**B1. What is inheritance?**

Different kinds of objects often have a certain amount in common with each other. Mountain bikes, road bikes, and tandem bikes, for example, all share the characteristics of bicycles (current speed, current pedal cadence, current gear). Yet each also defines additional features that make them different: tandem bicycles have two seats and two sets of handlebars; road bikes have drop handlebars; some mountain bikes have an additional chain ring, giving them a lower gear ratio.

Object-oriented programming allows classes to *inherit* commonly used state and behavior from other classes. In this example, Bicycle now becomes the *superclass* of MountainBike, RoadBike, and TandemBike. In the Java programming language, each class is allowed to have one direct superclass, and each superclass has the potential for an unlimited number of *subclasses*:

**B2. Which inheritance is not supported by Java? Why?**

A class can implement any number of interfaces but can extend only one class. Multiple inheritance is not supported because it leads to deadly diamond problem. However, it can be solved but it leads to complex system so multiple inheritance has been dropped by Java founders

**B3. What is advantage of inheritance?**

One of the key benefits of inheritance is to minimize the amount of duplicate code in an application by sharing common code amongst several subclasses, where equivalent code exists in two related classes. This also tends to result in a better organization of code and smaller, simpler compilation units

B4. **Difference between inheritance and encapsulation.**

Inheritance is an object oriented concept which creates a parent-child relationship. It is one of the ways to reuse the code written for parent class but it also forms the basis of Polymorphism.

Encapsulation is an object oriented concept which is used to hide the internal details of a class, for example, HashMap encapsulate how it store elements and calculate hash values.

**B5. Difference between inheritance and abstraction.**

Interface can't provide the implementation of abstract class. Inheritance vs Abstraction: A Java interface can be implemented using keyword “implements” and abstract class can be extended using keyword “extends”. ... A Java abstract class can have class members like private, protected, etc.

**B6. Difference between inheritance and polymorphism.**

**Inheritance** is creating a class that derives its feature from an already existing class. On the other hand, **polymorphism** is an interface that can be defined in multiple forms. **Inheritance** is implemented on the classes whereas, the **polymorphism** is implemented on methods/functions

**B7. Can we override static method in Java?**

Answer is, No, you can not override static method in Java, though you can declare method with same signature in sub class. ... As per Java coding convention, static methods should be accessed by class name rather than object. In short Static method can be overloaded, but can not be overridden in Java

**B8. Can we overload static method in Java?**

Answer is, No, you can not override static method in Java, though you can declare method with same signature in sub class. ... As per Java coding convention, static methods should be accessed by class name rather than object. In short Static method can be overloaded, but can not be overridden in Java.

**B9. Can a class implement more than one interface?**

A Java class can only extend one parent class. Multiple inheritance ( extends ) is not allowed. Interfaces are not classes, however, and a class can implement more than one interface. The parent interfaces are declared in a comma-separated list, after the implements keyword

**B10. Can a class extend more than one class in Java?**

Java does not support multiple inheritance, that's why you can't extend a class from two different classes at the same time. Rather, use a single class to extend from, and use interfaces to include additional functionality

**B11. Can an interface extend more than one interface in Java?**

A Java class can only extend one parent class. Multiple inheritance is not allowed. Interfaces are not classes, however, and an interface can extend more than one parent interface. The extends keyword is used once, and the parent interfaces are declared in a comma-separated list.

**B12. What will happen if a class implements two interfaces and they both have a method with same name and signature?**

No, its an error

If two interfaces contain a method with the same signature but different return types, then it is impossible to implement both the interface simultaneously. According to JLS (§8.4.2) methods with same signature is not allowed in this case.

**B13. Can we pass an object of a subclass to a method expecting an object of the super class?**

You can call a method of the subclass by first casting the object hold by reference variable of superclass into the subclass. Once you hold the object in subclass reference type, you can call methods from the subclass.

**B14. Are static members inherited to sub classes?**

Static members are not inherited in java as they are the properties of class and they are loaded in class area. they had nothing to do with object creation. however only the child classes can access the static members of their parent classes.

**B15. What happens if the parent and the child class have a field with same identifier?**

Super class field member will be hidden at the sub class and super class field could be accessed using super keyword.

**B16. Are constructors and initializers also inherited to sub classes?**

No, Constructors and initializers (Static initializers and instance initializers) are not inherited to sub classes. But, they are executed while instantiating a sub class.

**B17. How do you restrict a member of a class from inheriting by its sub classes?**

By declaring that member as a private. Because, private members are not inherited to sub classes.

**B18. How do you implement multiple inheritance in java?**

Using interface, we can implement multiple inheritance in java. Classes in java can not extend more than one classes, but a class can implement more than one interfaces.

**B19. Can a class extend by itself in Java?**

Inheritance is implemented in JAVA using below two keywords,

* extends.
* implements.

extends inherits between two classes and two interfaces.

**B20. How do you override a private method in java?**

Private methods cannot be overridden as it is not visible outside of the class.

**B21. When to overload a method in Java and when to override it?**

If a class has a better implementation or most appropriate method implementation then override the method while overloading is performing the same functionality however with different input datatypes/object.

**B22. What is the order of extends and implements keyword on Java class declaration?**

The extends always precedes the implements keyword in any Java class declaration.

When the Java compiler compiles a class into bytecode, it must first look to a parent class because the underlying implementation of classes is to point to the bytecode of the parent class - which holds the relevant methods and fields.

**B23. How do you prevent overriding a Java method without using the final modifier?**

Declare that method using private or static keyword to prevent overriding.

**B24. What are the rules of method overriding in Java?**

private, final and static methods cannot be overridden.

The overriding method must have same argument list.

The overriding method must have same return type or covariant return type.

The overriding method cannot reduce the method visibility.

The overriding method must not throw new or broader checked exceptions.

Overriding method can increase access of overridden method.

**B25. Difference between method overriding and overloading in Java.**

Method overloading deals with the notion of having two or more methods in the same class with the same name but different arguments. Method overriding means having two methods with the same arguments, but different implementations. One of them would exist in the parent class, while another will be in the derived, or child class.

**B26. What happens when a class implements two interfaces and both declare field (variable) with same name?**

The field becomes ambiguous and we get compile time error.

**B27. Can a subclass instance method override a superclass static method?**

No. It results in compilation error at the subclass.

**B28. Can a subclass static method hide superclass instance method?**

No. It results in compilation error in the subclass.

**B29. Can a superclass access subclass member?**

No. Parent class cannot access the child class members. Super class reference variable cannot see subclass object members.

**B30. Difference between object oriented and object based language.**

Object oriented programming languages follow all concepts belonging to OOP. Object-based language doesn't support all the features of OOPs like Polymorphism and Inheritance Object-based language has in-built object like JavaScript has window object. Object-based languages are JavaScript, VB etc

**B31. Explain Diamond problem.**

The “diamond problem” is an ambiguity that can arise as a consequence of allowing multiple inheritance. It is a serious problem for languages (like C++) that allow for multiple inheritance of state. In Java, however, multiple inheritance is not allowed for classes, only for interfaces, and these do not contain state.

**B32. Why Java does not support operator overloading?**

Java doesn't "need" operator overloading because it's just a choice made by its creators who wanted to keep the language more simple. Unlike C++ , Java doesn't support operator overloading. ... The Java designers wanted to prevent people from using operators in a confusing manner, but it was not worth it.

**B33. What is Encapsulation in Java?**

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. ... Declare the variables of a class as private. Provide public setter and getter methods to modify and view the variables values.

**B34. Which of the Java OOPS feature promotes access protection or data hiding?**

**Encapsulation** promotes access protection and data hiding.