What is encapsulation?

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 example void setSSN(int ssn))and read (for example 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. Lets see an example to understand this concept better.

Example of Encapsulation in Java

How to implement encapsulation in java:
1) Make the instance variables private so that they cannot be accessed directly from outside the class. You can only set and get values of these variables through the methods of the class.
2) Have getter and setter methods in the class to set and get the values of the fields.

```
class EncapsulationDemo{

    private int ssn;

    private String empName;

    private int empAge;

    //Getter and Setter methods

    public int getEmpSSN(){

        return ssn;

    }

    public String getEmpName(){

        return empName;

    }
    public int getEmpAge(){
        return empAge;
    }
    public void setEmpAge(int newValue){
        empAge = newValue;
    }
    public void setEmpName(String newValue){
        empName = newValue;
    }
    public void setEmpSSN(int newValue){
        ssn = newValue;
    }
}
public class EncapsTest{
    public static void main(String args[]){
         EncapsulationDemo obj = new EncapsulationDemo();
         obj.setEmpName("Mario");
         obj.setEmpAge(32);
         obj.setEmpSSN(112233);
         System.out.println("Employee Name: " + obj.getEmpName());
         System.out.println("Employee SSN: " + obj.getEmpSSN());
         System.out.println("Employee Age: " + obj.getEmpAge());
    }
```

```
}
```

Output:

```
Employee Name: Mario

Employee SSN: 112233

Employee Age: 32
```

In above example all the three data members (or data fields) are private(see: Access Modifiers in Java) which cannot be accessed directly. These fields can be accessed via public methods only. Fields empName, ssn and empAge are made hidden data fields using encapsulation technique of OOPs.

<u>Advantages of Encapsulation in Java</u>

Encapsulation is binding the data with its related functionalities. Here functionalities mean "methods" and data means "variables"

So we keep variable and methods in one place. That place is "class." Class is the base for encapsulation.

With Java Encapsulation, you can hide (restrict access) to critical data members in your code, which improves security

As we discussed earlier, if a data member is declared "private", then it can only be accessed within the same class. No outside class can access data member (variable) of other class.

However, if you need to access these variables, you have to use public "getter" and "setter"methods.