25 IMPORTANT

OOPS CONCEPT

INTERVIEW QUESTIONS

Q 1. What is object-oriented programming (OOPs)?

Ans: Object-Oriented Programming (OOP) is a programming paradigm that organizes code into objects, which are instances of classes.

It focuses on encapsulating data and behavior together, promoting modularity and reusability.

Q 2. What is encapsulation?

Ans: Encapsulation is the concept of bundling data (attributes) and methods (functions) that operate on that data into a single unit (class). It hides the internal details of an object and provides controlled access through methods.

Q 3. What is inheritance?

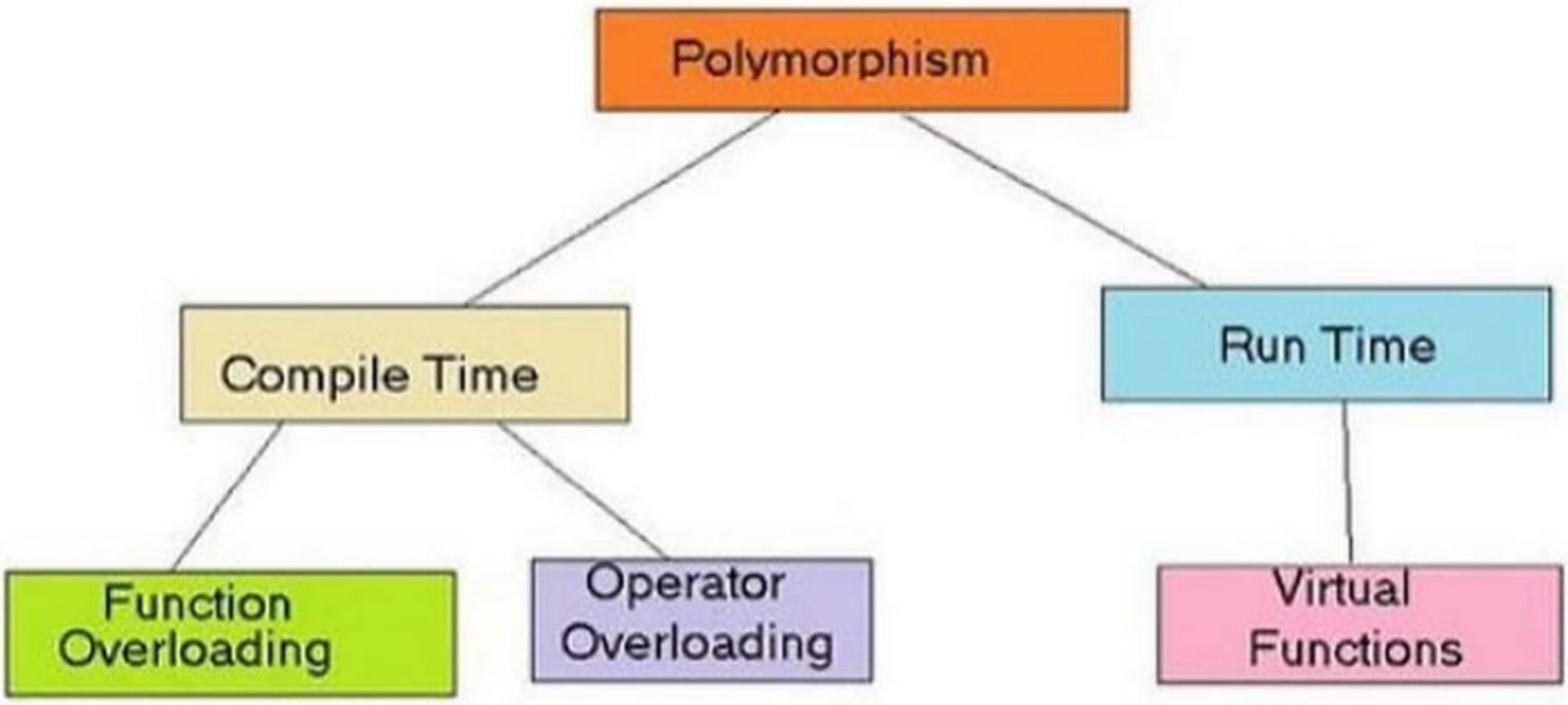
Ans: Inheritance is a mechanism where a new class (subclass or derived class) inherits properties and behaviors from an existing class (superclass or base class).

