Practical 02 - Encapsulation

We have already discussed a about encapsulation while discussing OOPs concepts.

The whole idea behind encapsulation is to hide the implementation details from users. If a data member is private it means it can only be accessed within the same class. No outside class can access private data member (variable) of other class. However if we setup public getter and setter methods to update (for e.g. void setSSN(int ssn))and read (for e.g. int getSSN()) the private data fields then the outside class can access those private data fields via public methods. This way data can only be accessed by public methods, thus making the private fields and their implementation hidden for outside classes. That's why encapsulation is known as data hiding.

```
public class EncapsulationDemo{
    private String empName;

    //Getter and Setter methods

    public String getEmpName(){
        return empName;
    }

    public void setEmpName(String newValue){
        empName = newValue;
    }

}
public class EncapsTest{
    public static void main(String args[]){
         EncapsulationDemo obj = new EncapsulationDemo();
         obj.setEmpName("Mario");
         System.out.println("Employee Name: " + obj.getEmpName());
    }
}
```

. Let's break down the code step by step:

- We have a class **'EncapsulationDemo'** with a private data member **'empName'**. By making **'empNam'e** private, we are hiding the implementation details of this variable from other classes.
- To access and modify the value of **'empName'**, we provide public getter and setter methods**: 'getEmpName()**' and '**setEmpName(String newValue**)'. These methods allow other classes to read and update the value of **'empName'** indirectly.

- In the '**EncapsTest**' class, we create an object **'obj'** of the **'EncapsulationDemo'** class.
- We use the **'setEmpName("Mario")**' method to set the value of '**empName**' to "Mario".
- Finally, we use the **'getEmpName()**' method to retrieve the value of **'empName'** and print it to the console using **'System.out.println()'**