Type of Classes

- **Concrete Class**
    - can be created using '**new**' keyword
    - all methods should have implementation
    - It can be child class from interface or extend abstract class
    - A **class access modifier** can be **public** or **package private (default)**
- **Abstract Class (0 to 100% Abstraction)**
    - abstraction can be achieved through **interface** or **abstract class**
        - interface - 100% abstraction
        - Abstract class - 0 to 100% abstraction
    - keyword '**abstract**'
    - In child classes we can add more abstraction or implementation of parent abstract methods (concrete class)
    - no objects of abstract class can be created
        - though reference of child class can be stored into parent abstract class
- **Super and Sub Class**
    - **Parent** - super class
    - **Child** - sub class
    - **Object class** - in the absence of explicit super class, the class is implicitly a subclass of Object class
        - A parent of all the classes
        - It has some common methods
- **Nested Class**
    - A class within another class
    - If a class will only be used by only one class, we can make first class nested inside second class
    - helps in grouping logically related classed in one file itself
    - scope is same as its outer class
    - we can have class **inheritance** between two nested class inside parent class
    - we can also extend static nested class to some other class outside parent class
    - for inheritance of inner non-static class to some other class outside parent class,
        - we need to first create an object of parent of inner class in inherited class
        - then we call super method on object of parent class of inner class

Types of Nested Class : -

1. Static Nested class
    - **OuterClass.NestedClass nestedObject = new OuterClass.NestedClass()** (creating obje of static nested class)
    - Outer class cannot be **protected** or **private**
    - **but nested class can be private, public, protected or default**
    - no objects of private nested class can be created outside of parent class
2. Non - Static Nested class / Inner class
    - Nested class without static keyword

- Need object of parent class to get access to inner class
   - **OuterClass.InnerClass innerObject = outerClassObj. new InnerClass()**

1. Local Inner class
   - Inner class created inside block scope (if condition, loops)
   - It can not be public, private, protected
   - Only default is allowed
   - Cannot be initiated outside the block, where it is defined
2. Member Inner class
   - inner class inside outer class
   - It can be public, private, protected or default
3. Anonymous Inner class
   - An inner class without a name
   - Used when we want to override behaviour of method without creating subclass
   - Anonymous subclass gets created and name is given by compiler