It promotes code reuse and supports hierarchical relationships.

Q 4. What is polymorphism?

Ans: Polymorphism allows objects of different classes to be treated as instances of a common superclass.

It enables methods to be invoked on objects without knowing their specific types, as long as they adhere to the common interface.



Q 5. What is a constructor?

Ans: A constructor is a special method that is automatically called when an object is created.

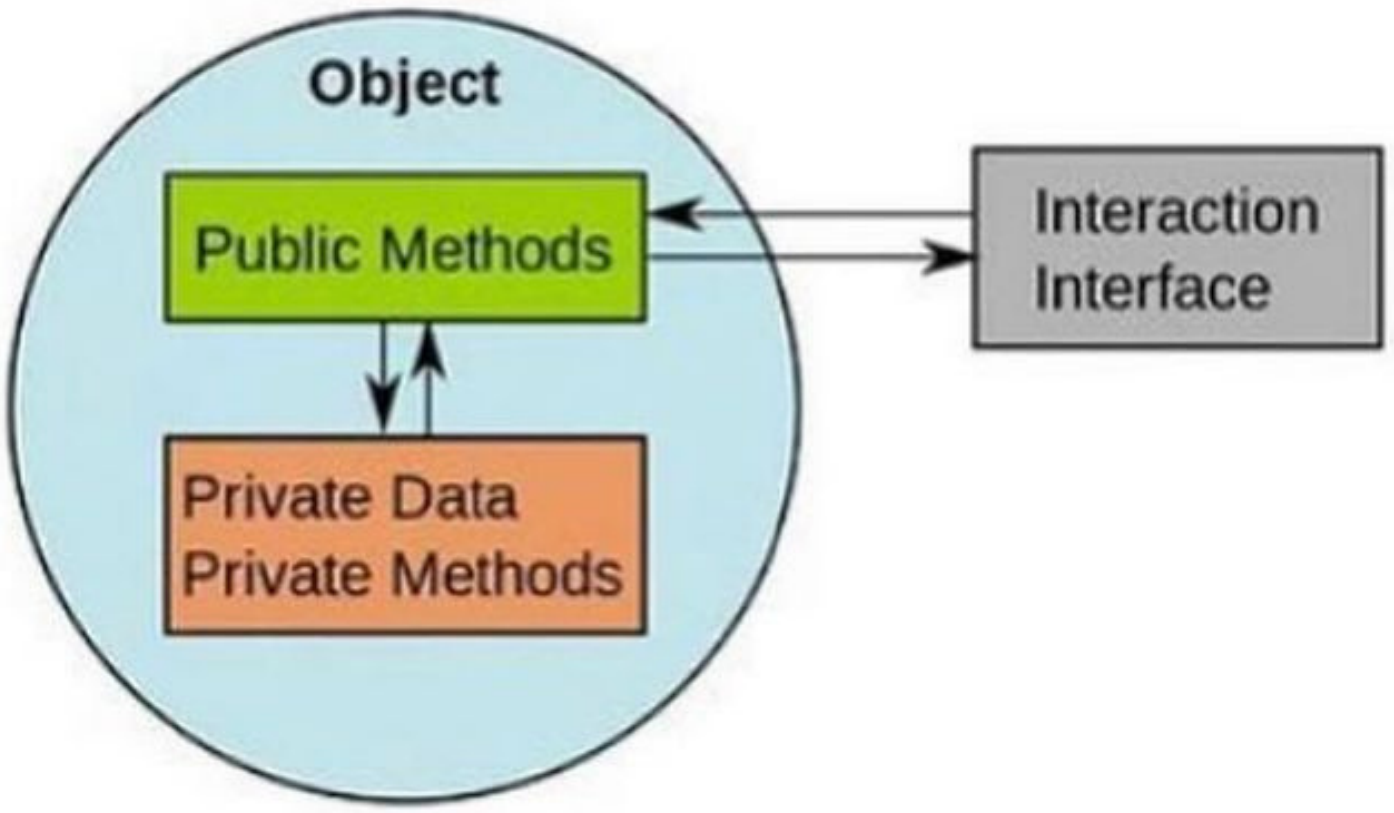
It initializes the object's attributes and prepares the object for use. Constructors often have the same name as the class.

