OBJECT ORIENTED PROGRAMMING (OOP)

An <u>Object Oriented Programming</u> is a modular approach, which allows the data to be applied within stipulated program area. It give more emphasis to data.

<u>PRINCIPLE OF OOPs</u>

1)Encapsulation

2) Inheritance

3)Polymorphism

4)Abstraction

ENCAPSULATION

ENCAPSULATION

❏ It is the process of binding properties and behaviours of the object together.

OR

❏ Hiding the internal states and requiring all the interactions to be performed only through non static method is known as Encapsulation.

❏ internal state-->value of variable
❏ interaction-->reading and modifying the value

❏ We can achieve encapsulation in java with the help of the class.
❏ Class has both state and behaviour of an Object.
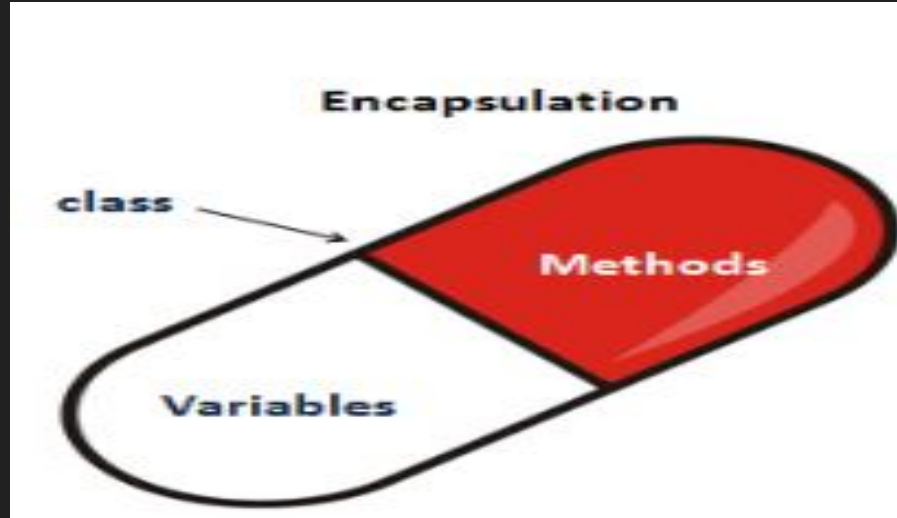
❏ **By using encapsulation we can achieve the "Data Hiding".**

DATA HIDING

❏ It is a process of restricting the direct access of data members of an object and provide indirect secured access of data members via methods of the same object is known as "Data Hiding".

❏ Data hiding helps to verify and validate the data before storing and modify it.

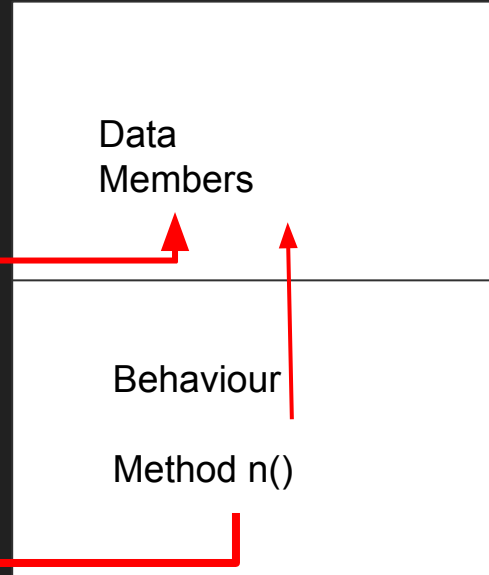❏ We can achieve data hiding with the help of private access specifier/modifier.

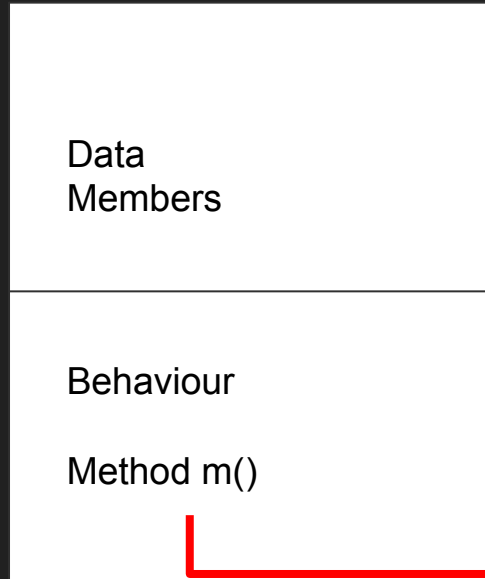# Example without Data hiding

class A

class B

Data
Members
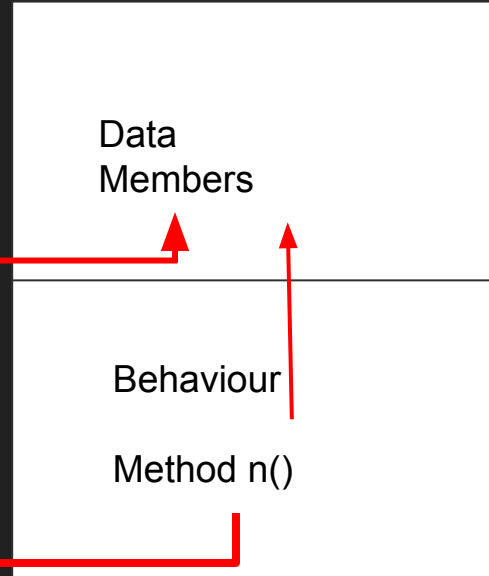
Direct access

Data
Members

Behaviour

Method m()

