I. **What we mean by coding against interface rather than class? and if u get it so What we mean by code against abstraction not concreteness?**

Interface is only a code contract and only contains the signature of methods and not the body or elementation of this method. On the other hand, the class contains both the signature and body of methods. So when we code against interface rather than class, this means that we prefer using interfaces than class so that we can use the same name of methods and implement whatever required for each class, that increases flexability and readability as I am no longer detained with the implemented method defined in the class ,but for each class I can make a new body with the same name so I don't need to remember each method name for each class..

Abstraction is a concept to hide all complex details of a method and provide a simple interface to be dealed with rather than a complete, detailed system that can be confusing. This technique helps to avoid dealing with unnecessary details of an interface and focus on the important ones. Also it helps when changing anything in the system, we had to manually apply the changes to all parts that apply that change in the code, but when using this concept, you can do whatever changes you want and don't worry to track it as you use different naming in each class. That eases maintenance very much.

The main concept is to generalize the code so that it can be written once and used any time in any part of code without needing to write another code.

II. **What is abstraction as a guideline and how we can implement this through what we have studied?**

The abstraction is a concept to hide all complex details and provide a simple interface user deals with it. It makes the user focuses on what does it do rather how does it do. The application of this in classes gives us an abstract class that is nearly like interface, they both provide methods' signature. In case of abstract classes, the inherited classes must implement the configuration provided by the abstract class for abstract methods only and defined methods don't need this as it is fully defined before. This increases flexibility, encapsulation and maintainability