

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());
    }
}
```

Exercise 3-1: Develop a code for the following scenario.

“An encapsulated class contains three variables to store Name, Age and Salary of the employee. Evelop getters and setters to set and get values . Develop a test class to test your code.”

Now modify the same code by trying to replace the setters using a constructor.

```
class Employee {

    private String name;
    private int age;
    private double salary;

    public String getName() {
        return name;
    }
}
```

```

    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }
}

class TestEmployee {

    public static void main(String[] args) {
        Employee employee = new Employee();
        employee.setName("John Doe");
        employee.setAge(30);
        employee.setSalary(100000);

        System.out.println("Employee name: " + employee.getName());
        System.out.println("Employee age: " + employee.getAge());
        System.out.println("Employee salary: " + employee.getSalary());
    }
}

```

Output:

```

Employee name: John Doe
Employee age: 30
Employee salary: 100000

```

Exercise 3-2: Code for the last example that we have discussed during the class. We need the following Output. (Use Netbeans code generation option where necessary)

Employee Name: xxxxx (Use setter to set and getter to retrieve)

Basic Salary: xxxxx (Use setter to set and getter to retrieve)

Bonus: xxxxx (You may use the constructor to pass this value)

Bonus Amount: xxxxx (Develop a separate method to calculate Bonus amount. Bonus amount is the total of Bonus and Basic Salary)

E.g.

Employee Name: Bogdan

Basic Salary: 50000

Bonus: 10000

Bonus Amount: 60000

```
public class Employee {
    private String name;
    private int basicSalary;
    private int bonus;

    public Employee(String name, int bonus) {
        this.name = name;
        this.bonus = bonus;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setBasicSalary(int basicSalary) {
        this.basicSalary = basicSalary;
    }

    public int getBasicSalary() {
        return basicSalary;
    }

    public int getBonus() {
        return bonus;
    }
}
```

```
}

public int getBonusAmount() {
    return basicSalary + bonus;
}

public static void main(String[] args) {
    Employee employee = new Employee("Bogdan", 10000);
    employee.setBasicSalary(50000);

    System.out.println("Employee Name: " + employee.getName());
    System.out.println("Basic Salary: " + employee.getBasicSalary());
    System.out.println("Bonus: " + employee.getBonus());
    System.out.println("Bonus Amount: " + employee.getBonusAmount());
}
}
```

Output:

```
Employee Name: Bogdan
Basic Salary: 50000
Bonus: 10000
Bonus Amount: 60000
```