Q 9. What is an interface?

Ans: An interface is a contract that defines a set of methods that a class must implement.

It allows multiple classes to adhere to the same interface, promoting a form of multiple inheritance.

Interfaces only declare method signatures, not implementations.



Q 10. What is a static method?

Ans: A static method belongs to the class itself, not to instances of the class.

It can be called using the class name and is often used for utility functions or operations that don't require instance-specific data.

Q 6. What is method overloading?

Ans: Method overloading is the ability to define multiple methods in a class with the same name but different parameter lists.

The methods are differentiated based on the number or types of parameters they accept.

Q 7. What is method overriding?

Ans: Method overriding is the process by which a subclass provides a specific implementation for a method that is already defined in its superclass.

The overridden method in the subclass has the same name, return type, and parameters.

Q 8. What is an abstract class?

Ans: An abstract class is a class that cannot be instantiated on its own.

It may contain abstract methods (methods without a body) that must be implemented by its concrete subclasses. Abstract classes provide a common interface.

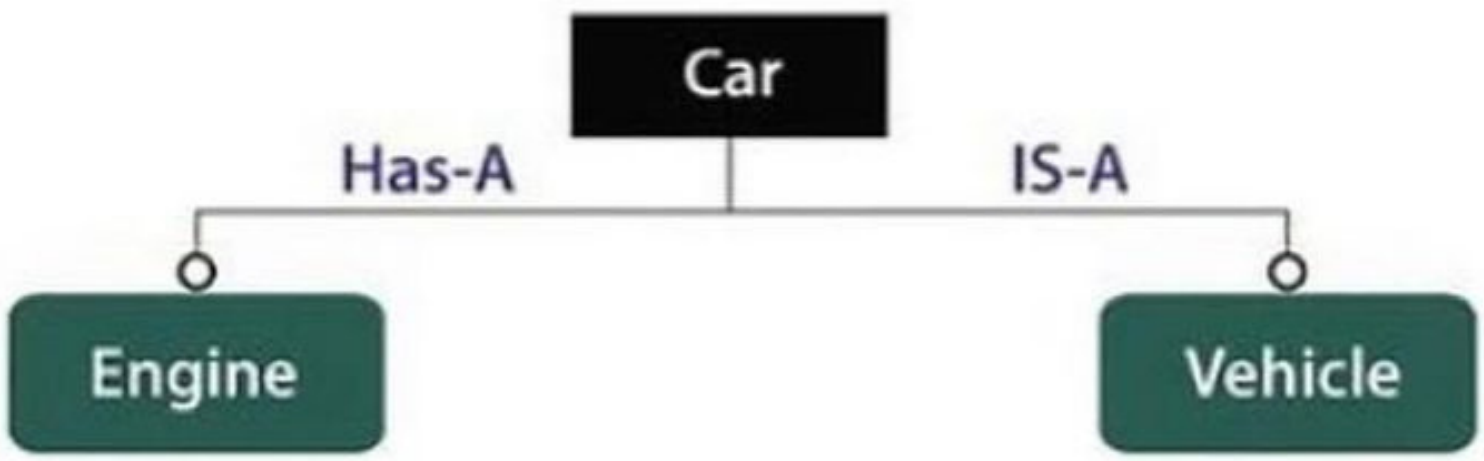
Q 11. What is a final class?

Ans: A final class is a class that cannot be subclassed. It prevents other classes from extending it and inheriting its behavior.

