TASK 1 – DEBIT CARD (FACTORY PATTERN)

BANKFACTORY.JAVA

```
import com.card.Classic;
public class BankFactory {
public DebitCard getInstance(String str)
{
if(str.equals("One"))
    {
        return new Classic();
    }
else
{
    return new Gold();
}
}
}
```

CLIENT.JAVA

```
import com.card.DebitCard;
import com.card.Gold;
import com.card.Classic;
public class Client {
public static void main(String[] args)
{
       BankFactory BK = new BankFactory();
       DebitCard obj = BK.getInstance("One");
       obj.getCardType();
       obj.getCardLimit();
}
}
```

CLASSIC.JAVA

```
package com.card;

public class Classic implements DebitCard {
@Override
public void getCardType()
{
       System.out.println("Type: Classic");
}
@Override
public void getCardLimit()
{
       System.out.println("Limit: 50,000");
}
}
```

GOLD.JAVA

```java
package com.card;
public class Gold implements DebitCard{
@Override
public void getCardType()
{
	System.out.println("Type: Gold");
}
@Override
public void getCardLimit()
{
	System.out.println("Limit: 100,000");
}}
```

DEBITCARD.JAVA

```java
package com.card;
public interface DebitCard {
	void getCardType();
	void getCardLimit();
}
```

OUTPUT

```
<terminated> Client (2) [Java Application] C:\Program Files\Java\jre1.8.0_281\bin\javaw.exe  (May 30, 2021 5:57:
Type: Classic
Limit: 50,000
```

TASK 2 – LAPTOP (FACTORY PATTERN)

CLIENT.JAVA

```
import com.LAPTOP.Dell;
import com.LAPTOP.HP;
import com.LAPTOP.Laptop;
import com.LAPTOP.Lenovo;

public class Client {
	public static void main(String[] args)
	{
		LaptopFactory LF = new LaptopFactory();
		Laptop obj = LF.getInstance("HP");
		obj.Performance();
		obj.Specs();
		obj.Price();
	}
}
```

LAPTOPFACTORY.JAVA

```
import com.LAPTOP.Dell;
import com.LAPTOP.HP;
import com.LAPTOP.Laptop;
import com.LAPTOP.Lenovo;

public class LaptopFactory {
public Laptop getInstance(String str)
{
if (str.equals("Dell")) {
return new Dell();
}
else if (str.equals("HP")) {
return new HP();
}
else
return new Lenovo();
}}
```

DELL.JAVA

```
package com.LAPTOP;
public class Dell implements Laptop{
@Override
public void Performance(){
	System.out.println("Performance: Very Fast.");}
@Override
public void Specs(){
	System.out.println("Specification: Core i5, 4GB RAM.");}
@Override
public void Price(){
	System.out.println("Price: 98,000.");}
}
```

LENOVO.JAVA

```java
package com.LAPTOP;
public class Lenovo implements Laptop{
@Override
public void Performance()
{
	System.out.println("Performance: Great.");
}
@Override
public void Specs()
{
	System.out.println("Specification: Core i3, 8GB RAM.");
}
@Override
public void Price()
{
	System.out.println("Price: 80,000.");
}
}
```

HP.JAVA

```java
package com.LAPTOP;
public class HP implements Laptop{
@Override
public void Performance(){
	System.out.println("Performance: Amazing.");}
@Override
public void Specs(){
	System.out.println("Specification: Core i5, 8GB RAM.");}
@Override
public void Price(){
	System.out.println("Price: 104,000.");
}}
```

LAPTOP.JAVA

```java
package com.LAPTOP;
public interface Laptop {
	void Performance();
	void Specs();
	void Price();
}
```

OUTPUT

```java
import com.LAPTOP.Dell;

public class Client {
    public static void main(String[] args)
    {
        LaptopFactory LF = new LaptopFactory();
        Laptop obj = LF.getInstance("HP");
        obj.Performance();
        obj.Specs();
        obj.Price();
    }
}
```

```
<terminated> Client (3) [Java Application] C:\Program Files\Java\jre1.8.0_281\bin\javaw.exe  (May 30, 2021 6:28:
Performance: Amazing.
Specification: Core i5, 8GB RAM.
Price: 104,000.
```

TASK 3 – RAR(S) (ADAPTER PATTERN)

CLIENT.JAVA

```
public class Client {
public static void main(String[] args) {
              JavaCodeRar rarFile = new JavaCodeRar();
              rarFile.OpenCode();
              rarFile.UpdateCode();
              JavaCodeTar tarFile = new JavaCodeTar();
              tarFile.UnlockCode();
              tarFile.ChangeCode();
              ZipCodeExtractor zipFile= new FileAdapter(tarFile);
              zipFile.OpenCode();
              zipFile.UpdateCode();}
}
```

JAVACODERAR.JAVA

```
public class JavaCodeRar implements ZipCodeExtractor
{
       public void OpenCode(){
              System.out.println("Opening Rar........");}
       public void UpdateCode(){
              System.out.println("Updating Rar........\n");}
}
```

JAVACODETAR.JAVA

```
public class JavaCodeTar
{
       public void UnlockCode(){
              System.out.println("Unlocking Tar........");}
       public void ChangeCode(){
              System.out.println("Changing Tar........");}
}
```

FILEADAPTER.JAVA

```
public class FileAdapter implements ZipCodeExtractor
{
       JavaCodeTar tar;
       public FileAdapter(JavaCodeTar tar){
              this.tar=tar;}
       public void OpenCode() {
              tar.UnlockCode(); }
       public void UpdateCode(){
              tar.ChangeCode();}
}
```

ZIPCODEEXTRACTOR.JAVA

```
public interface ZipCodeExtractor {
       public void OpenCode();
       public void UpdateCode(); }
```

