**Abstraction in OOPs**

**Introduction:**

Abstraction refers to the practice of hiding implementation details and providing a simplified view of a system to its users. It is used to simplify complex systems by exposing only the necessary features and behaviors, while hiding the underlying complexity.

**Real-life Example**

When you use a TV-Remote, you interact with buttons like power, volume, and channel without needing to know how they work internally. The remote abstracts away the complexities of the TV's internal circuitry, providing a simplified interface for controlling it.

Abstraction is a fundamental concept achieved through abstract classes and interfaces, but before that I will explain the distinction between the word concrete and abstract : concrete is something final and complete, while abstract is incomplete or pending. In programming, an abstract class serves as a blueprint for other classes, containing both abstract and concrete methods. Abstract methods lacks definition and hence abstract classes cannot be instantiated. Concrete classes on the other hand provides definition for all its methods and can be instantiated. The syntax for defining abstract classes and methods involves using the "abstract" keyword in the class and method declaration. If a class contains one or more abstract methods, it must also be declared as abstract. Any class extending an abstract class must provide concrete definition for all abstract methods, ensuring completeness.

Conversely, interfaces in Java provide a parallel mechanism for achieving abstraction. Unlike abstract classes, interfaces solely consist of abstract method declarations, executing 100% abstraction. They serve as contracts, specifying a set of methods that implementing classes must define.

**Java Example**

"Consider a 'Vehicle' class in Java. It's like driving a car – you don't need to know the engine's inner workings. In the 'Vehicle' class, you'd have methods like 'accelerate()' and 'brake()', which mimic the actions you take while driving. in Java, using these methods lets you control the vehicle without dealing with complex details."

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Example In Android Development**

Imagine you're using a smartphone app to order food. You select items from a menu and tap 'Order.' What's happening behind the scenes – the process of contacting the restaurant, confirming the order, and managing the delivery – is hidden from you. You're abstracted from those complexities.