**Abstraction:**

Abstraction is the concept to hide the data and implementation of the data inside method. Like Age Calculation, it simply hides the process how age is being calculated. If we provide the DOB then it will return the age.

**Encapsulation:**

Encapsulation is the process or technique to implement the Abstraction. It also restricts the access of member to outside world. It uses access modifier public, private, protected and default to achieve the encapsulation.

**Inheritance:**

Inheritance is the process or technique by which one object inherits all the property of other object. The object is being Inherited is called super Object and the child class is called sub object. This mechanism is being used for code reusability.

**Polymorphism:**

Polymorphism is the process or concept to create to different objects with same name. There is two type of polymorphism Static and Dynamic polymorphism.

**Static Polymorphism**: This process or concept is being implemented by overloading the method, also known as Compile time Polymorphism.

**Dynamic Polymorphism**: This process or concept is being implemented by overriding the method, also known as Run time Polymorphism.

**List** – can have any number of **Null** element.

**Set** -Set can have only one Null Element.

**Map** – **HashMap and Linked HashMap** - only one **Null** key & multiple **Null** values,

**Non-Synchronized**, can be synchronized by **Collections.synchronizedMap(hashMap**);

**HashTable** - **doesn't allow any Null** **key or value**., it’s a **legacy class**, **Synchronized**

**TreeMap** – Does not allow **Null** key and multiple **Null** Values, by default it follow **ascending** Order. Uses below methods - tailMap(), firstKey(), lastKey(), pollFirstEntry(), pollLastEntry()**.**

We can synchronize a method by using **synchronized** keyword before method name.

**public static synchronized <return\_type> method1() { }**

The synchronized keyword can be used to mark four different types of blocks:

1. Instance methods (**Class level lock**)– add keyword **Synchronized** at method declaration
2. Static methods (**Object level lock**)– add **Synchronized** keyword at method declaration
3. Code blocks inside instance methods (**Class level lock**) - **Synchronized (ClassName.class)/(this) { Block}**
4. Code blocks inside static methods (**Object level lock**)– - **Synchronized (ClassName.class){ Block}**