OUTPUT

```java
public class Client {
public static void main(String[] args) {
        JavaCodeRar rarFile = new JavaCodeRar();
        rarFile.OpenCode();
        rarFile.UpdateCode();
        JavaCodeTar tarFile = new JavaCodeTar();
        tarFile.UnlockCode();
        tarFile.ChangeCode();
        ZipCodeExtractor zipFile= new FileAdapter(tarFile);
        zipFile.OpenCode();
        zipFile.UpdateCode();}
}
```

```
<terminated> Client (5) [Java App
Opening Rar........
Updating Rar........

Unlocking Tar........
Changing Tar........
Unlocking Tar........
Changing Tar........
```

TASK 4 – IMAGE VIEWER (ADAPTER PATTERN)

CLIENT.JAVA

```
public class Client {
	public static void main(String[] args) {
		JPGFiles jpg = new JPGFiles();
		jpg.JPG();
		jpg.open();
		PNGFiles png=new PNGFiles();
		png.PNG();
		png.open();
		GIF gif= new GIF();
		gif.open();
		ImageViewer iv = new GIFAdapter(gif);
		iv.open();
	}
}
```

GIF.JAVA

```
public class GIF
{
	public GIF(){
		System.out.println("GIF Loading..........");}
	public void open(){
		System.out.println("GIF Opened..........");}
}
```

GIFADAPTER.JAVA

```
public class GIFAdapter implements ImageViewer
{
	GIF gif=new GIF();
	public GIFAdapter(GIF gif){
		this.gif=gif;}
	public void open(){
		System.out.println("GIF Opened using Adapter..........");}
}
```

IMAGEVIEWER.JAVA

```
public interface ImageViewer {
	public void open();}
```

JPGFILES.JAVA

```
public class JPGFiles implements ImageViewer
{
	public void JPG(){
		System.out.println("JPG Loading..........");}
	public void open(){
		System.out.println("JPG Opened..........");}
}
```

PNGFILES.JAVA

```java
public class PNGFiles implements ImageViewer
{
	public void PNG(){
		System.out.println("PNG Loading..........");}
	public void open(){
		System.out.println("PNG opened..........");}
}
```

OUTPUT

```java
public class Client {
    public static void main(String[] args) {
        JPGFiles jpgFileObject = new JPGFiles();
        jpgFileObject.JPG();
        jpgFileObject.open();
        PNGFiles pngFileObject=new PNGFiles();
        pngFileObject.PNG();
        pngFileObject.open();
        GIF gifFileObjectObject= new GIF();
        gifFileObjectObject.open();
        ImageViewer ImageObject = new GIFAdapter(gifFileObject
        ImageObject.open();
    }
}
```

```
<terminated> Client (6) [Java Application] C:\Prograı
JPG Loading..........
JPG Opened..........
PNG Loading..........
PNG opened..........
GIF Loading..........
GIF Opened..........
GIF Loading..........
GIF Opened using Adapter..........
```

TASK 5 – SUBWAY SHOP (TEMPLATE PATTERN)

SUBWAYINTERFACE.JAVA

```
public abstract class SubwayInterface {
public SubwayInterface()
{
	super();
}
public abstract void prepareSubway();
public abstract void selectBread();
public abstract void addingVeggies();
public abstract void addingChicken();
public abstract void addingSauces();

public void makeSubway()
{
	prepareSubway();
	selectBread();
	addingVeggies();
	addingChicken();
	addingSauces();
}
@Override
public String toString() {
StringBuilder builder = new StringBuilder();
builder.append("Subway [Bread=").append("PitaBread").append(",Veggies=").append("No
Veggies").append(",Chicken=").append("Roasted").append(",
Sauces=").append("Jalpeno").append("]");
return builder.toString();
}
}
```

CLIENT.JAVA

```
public class Client {
public static void main(String[] args) {
SubwayInterface sub = new Veg();
sub.makeSubway();
if (sub != null) {
System.out.println("Below Subway delievered: ");
System.out.println("================================================================
=====");
System.out.println(sub);
System.out.println("================================================================
=====");
}
}

}
```

VEG.JAVA

```
public class Veg extends SubwayInterface {
@Override
public void prepareSubway()
{
	System.out.println("Preparing Your Order.....");
}
@Override
public void selectBread()
{
	System.out.println("Selecting Bread.....");
}
@Override
public void addingVeggies()
{
}
@Override
public void addingSauces()
{
	System.out.println("Adding Sauces.....");
}
@Override
public void addingChicken()
{
	System.out.println("Adding Roasted Chicken.....");
}
}
```

NONVEG.JAVA

```
public class NonVeg extends SubwayInterface {
@Override
public void prepareSubway()
{
	System.out.println("Preparing Your Order......");
}
@Override
public void selectBread()
{
	System.out.println("Selecting Bread.....");
}
@Override
public void addingVeggies()
{
	System.out.println("Adding Vegetables.....");
}
@Override
public void addingSauces()
{
	System.out.println("Adding Sauces.....");
}
@Override
public void addingChicken()
{}}
```

OUTPUT

```java
public class Client {
public static void main(String[] args) {
SubwayInterface sub = new Veg();
sub.makeSubway();
if (sub != null) {
System.out.println("Below Subway delievered: ");
System.out.println("=========================================
System.out.println(sub);
System.out.println("=========================================
}
}
```

```
<terminated> Client (4) [Java Application] C:\Program Files\Java\jre1.8.0_281\bin\javaw.exe  (May 30, 2021 9
Preparing Your Order.....
Selecting Bread.....
Adding Roasted Chicken.....
Adding Sauces.....
Below Subway delievered:
=====================================================================
Subway [Bread=PitaBread,Veggies=No Veggies,Chicken=Roasted, Sauces=Jalpeno]
=====================================================================
```