Encapsulation

1. **Define Encapsulation**

Dictionary Meaning: *"the action of enclosing something in or as if in a capsule."*
OOP Definition1: *"Encapsulation refers to the bundling of fields and methods inside a single class". It prevents outer classes from accessing and changing fields and methods of a class. This also helps to achieve data hiding.*
OOP Definition2: *"The process of binding data and corresponding methods (behavior) together into a single unit is called encapsulation in Java."*

```
Class
{
  Data Members(e.g. int a=10)
        +
  Methods(e.g. add() )
}
```

*The wrapping up of data and functions into a single unit is known as **encapsulation***

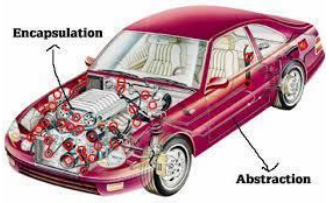
2. **Define Encapsulation with real world example**

Example 1: Capsule

A **Capsule** is wrapped or encapsulates various combinations of medicine. A Java Class is an example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

Example 2: Car

In a car engine, encapsulation refers to the concept of bundling the different components of the engine together and hiding their implementation details from the outside world. As a driver you know how to start the car by pressing the start button and internal details of the starting operations are hidden from you. So the entire starting process is hidden from you otherwise we can tell starting operation is encapsulated from you.



Example 3: Mobile Phone

You have a Mobile Phone, you can dial a number using keypad buttons. Even you don't know how these are working internally. This is achieved in JAVA using **Abstraction**. But how the Mobile Phone internally working? How keypad buttons are connected with internal circuit? This is achieved using **Encapsulation**.

3. **How to Achieve or Implement Encapsulation in Java**

There are two important points whereby we can achieve or implement encapsulation in Java program.

i. Declaring the **instance variable** of the class as **private** so that it cannot be accessed directly by anyone from outside the class.
ii. Provide the public **setter** and **getter** methods in the class to set/modify the values of the variable/fields.

4. **Advantage of Encapsulation in Java**

There are the following advantages of encapsulation in Java:

i. The encapsulated code is more **flexible** and easy to change with new requirements.
ii. It prevents the other classes from accessing the **private** fields.
iii. Encapsulation allows **modifying** the implemented code without breaking other code that has implemented the code.
iv. It keeps the data and codes **safe** from external **inheritance**. Thus, encapsulation helps to achieve **security**.
v. It improves the **maintainability** of the application.
vi. If you don't define the setter method in the class, then the fields can be made **read-only**.
vii. If you don't define the getter method in the class, then the fields can be made **write-only**.
viii. If you define both getter and setter methods in the class, then the fields can be made both read-write.

Encapsulation

5. What are the important features of Encapsulation?

Encapsulation means combining the data of our application and its manipulation in one place. It allows the state of an object to be accessed and modified through behaviour. It **reduces** the **coupling** of modules and increases the **cohesion** inside them.

6. What is data hiding in Java?

The insulation of the data from direct access by the program is called data hiding or information hiding. It is the process of enclosing one or more details from outside world through access right.

```
Encapsulation = Data Hiding + Abstraction
```

7. Advantages of Data Hiding.

i. Protects an object from unwanted access
ii. It reduces implementation errors.
iii. Simplifies the maintenance of the application and makes the application easy to understand.
iv. Protection of data from accidental corruption.

8. What do you mean by Tightly Encapsulated Class in Java

- If each variable is declared as private in the class, it is called tightly encapsulated class in Java. For a tightly encapsulated class, we are not required to check whether the class contains getter and setter method or not and whether these methods are declared as public or not.

Example

```
public class Bank{
 private double balance;
 private String accType;
 public double getbalance()
 {
   return balance;
 }
}
```

- If the parent class is not tightly encapsulated, no child class is tightly encapsulated because the parent class's non-private data members by default are available to every child class.
- Thus, we can say that data hiding, encapsulation, and tightly encapsulated class concepts are used for security purposes.

9. What is the difference between Abstraction and Encapsulation?

- Abstraction solves the problem at the design level whereas encapsulation solves the problem at the implementation level.
- Abstraction is implemented in Java using Interface and Abstract class whereas encapsulation is implemented using private and protected access modifiers.
- Abstraction is used to hide the unwanted data and giving relevant data whereas encapsulation is used for hiding data and code in a single unit to prevent access from outside.
- The real-time example of Abstraction is TV Remote Button whereas the real-time example of Encapsulation is medical medicine.

10. Can we achieve abstraction without encapsulation in Java?

Yes, we can achieve abstraction without encapsulation because both are different things and different concepts.

