09 Feb'23 Encapsulation Assignment-16

1. What is Encapsulation in Java? Why is it called Data hiding?
Ans: Binding of data and corresponding methods into a single unit is called **"Encapsulation"**.
● If any java class follows data hiding and abstraction then such class is referred as **"Encapsulated class"**.

Encapsulation = Data Hiding + Data Abstraction.

● Our internal data should not go to the outside world directly, that is, outside
people should not access our internal data directly.

● By using private modifiers we can implement **"data hiding".**

Example:
```
class Account
{
private double balance;
}
```

Advantages of Data Hiding is security.
Recommended modifier for data members is private.

2. What are the important features of Encapsulation?
Ans: The following are the significant benefits of encapsulation.
A class can have complete control over its data members and data methods.
The class will maintain its data members and methods as read-only.
Data hiding prevents the user from the complex implementations in the code.

3. What are getter and setter methods in Java Explain with an example
Ans: Setter methods are used to set the value to the instance variables of the class.

09 Feb'23 Encapsulation Assignment-16

Syntax for setter method

a. compulsory the method name should start with set.
b. it should be public.
c. return type should be void.
d. compulsorily it should have some argument.

Getter methods are used to get the value from the instance variables of the class.

Syntax for getter method

a. compulsory the method name should start with get.
b. it should be public.
c. return type should not be void.
d. compulsorily it should not have any argument.

```
class Student
{
private String name;
private Integer id;
private String address;
//setters
public void setName(String name){
this.name = name;
}
public void setId(Integer id){
this.id = id;
}
public void setAddress(String address){
this.address = address;
}
//getters
public Integer getId(){
return id;
}
public String getName(){
```

09 Feb'23 Encapsulation Assignment-16

```
return name;
}
public String getAddress(){
return address;
}
}
public class Demo
{
public static void main(String[] args)
{
Student std = new Student();
std.setId(10);
std.setName("sachin");
std.setAddress("MI");
System.out.println("Id is :: "+std.getId());
System.out.println("Name is :: "+std.getName());
System.out.println("Address is :: "+std.getAddress());
}
}
```

Output:

```
Name is :: sachin
Id is :: 10
Address is :: MI
```

Note:

if the property is of type boolean then for the getter method we can prefix with either **"is/get".**

Example:

```
public class Student{
private boolean married;
public void setMarried(boolean married){
this.married=married;
}
public boolean isMarried()(){
```

return married;
}
}

4. What is the use of this keyword explain with an example
Ans: **‘this’** is a keyword used to represent the method belonging to that particular current object. Every class will have variables and methods. Variables can be local, static, or instance types from the superclass.

For example, variables are declared within the class but outside of the method in order to access them, which can be done via an object, and we use this keyword to do so.

Note that the **super keyword** is not the same as **“this.”** We use the **super** keyword to override methods with the same name in the **superclass.**

5. What is the advantage of Encapsulation?
Ans:
● We can achieve security.
● Enhancement becomes easy.
● Maintainability and modularisation becomes easy.
● It provides flexibility to the user to use the system very easily.

6. How to achieve encapsulation in Java? Give an example.
Ans: Java achieves encapsulation implementation by making all the data members private and then providing getter and setter methods that are public so that we can read the values of the private variables and set new values for these variables.

```
//Student_Id and name bundled in a unit "Student" =&gt; encapsulation
class Student {
    private int Student_Id;
    private String name;
    //getters, setters for Student_Id and name fields.
    public int getId() {
```

09 Feb'23 Encapsulation Assignment-16

```
        return Student_Id;
    }
    public void setId(int s_id) {
        this.Student_Id = s_id;
    }
    public String getname() {
        return name;
    }
    public void setname(String s_name) {
        this.name = s_name;
    }
}
class Main{
    public static void main(String[] args) {
        //create an object of Student class
        Student s=new Student();
        //set fields values using setter methods
        s.setId (27);
        s.setname("Tom Lee");
        //print values using getter methods
        System.out.println("Student Data:" + "\nStudent ID:" + s.getId()
                           + " Student Name:" + s.getname());
    }
}
```

output:

Student Data:
Student ID: 27 Student Name: Tom Lee