Behaviour
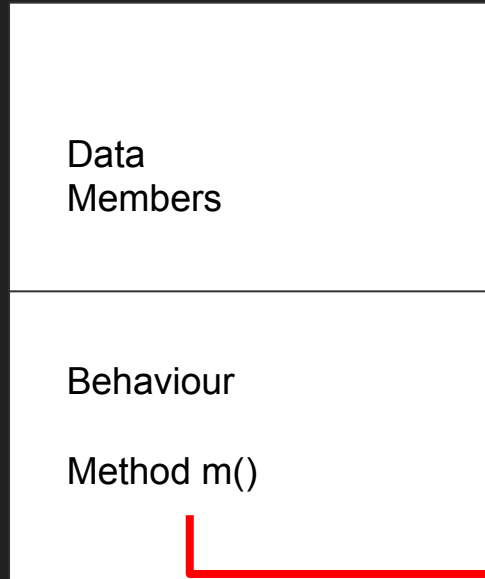
Method n()

Example with Data hiding
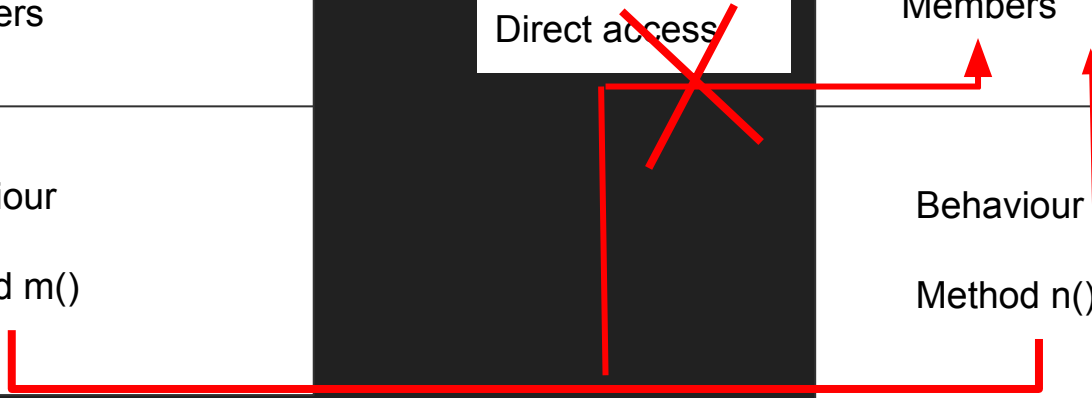
Steps to achieve Data Hiding

Step 1:- Prefix data members of a class with "Private" modifier.

Step 2:- Design a Getter and Setter method

Private Modifier

1. **Private is a keyword in java which acts as access specifier/modifier , when a member of a class is made private.**
2. **It can be used only inside the same class, we cannot use it outside the class.**

Getter & Setter method

In order to use the private variable of a class outside the class, we should make use of getter and setter methods.

Getter Method

❖ Getter method is used for reading the value of a variable.

❖ The return type of the getter method is the type of the hidden value.

Setter method

❖ Setter method is used for modifying the value of a variable.
❖ The return type of the Setter method is always void.

Note:-

The validation and verification can be done in this method before storing the data and before reading private data member.

Note:-

➔ **If the private variable should be both readable as well modifiable outside the class , we should design both getter and setter methods.**

➔ **if the private variable should be only readable outside the class , then we should design only getter method.**

➔ **if the private variable should be only modifiable outside the class ,then we should design only setter method.**

Syntax of getter method :-

```
    public  datatype_of_the_attribute   getName_of_Attribute()

    {

       return attribute;

    }
```

 Eg:-

```
    public String getName()

    {

     return name;

    }
```

<u>Syntax of setter method</u> :-

```
public void setName_of_Attribute(datatype_of_the_attribute attribute)

  {

    this.attribute=attribute;

  }
```

Eg:-
```
public void setName(String name)

  {

    this.name=name;

  }
```

Advantages of Encapsulation

★   The encapsulated code is more flexible and easy to change with new requirements.

★   It prevents the other classes to access the private field.

★   It allows modifying implemented code without breaking others code who have implemented the code.

★   It provides security to the data member.

★   It improves the maintainability of the application.