Q.1 What is inheritance?

- ➢ Inheriting is the process of deriving properties and characteristics from another class. It is a key concept in object-oriented programming.
- ➢ Ther are four main types of inheritance in darts: single inheritance, multiple inheritance, multilevel inheritance, hierarchical inheritance.

Q.2 Which inheritance is not supported by Dart? Why?

B3. What is advantage of inheritance?

- ➢ In dart multiple inheritance is not supported. Dart user single inheritance, meaning a class can only inherit from one superclass. This design choice simplifies the language and reduces complexity, making it easier to understand and maintain code. It also helps avoid issues like the diamond problem, which can arise in languages that support multiple inheritance.
- ➢ advantage of inheritance: Advantages of inheritance in dart it promotes reusability of the code and reduces redundant code. It helps to design a program in a better way

Q.3 Difference between inheritance and encapsulation.

B5. Difference between inheritance and abstraction.

- ➢ 1.Inheritance and encapsulation.
- ★ **Inheritance**: supports hierarchical relationship between classes and facilitates code reuse. A child class inherits all the features of its parent class, and method from the parent can be overridden in the child.

- ★ **<u>Encapsulation</u>**: Allows for controlled access to an object's data and behavior and hides implementation details. Encapsulation is possible without inheritance.
- ➢ 2. inheritance and abstraction.
- ★ **<u>Inheritance</u>**: Allow the use of properties and methods from an existing class. It also promotes a hierarchical structure and facilitates code reusability.
- ★ **<u>Abstraction</u>**: Hides internal details and only displays functionality to users. It also simplifies complex systems and enforces a high-level design.

Q.4 Difference between inheritance and polymorphism.

- ➢ **<u>Inheritance:</u>** inheritance is a concept of object-oriented programming. It allows a new class to inherit properties and behaviors from an existing class. This creates a hierarchical relationship between classes.
- ➢ **<u>Polymorphism:</u>** polymorphism is an interface that can take different shapes and forms. It allows objects of different classes to be treated as objects of a common superclass. It enables a single interface to represent different data types or classes.

Q.5 Can we override static method in Dart?

- ➢ The override static methods two static methods you declared there are in fact two different static methods, not the same, overridden one. Answer for a different question, but related: Dart doesn't inherit static methods to derived classes.

Q.6 Can we overload static method in Dart?

- No, it is not possible to override static methods in dart. This means that no one can change what is printed when a static method is constructed.

Q.7 Can a class implement more than one interface? B10. Can a class extend more than one class in Dart?

★ **Can a class implement more than one interface**

- In dart, an interface is defined using the abstract keyword. A class can implement multiple interfaces, and it must implement all the methods and properties defined in each interface.

★ **Can a class extend more than one class in Dart?**

- In dart, multi-level inheritance occurs when various classes inherit in a chain (i.e., one class extends a pare).A subclass is inherited by another subclass or forms an inheritance chain.). A subclass is inherited by another sub

Q.8 Can an interface extend more than one interface in Dart?

- Implementing multiple interfaces it doesn’t support multiple inheritance, in which a class can extend multiple classes. However. A class can extend a single class and implements multiple interfaces.

Q.9 What will happen if a class implements two interfaces and they both have a method with same name and signature?

- ➢ If a class implements two interfaces and each interface has a method with the same signature and return type, then the class can only implement one method. The method is not distinguishable from the other method, and it doesn’t matter which interface the method belongs to.

Q.10 Can we pass an object of a subclass to a method expecting an object of the super class? B14. Are static members inherited to sub classes?

- ★ **Can we pass an object of a subclass to a method expecting an object of the super class?**
- ➢ The subclass's method should accept every object that the superclass's method takes.
- ➢ To fix common type problems, you can widen the method's parameter types. You can also use the covariant keyword if you have a valid reason to use a subtype.
- ★ **Are static members inherited to sub classes?**
- ➢ Yes, there’s no inheritance of static members. See static methods section of the language specification: inheritance of statics Methode has litter utility in dart. Static methods cannot be overridden.

Q11. What happens if the parent and the child class have a field with same identifier? B16. Are constructors and initializers also inherited to sub classes?

