Q1. What is Encapsulation in java? Why is it called Data hiding?

Ans. **Encapsulation=Data Hiding + Abstraction.**

Binding of data and corresponding methods into a single unit called Encapsulation. In java class follows data hiding and abstraction then such classes are called encapsulated classes.**It is called data hiding** because no outsider can access the instance members of the class directly.They are only accessible directly to the methods of the class.

Q2. What are the important features of Encapsulation?

Ans.Important features of Encapsulation are:

->We can achieve the security of the data.

->No outsider can access the private members of the class.

->We achieve data hiding and abstraction through encapsulation.

Q3. What are the getters and setter methods in java? Explain with an example?

Ans.**GETTER:**These methods are used to get the value of the instance variable.

Syntax:

->These methods should start with the **get** name.

->They do not have an argument type.

->They have return type exactly the type of instance variable type.

->They must have public scope.

SETTER: These methods are used to set the value of the instance variable.

Syntax:

->These methods should start with the **set** name.

->They must have an argument type.

->They do not have return type i.e. **void**.

->They must have public scope.

```
class student
{
    private int age;//Instance variable
    private String name;//Instance variable

    //SETTERS
    public void setAge(int age)
    {
        this.age=age;
    }
    public void setName(String name)
    {
        this.name=name;
    }

    //GETTERS
```

```
    public int getAge()
    {
        return age;
    }
    public String getName()
    {
        return name;
    }

}
public class Setters_Getters {
    public static void main(String args[])
    {
        student sobject=new student();
        sobject.setAge(20);
        sobject.setName("MOHIT");
     int age=sobject.getAge();
     String name=sobject.getName();
     System.out.print("NAME: "+name);
     System.out.println("    Age: "+age);
    }
}
OUTPUT:
NAME: MOHIT       Age: 20
```

Q4. What is the use of this keyword? Explain with an example?
Ans. We use **this keyword** to distinguish between the local variable of methods and instance variable of class. We use **this keyword** when we have a similar name for both instance variables and local variables of methods in the class.
->It always points to the current object of the class.
->It is used to resolve the **shadowing problem.**

```
class student
{
    private int age;//Instance variable
    private String name;//Instance variable

    //SETTERS
    public void setAge(int age)
    {
```

```
        this.age=age;//use of this keyword
    }
    public void setName(String name)
    {
        this.name=name;
    }

    //GETTERS
    public int getAge()
    {
        return age;
    }
    public String getName()
    {
        return name;
    }

}
public class this_Keyword {
    public static void main(String args[])
    {
     System.out.println("Use of this keyword in the class");
    }
}
OUTPUT:
Use of this keyword in the class
```

Q5. What is the advantage of Encapsulation?

Ans. Advantage of Encapsulation are:

-> We achieve data hiding through this concept.

->We can achieve security.

->Enhancement became easy.

->Maintainability and modularisation becomes very easy.

Q6. How to achieve encapsulation in Java? Give an example?

Ans. By using a private modifier for the instance variable inside the class through which no outsider can access the instance variable directly.

```
class student
{
    private int age;//Instance variable
```

```
    private String name;//Instance variable

    //SETTERS
    public void setAge(int age)
    {
        this.age=age;
    }
    public void setName(String name)
    {
        this.name=name;
    }

    //GETTERS
    public int getAge()
    {
        return age;
    }
    public String getName()
    {
        return name;
    }

}
public class Setters_Getters {
    public static void main(String args[])
    {
        student sobject=new student();
        sobject.setAge(20);
        sobject.setName("MOHIT");

     System.out.print("Name: "+sobject.getName());
     System.out.println("    Age: "+sobject.getAge());
    }
}
OUTPUT:
Name: MOHIT     Age: 20
```