**Digital Nurture 4.0  
Superset ID- 6397694**  
  
Week 1- DESIGN PATTERNS AND PRINCIPLES

**Exercise 1: Implementing the Singleton Pattern**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Solution**  
  
**1.Logger class**

**public** **class** Logger {

**private** **static** Logger *instance*;

**public** Logger() {

System.***out***.println("singleton pattern Logger instance created.");

}

**public** **static** Logger getInstance() {

**if** (*instance* == **null**) {

*instance* = **new** Logger();

}

**return** *instance*;

}

**public** **void** log(String message) {

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

}

}

**2.Logger Test Class**

**public** **class** LoggerTest {

**public** **static** **void** main(String[] args) {

Logger logger1 = Logger.*getInstance*();

Logger logger2 = Logger.*getInstance*();

logger1.log("This is the first object.");

logger2.log("This is the second object.");

**if** (logger1 == logger2) {

System.***out***.println("Both logger1 and logger2 refer to the same instance.");

}

**else** {

System.***out***.println("Different instances exist! Singleton pattern failed.");

}

}

}

**Output**