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

- ➢ In dart, the super keyword can be used to access data members of the parent class if both parent and child have members with the same name. The super keyword can also be used to:
- – Call properties and method of the superclass
- – Call the parameterized constructor of the parent class.
- – Prevent overriding the parent method.
- ★ **Are constructors and initializers also inherited to sub classes?**
- ➢ Constructors aren't inherited Subclasses don't inherit constructors from their superclass. A subclass that declares no constructors has only the default (no argument, no name) constructor.

Q.12. How do you restrict a member of a class from inheriting by its sub classes?

- ➢ In Dart, hierarchical inheritance is when two classes inherit from a single class.
- ➢ To restrict the use of inheritance, you can use sealed classes. When a class is designated as sealed, it can only be subclassed within the package where it is declared.

Q.13. How do you implement multiple inheritance in Dart?

- ➢ Dart does not support multiple inheritance. But you can use mixins to achieve a similar effect. For example, you could create a car class that inherits from a vehicle class and a loggable mixin. This would allow the car class to reuse the code from both the vehicle class and the loggable mixin.

Q.14. Can a class extend by itself in Dart?

- Dart allows single inheritance, which means a class can extend from a single parent. But a single parent can have multiple children, and each child can have its own children, building up a hierarchy many classes deep.

Q.15. How do you override a private method in Dart?

- Copy the package folder in your current code and change it as per your need.
- Create a public method in library class and pass that private method in it. You can have the access of that private method as defined public method.

Q.16. When to overload a method in Dart and when to override it?

- Method overloading is when two or more methods in the same class have the same name but different parameters. Method overriding is when two methods have the same name and parameters.

Q.17 How do you prevent overriding a Dart method without using the final modifier?

- In short, apart from final modifier, you can also use static and private modifier to prevent a method from being overridden.

Q.18 What the order is of extends and implements keyword on Dart class declaration?

- **Extends**: Creates a subclass. For example, the class car could extend the class vehicle. In Dart, a class can only extend one class.
- **Implements**: Implements interfaces. Every class implicitly defines an interface. You can also use the keyword with.

Q.19. What are the rules of method overriding in Dart?

- In Dart, method overriding is when a child class redefines a method from its parent class. Here are some rules for method overriding in Dart:
- The overriding method must be defined in the subclass, not in the same class.
- The return type, list of arguments, and sequence must be the same as the parent class method.
- The method must be an exact copy of the parent class's method, from name to argument list.
- The parameter types must be the same type as or a supertype of the overridden method's parameter types.
- The overriding function can make a parameter's data type wider and a return type narrower.

Q.20. Difference between method overriding and overloading in Dart

- Here are some more details about method overriding and overloading:

- ★ **Method overriding**: The method signature must be the same. The method name, parameters, and return type must all be the same.
- ★ **Method overloading**: The method signature must be different. The method name must be the same, but the parameters can be different. The return type can be the same or different.

Q.21. What happens when a class implements two interfaces and both declare field (variable) with same name?

- ➢ If a type implements two interfaces, and each interface define 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.

Q.22. Can a subclass instance method override a superclass static method?

- ➢ An instance method cannot override a static method, and a static method cannot hide an instance method.

Q.23. Can a subclass static method hide superclass instance method?

- ➢ In Dart, static methods cannot be overridden. This means that no one can change what is printed when a static method is constructed.

- ➢ Static methods are associated with the class itself, not with the instance of a class. Therefore, when a subclass inherits a static method from its parent class, it cannot modify the behavior of the static method in any way.

Q.24. Can a superclass access subclass member?

- ➢ A superclass cannot access specific members of a subclass. A superclass has no knowledge of its subclasses, including which classes have inherited it. A superclass also cannot directly access variables or methods that are declared in a subclass.

Q.25. Difference between object oriented and object based language.

- ➢ Here are some differences between object-oriented and object-based languages
- ★ **<u>Object-oriented languages</u>**: Support all features of object-oriented programming (OOP). They don't have an in-built object.
- ★ **<u>Object-based languages</u>**: Don't support all features of OOPs, such as polymorphism and inheritance, and have an in-built object.