Q 12. What is composition?

Ans: Composition is a design principle where a class contains objects of other classes as part of its attributes.

It allows creating complex structures by combining simpler classes.



Q 13. What is a super keyword?

Ans: The **super keyword** is used to refer to the parent class or superclass. It can be used to call methods and constructors from the superclass.

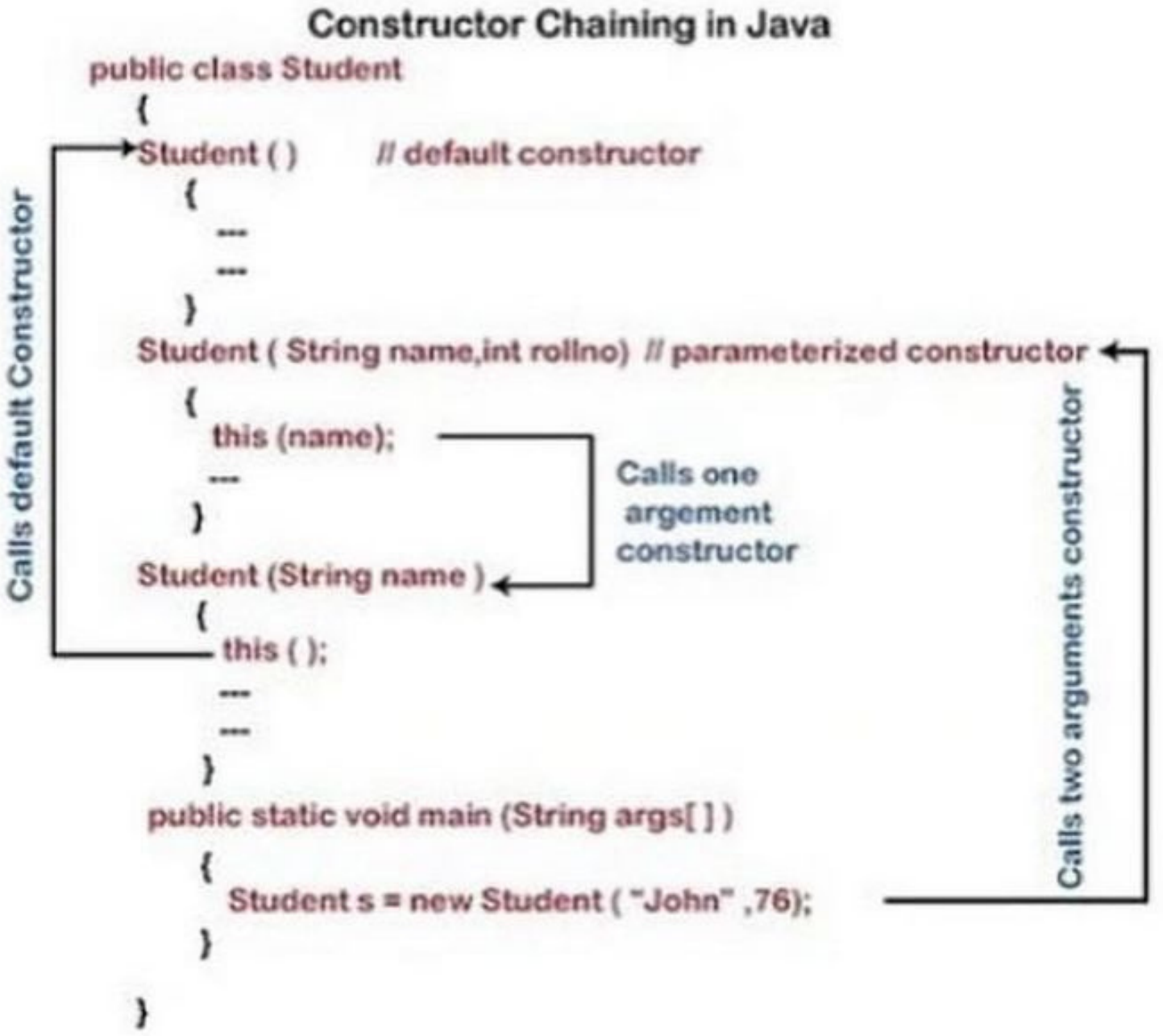
Q 14. What is method hiding?

