Task: What is the role of an abstract class in Dart? How is it different from a concrete class or a mixin?

An abstract class in Dart is used to define which fields and/or methods should a class that implements it have. I think of it like a blueprint – it uses inheritance to guide the class writer and tells him what this class should do. When a class is written in an image of an abstract class, a keyword “implements” is used: class Car implements Vehicle {}

A car is a type of vehicle. All vehicles are able to move at a certain speed, so each vehicle should have that field. Vehicles also have their different means of starting – putting a key into the ignition in a car, or turning the pedals on a bike. The abstract class Vehicle in this case, would require the implementing class (Car) to have the field “speed” and method “start”. How exactly those methods are written (implemented) is not specified, so they can be different for each vehicle like in the previous example with the bike.

The difference from a concrete class (which is just a regular class – Car in this case) is apparent after this explanation – a concrete class contains its own functions and fields that are used for whatever, while an abstract class simply tells what a given class should contain in itself.

A mixin is different from an abstract class in that it doesn’t use inheritance and doesn’t provide “guidance”. In short, it is a class that contains a bunch of code that other classes can use whenever they add a mixin using the “with” keyword: class Car with Engine

Where Engine is a mixin class that could have methods and properties like start and stop, power and such. This can then be used by the class Car to, for example, calculate the speed using Engine’s power.