11. Differentiate data hiding and Encapsulation.

Data hiding	Encapsulation
Data hiding means protecting the variable of a class from outside world access.	Encapsulation means wrapping or binding data in a single unit.
Focuses on securing the data along with hiding.	Focuses more on wrapping data to make classes and the methods in the simplest form for the system.
It is a technique and process both.	It is a sub-process in data hiding.
Here, data is always private and inaccessible.	Here, data may be private or public.

Encapsulation

12. Where Encapsulation is used?

Encapsulation is used in various design patterns and real-life problems that make use of the Encapsulation object-oriented concept. It is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties direct access to them.

13. How does getter and setter functions help in encapsulation?

In encapsulation, the getters and setters method is because to hide the attributes of an object class from other classes so that no accidental modification of the data happens by methods in other classes.

Example:

Declare class variables/attributes as private.

Provide public get and set methods to access and update the value of a private variable.

```
public class Student {
  private String name;
  // Getter
  public String getName() {
    return name;
  }
  // Setter
  public void setName(String newName) {
    this.name = newName;
  }
}
```

14. Is Encapsulation violating reflection?

Encapsulation wants to give modules a safe space and Reflection wants to break into all code. Thus the manipulation of the data is done at one place unknowing to other members of variables. Encapsulation principles are broken using the reflection as it can access the private classes without setting getter and setter methods.

15. Differentiate Encapsulation and Polymorphism.

Packaging data and methods in a single unit are Encapsulation. Their own methods are used for accessing the data which are hidden in the objects.

The word polymorphism means Poly means many and morph means form.

Polymorphism means the ability to take more than one form by using a single interface. When single or many types are not defined by name it represents any type by using abstract symbols. Polymorphism is a process used for implementing inherited data.

16. What is a final class as it relates to encapsulation?

A final class as it relates to encapsulation is a class that a programmer cannot extend. No subclass within the code can inherit any variables or methods from a final class. It is advisable to write a final class when you want to create a class that is impossible to extend.

Example:

Consider you have a class that has sensitive data relating to a customer's payment and billing information. You can establish this class as a **final** class so that no subclasses can access it.

17. Which access modifier is used for encapsulation in Java?

The access modifier "***private***" is used for encapsulation in Java.

18. What do you understand by interfaces and abstract classes? How do they help with encapsulation?

Interfaces and abstract classes help with encapsulation by allowing you to specify the behaviour of a class without having to provide a concrete implementation. This means that you can hide the details of how a class works from the outside world, and only expose the interface that you want other classes to use. This makes it much easier to change the internals of a class without breaking any code that depends on it.

Encapsulation

19. Why to use private constructors?

Private constructors are used in order to prevent a class from being instantiated. This is often done in order to enforce the Singleton pattern, where only one instance of a class is allowed to exist.

20. What's your opinion on public static final members (constants) in a Java class?

Public static final members are a great way to create constants in a Java class. By making the members public, you are ensuring that they can be accessed by any other class that needs to use them. By making them static, you are ensuring that there is only one copy of the constant that is shared by all instances of the class. And by making them final, you are ensuring that the value of the constant cannot be changed.

21. What are some advantages of using constants instead of variables or hardcoded values?

i. Constants are immutable, meaning they cannot be changed once they are defined. This can be helpful in ensuring that your code always uses the same value for a given quantity, which can make your code more reliable.
ii. Additionally, constants can improve the readability of your code by making it clear what values are being used.
iii. Finally, constants can make your code more flexible, as they can be easily changed if the need arises.

22. Can you give me some examples of where the object state should be hidden from the outside world?

Example-1: you might want to hide an object's state is when you are working with sensitive data that you don't want to be tampered with.

E.g. Bank balance, Exam Result etc.

Example-2: Another example might be when you are working with an object that is in a delicate state and you don't want outside forces to be able to change it and potentially break it.

E.g. Military data.

23. What are some best practices for writing production-quality code that makes use of encapsulation?

i. Always make sure that your data is properly encapsulated and hidden from outside access.
ii. Be sure to write code that is easy to understand and follow, as this will make it easier for others to work with your code.
iii. Thoroughly test your code before putting it into production, as this will help to ensure that it is working as intended.

24. What is coupling?

Coupling is the degree of interdependence between software modules; a measure of how closely connected two modules are. When module A calls module B, they are said to be coupled. The more modules that module A calls, the higher its coupling.

25. What is cohesion?

Cohesion is a measure of how well the elements of a module work together to achieve a specific goal. A module with high cohesion is one where the elements are all focused on a single task, while a module with low cohesion is one where the elements are more general purpose and can be used for multiple tasks.