Ans: Feature engineering involves transforming raw data into meaningful features to improve model performance.

Q 15. What is a constructor chaining?

Ans: Constructor chaining is the process of calling one constructor from another within the same class or between a superclass and a subclass.

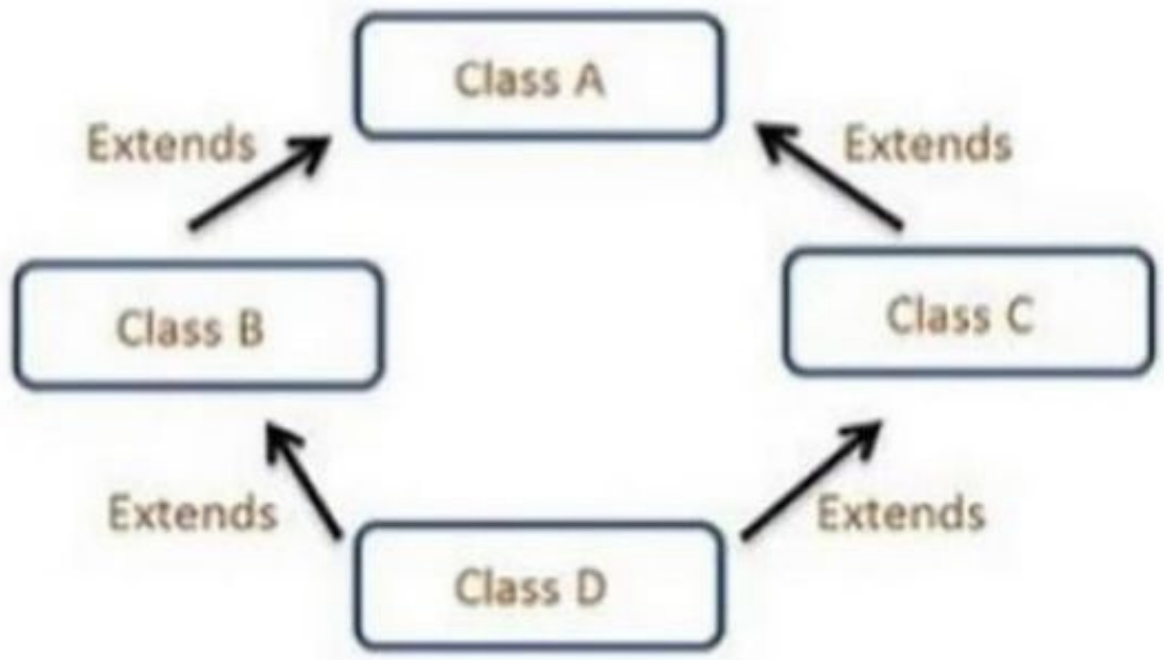
It allows for code reuse and efficient initialization.



Q 16. What is the diamond problem in multiple inheritance?

Ans: The diamond problem occurs in languages that support multiple inheritance, where a class inherits from two classes that have a common base class.

This can lead to ambiguity in method resolution. Some languages provide mechanisms to handle this, like virtual inheritance.



Q 17. What is a shallow copy and a deep copy?

Ans: A shallow copy copies the references of objects contained within an object, while a deep copy creates new instances of the objects contained.

A deep copy results in a completely independent copy of the original object and its contained objects.

Q 21. What is the SOLID principle?

Ans: The SOLID principle is an acronym for a set of design principles that promote maintainable and scalable code:

Q 22. What is the difference between an abstract class and an interface?

Ans: An abstract class can have both abstract and concrete methods, while an interface only has method signatures (no method implementations).

A class can implement multiple interfaces, but it can inherit from only one abstract class.