**Abstract Class: -** To reuse the Common Code amongst many Derived Classes.

* An Abstract Class cannot be Instantiated i.e. we cannot create an object of an Abstract Class.
* An Abstract Member cannot provide a Default Implementation. It should be provided by the Derived Classes by Overriding.
* An Abstract Member cannot be marked as PRIVATE. It can be either marked as **PROTECTED**/**INTERNAL**/**PUBLIC**.
* An Abstract Class can have Abstract as well as Non-Abstract members.
* The Access Modifier of an Abstract Member should be the same in both Abstract Class and its Derived Classes.
* Derived Class must have to provide an implementation for the Abstract Members. Otherwise we'll get a compile time error.
* An Abstract Class cannot be marked as **STATIC**/**SEALED**.
* An abstract class has a **PROTECTED** Constructor (by default) allowing derived types to initialize it.
* An Abstract Member cannot be marked as **VIRTUAL**.
* An Abstract Class can have VIRTUAL methods.
* An Abstract Class can implement an interface.
* If some functionality is going to be the same for all the Derived Classes, then that functionality should be Non-Abstract in the Abstract class so that it cannot be overridden in Derived Classes and they must have to use it as it is.
* A derived class can override a base class member only if the base class member is declared as [VIRTUAL](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/virtual) or [ABSTRACT](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/abstract).
* An Abstract Class can contain constructors.

<https://www.codeproject.com/Articles/6118/All-about-abstract-classes>