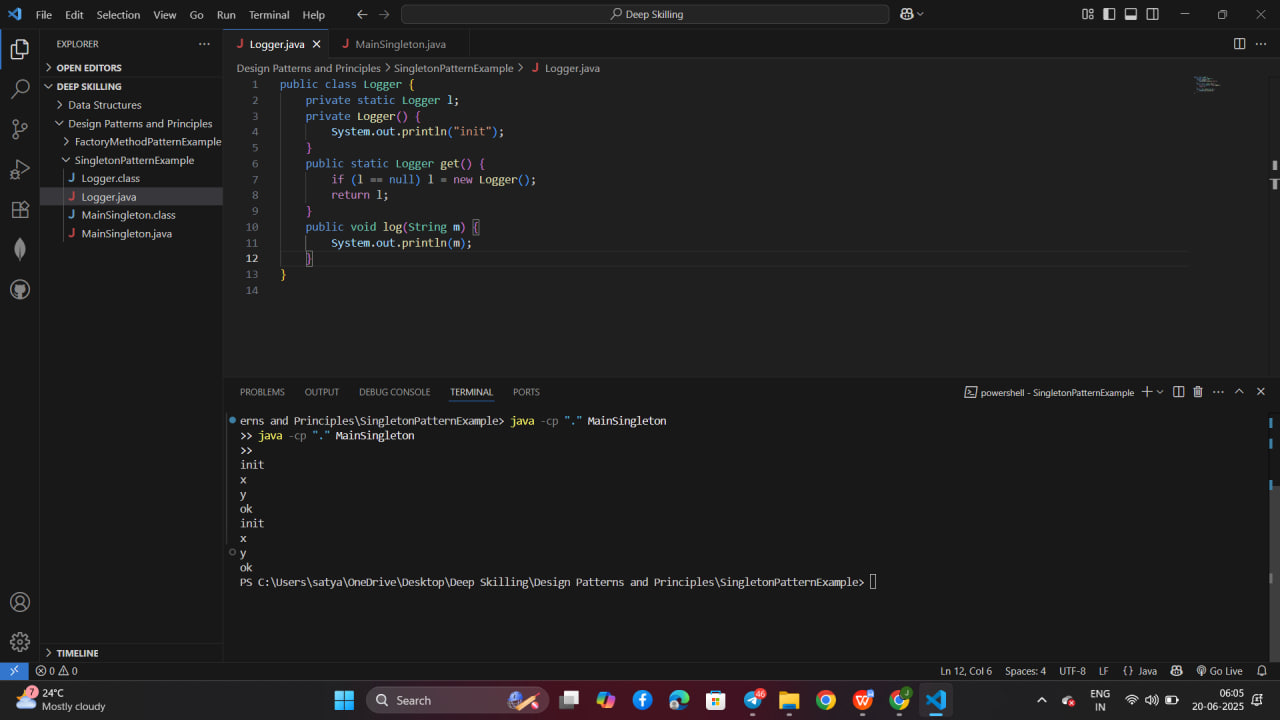
***→ Abstract factory (Make.java)***

abstract class Make {  
 abstract Doc get();  
}

***→ PDF factory class (MakePdf.java)***

public class MakePdf extends Make {  
 public Doc get() {  
 return new Pdf();  
 }  
}

***Output Preview for Exercise 1:***



***Output Preview for Exercise 2:***

