* **final**:
* When applied on class, we can’t inherit it or there can’t be any subclass of a final class.
* When applied on method, it can’t be overridden.
* When used on a property, its value can’t be modified.
* **default**:
* When applied on member function in an interface, it can’t be overridden by the class implementing the interface.
* **Interface**:
* Contains definitions for member functions.
* Member functions marked as ***default*** can be defined in an interface.
* All member functions are abstract and public by default in an interface.
* Multiple interfaces can be implemented by a class, using ***implements*** keyword.
* An interface can extend multiple interfaces by using ***extends*** keyword.
* We can’t instantiate an interface.
* We use interface when we have all member functions as abstract only.
* **Abstraction(hiding the implementation)**:
* We do not show the logic of the application.
* We achieve abstraction using class when we have combination of abstract and non-abstract methods/properties in a class. Otherwise we achieve it by using interface.
* **Encapsulation(data-hiding + Abstraction)**:
* We prevent the getting/setting of member properties through objects.
* We achieve encapsulation by defining the data members as private, which can be accessed only by using getters and setters for the data methods.
* **Runtime Polymorphism:**
* Assigning the object of subclass to base class during runtime.