B1. What is inheritance?

- inheritance is defined as the process of deriving the properties and characteristics of another class. Inheritance provides reusability. With the help of inheritance we can reduce code.

Syntax

Class childClass extends parentClass{

//body

}

B2. Which inheritance is not supported by Dart? Why?

Dart doesn't support multiple inheritance because it creates complexity in the program. In hierarchical inheritance, two or more classes inherit a single class. In the following example, the two-child classes Peter and James inherit the Person class.

B3. What is the advantage of inheritance?

Benefits of Inheritance Inheritance helps in code reuse. The child class may use the code defined in the parent class without re-writing it. Inheritance can save time and effort as the main code need not be written again.

B4. Difference between inheritance and encapsulation.

Inheritance dictates that a child class (subclass) inherits all the attributes and methods from a parent class (superclass).

Encapsulation dictates that one class must not have access to the (private) data of another class. The way these rules are written, it seems that they are mutually exclusive.

B5. Difference between inheritance and abstraction.

The main difference between abstraction and inheritance is that abstraction allows hiding the internal details and displaying only the functionality to the users, while inheritance allows using properties and methods of an already existing class. Object-Oriented Programming (OOP) is a major programming paradigm.

B6. Difference between inheritance and polymorphism.

Inheritance is one in which a new class is created that inherits the properties of the already existing class. It supports the concept of code reusability and reduces the length of the code in object-oriented programming. Polymorphism is that in which we can perform a task in multiple forms or ways. It is applied to functions or methods.

B7. Can we override static methods in Dart?

we cannot override static methods. The calling of the method depends upon the type of object that calls the static method. It means: If we call a static method by using the parent class object, the original static method will be called from the parent class.

B8. Can we overload static methods in Dart?

Function overloading is not supported in Dart at all. Function overloading requires static types. Dart at its core is a dynamically typed language. You can either use different names for the methods or optional named or unnamed parameters.

B9. Can a class implement more than one interface?

Classes can implement any number of interfaces. When you design your class, you can choose not to implement any interfaces, you can implement a single interface, or you can implement two or more interfaces. For example, in addition to IStorable, you might have a second interface, ICompressible, for files that can be compressed to save disk space.

B10. Can a class extend more than one class in Dart?

We cannot extend more than one class in Dart. If you try to extend more than one class then Dart will throw the error “ Each class definition can have at most one extended clause”. This is one of the core concepts of Object Oriented Programming in any programming language and Dart is also the same here.

B11. Can an interface extend more than one interface in Dart?

We cannot extend more than one class in Dart. If you try to extend more than one class then Dart will throw the error “ Each class definition can have at most one extended clause”. This is one of the core concepts of Object Oriented Programming in any programming language and Dart is also the same here. Implements:

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

This might lead to a problem: what happens if a class implements two or more interfaces that have default methods with identical method names and method signatures? The simple answer is that the implementing class must override the method. However, this must be done explicitly.

B13. Can we pass an object of a subclass to a method expecting an object of the superclass?

Yes, you can pass that because subclass and superclass are related to each other by Inheritance which provides IS-A property.

B14. Are static members inherited to subclasses?

No. Static members are not inherited. Super and subclasses, on the other hand, can both have a static method with the same signature. At the subclass, the static member of the super class will be hidden.

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

Yes, it is possible to have the same data member in Parent and Child classes. Now, we will see the capability or strength of the Parent and Child class. The Parent class reference can hold its own object and Child class object as well and The Child class reference holds its own object only.

B16. Are constructors and initializers also inherited to subclasses?

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

B17. How do you restrict a member of a class from inheriting by its subclasses?

We can restrict a member of a class from inheriting to its subclasses by declaring the member as a private. Because, private members are not inherited to subclasses.

B18. How do you implement multiple inheritance in Dart?

Dart doesn't support multiple inheritance because it creates complexity in the program. Hierarchical Inheritance In the hierarchical inheritance, two or more classes inherit a single class. In the following example, the two-child classes Peter and James inherit the Person class.

B19. Can a class extend by itself in Dart?

