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

**Code :->**

class ExcelDocument implements Document

{

}

class WordDocument implements Document

{

}

public class PdfDocumentFactory implements DocumentFactory {

public Document createDocument() {

return new PDFDocument();

}

}

public interface DocumentFactory

{

Document createDocument();

}

class WordDocumentFactory implements DocumentFactory

{

//Document createDocument()

//be default, all methods of interfaces are public to be accessible to the outsie worl

//hence implementations must be mae explicitly public rather than package-private

public Document createDocument()

{

return new WordDocument();

}

}

class ExcelDocumentFactory implements DocumentFactory {

public Document createDocument() {

return new ExcelDocument();

}

}

interface Document

{

}

class PDFDocument implements Document

{

}

import java.io.FileNotFoundException;

class Service1

{

static Document createDocument(String path) throws FileNotFoundException

{

DocumentFactory df;

if(path.endsWith(".pdf"))

{

df = new PdfDocumentFactory();

}

else if(path.endsWith(".excel"))

{

df = new ExcelDocumentFactory();

}

else if(path.endsWith(".word"))

{

df = new WordDocumentFactory();

}

else

{

throw new FileNotFoundException("Invalid file extension.");

//Wow, when you do throw

//There is just no compulsory need of df initializing!

}

//rather than if-else

//can add a Hashmap; - later!!

return df.createDocument();

}

}

import java.util.ArrayList;

class ClientTest1

{

static ArrayList<String> test\_paths = new ArrayList<>();

public static void main()

{

test\_paths.add("user/documents/admit\_card.word");

test\_paths.add("adhaar\_card.pdf");

test\_paths.add("user/documents/marksheet\_card.excel");

test\_paths.add("random\_invalid");

int t = test\_paths.size();

for(int i = 0;i<t;i++)

{

try

{

Service1.createDocument(test\_paths.get(i));

}

catch (java.io.FileNotFoundException fnfe)

{

fnfe.printStackTrace();

}

}

}

}

**Output:**

