Marker interface:

- If an interface doesn't contain any methods and by implementing that interface if our objects will get some ability such type of interfaces are called Marker interface (or) Tag interface (or) Ability interface.
- Serializable,Cloneable,RandomAccess,SingleThreadModel.
- By implementing a **Serializable** interface we can send that object across the network and we can save the state of an object into a file.
- By implementing **SingleThreadModel** interface Servlet can process only one client request at a time so that we can get "Thread Safety".
- By implementing a **Cloneable** interface our object is in a position to provide exactly duplicate cloned object.

Note:
Without having any methods in the marker interface how objects will get ability ?
Internally JVM is responsible to provide required ability.

When should we go for interface, abstract class and concrete class?

- If we don't know anything about implementation just we have requirement specification then we should go for an interface.
- If we are talking about implementation but not completely (partial implementation) then we should go for an abstract class.
- If we are talking about implementation completely and ready to provide service then we should go for a concrete class.

What is the Difference between interface and abstract class ?

interface

- If we don't' know anything about implementation just we have requirement specification then we should go for an interface.
- Every method present inside the interface is always public and abstract whether we are declaring or not.
- We can't declare interface methods with the modifiers private, protected, final, static, synchronized, native, strictfp.
- Every interface variable is always a public static final whether we are declaring or not following modifiers. Private, protected,transient, volatile.
- For the interface variables compulsory we should perform initialization at the time of declaration otherwise we will get compile time error.
- Inside the interface we can't take static and instance blocks.
- Inside the interface we can't take constructor.

Abstract class

- If we are talking about implementation but not completely (partial implementation) then we should go for abstract class.
- Every method present inside abstract class need not be public and abstract.
- There are no restrictions on abstract class method modifiers.
- Every abstract class variable need not be a public static final.
- There are no restrictions on abstract class variable modifiers.
- It is not required to perform initialization for abstract class variables at the time of declaration.
- Inside abstract class we can take both static and instance blocks.
- Inside the abstract class we can take constructor.