- **The difference between *extends*, *with*, and *implement***

In Dart, the *extends* keyword is used to inherit a class by deriving the properties and characteristics of the extended class. The class whose properties are inherited by another class is called the *Parent* class and is known as the super class, whereas the class that inherits the super class is called the *child* class.

For example:

```
class parent {
    void function(); }
class child extends parent {
    @override
    void function(){
        print(‘Hi’); }
}
```

Interfaces are defined as a set of methods available to the object. Dart doesn’t declare interfaces, but classes can be declared and treated as interfaces by using the *implements* keyword. A class can implement one or more interfaces, unlike inheritance which allows inheriting one class only. When a class implements another class (interface), the class must override all methods and attributes, otherwise, the compiler will detect an error.

For example:

```
class interface {
    int? num;
    void function(){
        print('Hi'); }
}
class child implements interface {
    @override
    int? num;
    @override
    void function(){
        print('Dart is cool!'); }
}
```

Mixins are a way of reusing the class's methods in multiple class hierarchies. Mixins can be explained as abstract classes used for reusing the methods in multiple classes which have the same attributes or methods. In other words, Mixins are a way to abstract and reuse a family of operations. It is similar to inheritance, but you can use multiple classes as one super class. We use *with* keyword for Mixins.

For example:

```
mixin temp {
void function(){
    print('hi'); }
}
mixin temp2 {
    void number(){
        print(10);
}
class test with temp, temp2 {
    @override
    void function(){
        print('Dart is cool'); }
    }
    }
```

- **The difference between regular classes and abstract classes**

In Dart programming language, abstract classes are classes that contain one or more abstract methods, which are methods with headings only and without bodies. To declare an abstract class, we must use the *abstract* keyword before the *class* keyword. For example, **abstract class University {}**. We could omit the abstract keyword but declare abstract methods in the class so that the compiler could recognize that it is an abstract class. In addition, abstract classes cannot be initialized, unlike regular classes. When an abstract class is extended, you must make sure that all abstract methods are overridden with implementation in the extender class. Otherwise, there would appear an error and you cannot run your class.

On the other hand, regular classes are a blueprint for creating objects and they encapsulate data for the objects. To declare a class in Dart, we use the *class* keyword and then the class's name. We can initialize objects from classes, add attributes, and create methods with implementation. Regular classes can extend abstract classes using the *extends* keyword. For instance, **class Student extends University {}**.

- **References**

1- https://www.geeksforgeeks.org/dart-extends-vs-with-vs-implements/#:~:text=In%20Dart%2C%20the%20extends%20keyword,class%20from%20an%20existing%20class.

2- https://www.geeksforgeeks.org/abstract-classes-in-dart/

3- https://www.tutorialspoint.com/dart_programming/dart_programming_classes.htm