You can inherit from or extend a class using the extends keyword. This allows you to share properties and methods between classes that are similar, but not exactly the same. Also, it allows different subtypes to share a common runtime type so that static analysis doesn't fail.

B20. How do you override a private method in Dart?

You can't override a private method, but you can introduce one in a derived class without a problem. This compiles fine:

class Base {

private void foo () {

} }

class Child extends Base { private void foo () {

} }

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

Method overriding occurs in dart when a child class tries to override the parent class’s method. When a child class extends a parent class, it gets full access to the methods of the parent class and thus it overrides the methods of the parent class. It is achieved by re-defining the same method present in the parent class.

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

Extends -In Dart, an extended keyword is often used to change class behavior using Inheritance. The ability of a class to acquire properties and features in another category is called Inheritance. It is the ability of a program to create a new class from an existing class. In simple terms, we can say that we use extends to form a subclass, and super refers to the superclass. A class whose properties the child class inherits is called a Parent Class. The parent class is also known as the base class or super class.

implements-An interface in Dart refers to the syntax or blueprint that any class must adhere to. It basically defines the array of methods available on the object. It provides the user with the blueprint of the class that any class should follow if it interfaces with that class. If a class inherits another class, it should override ( re-define ) the functions present inside that interfaced class in its own way as per the subject. In Dart, there isn't a specific or direct way of creating the interfaces. To implement them, we use the ' implement ' keyword. By default, every class is an interface in itself containing all the interface members and the members of any other interfaces that it implements.

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

Method overriding occurs in dart when a child class tries to override the parent class’s method. When a child class extends a parent class, it gets full access to the methods of the parent class and thus it overrides the methods of the parent class. It is achieved by re-defining the same method present in the parent class.

This method is helpful when you have to perform different functions for a different child class, so we can simply re-define the content by overriding it.

B24. What are the rules of method overriding in Dart?

The overriding method (the child class method) must be declared with the same configuration as the overridden method (the superclass method). The return type, list of arguments and its sequence must be the same as the parent class method. The overriding method must be defined in the subclass, not in the same class.

B25. Difference between method overriding and overloading in Dart.

Method overloading allows multiple methods in the same class to have the same name but different parameters.

Method overriding allows a parent class and a child class to have methods with the same name and same parameters. The child class method will override the parent class method.

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

If a type implements two interfaces, and each interface defines a method that has identical signature, then in effect there is only one method, and they are not distinguishable. If, say, the two methods have conflicting return types, then it will be a compilation error.

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

No, we cannot override static methods in Java. We can declare static methods with the same signature in subclass but it is not considered overriding as there would not be any run-time polymorphism. Overriding is a feature of OOP languages like Java that is related to run-time polymorphism.

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

Static Methods If a subclass defines a static method with the same signature as a static method in the superclass, then the method in the subclass hides the one in the superclass. The distinction between hiding a static method and overriding an instance method has important implications:

B29. Can a superclass access subclass member?

No, a superclass has no knowledge of its subclasses. Yes, a subclass has access to all non private members of its superclass.

B30. Difference between object oriented and object based language.

Object-oriented languages follow all the concepts of OOPs whereas the object-based language doesn’t follow all the concepts of OOPs like inheritance and polymorphism. Object-oriented languages do not have the inbuilt objects whereas Object-based languages have the inbuilt objects, for example, JavaScript has window objects.

B31. Explain Diamond problem.

The Diamond Problem is an ambiguity that arises in multiple inheritance when two parent classes inherit from the same grandparent class, and both parent classes are inherited by a single child class. Without using virtual inheritance, the child class would inherit the properties of the grandparent class twice, leading to ambiguity.

B32. Why does Dart not support operator overloading?

Dart did not support overloading originally because it was a much more dynamic language where the declared types did not have any semantic effect. That made it impossible to use static type based overload resolution.

B33. What is Encapsulation in Dart?

Encapsulation In Dart Encapsulation is the mechanism that confirms critical and sensitive data which is hidden from users. To achieve encapsulation, you can make fields private and use the public getter and setter method to access and set the value of that field.

B34. Which of the Dart OOPS features promotes access protection or data hiding?

Class , object, Data Encapsulation , Inheritance , Polymorphism ,