**Exercise 1 : Implementing the Singleton Pattern :**

Code :

Logger.java :

package SingletonPatternExample;

class Logger {

private static Logger instance;

private Logger()

{

}

public static Logger get()

{

if(instance==null)

{

instance=new Logger();

}

return instance;

}

}

testclass.java :

package SingletonPatternExample;

class testclass {

public static void main(String[] args) {

//Logger obj=new Logger(); //we cannot create instance of Logger class from another class since it is a private static instance of the Logger class

Logger obj1=Logger.get();

Logger obj2=Logger.get();

System.out.println(obj1);//both the objects are referring

System.out.println(obj2);//to the same instance of Logger class

}

}

Output :



**Exercise 2 : Implementing the Factory Method Pattern :**

Code :

WordDocument.java :

package FactoryMethodPatternExample;

public interface WordDocument {

void storedata();

}

PdfDocument.java :

package FactoryMethodPatternExample;

public interface PdfDocument{

void storedata();

}

ExcelDocument.java :

package FactoryMethodPatternExample;

public interface ExcelDocument{

void storedata();

}

word.java :

package FactoryMethodPatternExample;

public class word implements WordDocument {

public void storedata()

{

System.out.println("Storing in Word Document");

}

}

pdf.java :

package FactoryMethodPatternExample;

public class pdf implements PdfDocument{

public void storedata()

{

System.out.println("Storing in pdf document");

}

}

excel.java :

package FactoryMethodPatternExample;

public class excel implements ExcelDocument{

public void storedata()

{

System.out.println("Storing in excel document");

}

}

DocumentFactory.java :

package FactoryMethodPatternExample;

public abstract class DocumentFactory<T> {

public abstract T createDocument(String type);

}

wd.java :

package FactoryMethodPatternExample;

public class wd extends DocumentFactory{

public WordDocument createDocument(String type)

{

if(type.equalsIgnoreCase("word"))

{

return new word();

}

return null;

}

}

pd.java :

package FactoryMethodPatternExample;

public class pd extends DocumentFactory{

public PdfDocument createDocument(String type)

{

if(type.equalsIgnoreCase("pdf"))

{

return new pdf();

}

return null;

}

}

ed.java :

package FactoryMethodPatternExample;

public class ed extends DocumentFactory {

public ExcelDocument createDocument(String type)

{

if(type.equalsIgnoreCase("excel"))

{

return new excel();

}

return null;

}

}

usage.java :

package FactoryMethodPatternExample;

public class usage {

public static void main(String[] args)

{

wd obj1=new wd();

WordDocument w=obj1.createDocument("word");

w.storedata();//Output: Storing in Word Document

pd obj2=new pd();

PdfDocument p=obj2.createDocument("pdf");

p.storedata();//Output: Storing in pdf document

ed obj3=new ed();

ExcelDocument e=obj3.createDocument("excel");

e.storedata();//Output: Storing in excel document

}

}

Output :

