Why we should separate our BLC and ELC classes in java Application :

- If we write logic and main method in a single class than It is not a recommended approach because we cannot reuse our BLC classes.
- If our BLC classes are in a separate file then we can reuse the BLC classes in the same package and in different package also.

In a single .java file how many public classes we can write :

- In a single .java file, We can write **only one public class** and it must be our file name.

```
Case 1 :
-----------
Example.java
public class Example
{
}

Case 2 :
-----------
Test.java
-----------
public class Demo //Compilation error here file name must be Demo.java
{

}

Case 3 :
-----------
Sample.java
--------------
public class Sample   //Valid
{
}

class Alpha //Valid
{
}

public class Beta //Invalid
{
}
```

So the final conclusion is, In a single .java file we can decalre only class as public and it must be file name. In the same class we can also declare multiple non-public classes but it is not a recommended way because we cannot use these non public classes outside of the package OR in another package.

Note :WE SHOULD ALWAYS DECLARE OUR JAVA CLASSES WITH PUBLIC ACCESS MODIFIR I.E IN A SEPARATE FILE SO WE CAN REUSE THESE CLASSES ANYWHERE IN THE APPLICATION I.E **WORA**

How compiler will generate the .class file for any public OR non public classes ?

- Compiler will always generate 'n' number of .class files based on 'n' number of java classes whether it is public OR non public as shown below :

```
Sample.java
---------------
public class Sample
{
}

class Alpha
{
}

class Beta
{
}
```

Here 3 .class file will be created.

Working with static method and different return type :

- If a static method is present in ELC class then we can directly call the static method, On the other hand if the static method is available in the BLC class then we need to call with the help of class name.

//Program that shows how to reuse BLC class:

```
package com.ravi.blc;

//BLC
public class Rectangle
{
   public static void getAreaOfRectangle(int length, int breadth)
   {
       int area = length * breadth;
       System.out.println("Area of Rectangle is :"+area);

   }
}

package com.ravi.elc;

import java.util.Scanner;

import com.ravi.blc.Rectangle;

//ELC
public class Test1
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the Length of the Rectangle :");
        int length = sc.nextInt();
        System.out.print("Enter the breadth of the Rectangle :");
        int breadth = sc.nextInt();

        Rectangle.getAreaOfRectangle(length, breadth);
        sc.close();
    }

}
```

Note : Both the classes Rectangle.java and Test1.java are available in two different package.

WAP to accept a number from the user and find the following based on the different criteria :

a) If the given number is 0 OR -ve then return -1
b) If the given number is even then return the square of the number
c) For odd return cube of the number

```
package com.ravi.blc;

//BLC
public class Calculate
{
    public static int getSquareAndCube(int num)
    {
        if(num <=0)
        {
            return -1;
        }
        else if(num%2==0)
        {
            return (num*num);
        }
        else
        {
            return (num*num*num);
        }
    }
}
package com.ravi.elc;

import java.util.Scanner;

import com.ravi.blc.Calculate;

public class Test2 {

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a Number :");

        int num = sc.nextInt();

        System.out.println("Result is :"+Calculate.getSquareAndCube(num));
        sc.close();
    }
}
```

//WAP to find out the area of circle where radius must be double type and return value must be String type . If the radius is 0 OR -ve then return -1

```
package com.ravi.blc;

public class Circle
{
   public static String getAreaOfCircle(double radius)
   {
       if(radius <=0)
       {
           return ""+(-1);
       }
       else
       {
           final double PI = 3.14;
           double area = PI * radius * radius;
           return ""+area;
       }

   }
}

package com.ravi.elc;

import java.text.DecimalFormat;
import java.util.Scanner;

import com.ravi.blc.Circle;

public class Test3 {

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the value of Radius :");
        double radius = sc.nextDouble();

        String circle = Circle.getAreaOfCircle(radius);

        //Converting the String into double
        double areaOfCircle = Double.parseDouble(circle);

        DecimalFormat df = new DecimalFormat("0000.00");
        System.out.println("Area of circle is :"+df.format(areaOfCircle));
        sc.close();
    }

}
```

How to provide appropriate decimal value :

- In java.text package, There is a predefined class called DecimalFormat.
- This class provides the facility to format the decimal value according to user choice.

`DecimalFormat df = new DecimalFormat("00.00"); //String pattern ("00.00")`

- It contains a non static method called format which accept double as a parameter and return type is String.