Interfaces in Java

- An interface in java is a blue print of a class.
- It has static constants and abstract methods.
- Interfaces specifies what a class do but not how.
- To declare an interface, use interface keyword.

** It is used to provide total abstraction, i.e

- all methods in interfaces are declared with empty body and are public and all fields are public, static and final by default.

* > A class that implement interface must implement all the methods declared in the interface.

* > If a class implements an interface and does not provide method bodies for all functions specified in interface, then class must be declared abstract.

- To implement interface use implements keyword.
- Java Interface also represents the IS-A relationship.
- It cannot be instantiated just like the abstract class.
- Since Java 8, we can have default and static methods in an interface.
- Since Java 9, we can have private ~~[illegible]~~ methods in interface.

> By interface, we can support the functionality of multiple inheritance.

> It can be used to achieve loose coupling.

Syntax :

```
interface <interface_name>
{
     // declare constant fields
     // declare methods that abstract
     // by default.
}
```

*** The java compiler adds public and abstract keywords before the interface method. Moreover, it adds public, static and final keywords before data members.

```
interface Printable
{
    int MIN=5;
    void print();
}
```

Printable.java

→ compiler →

```
interface Printable
{
 public static final
 int MIN=5.
 public abstract void
   print();
}
```

printable.class

Ques: Why use interfaces when we have abstract classes?

Ans: abstract classes may contain non-final variables, whereas variables in interfaces in interfaces are final, public & static.

Ques: Multiple inheritance is not supported through class in java, but it is possible by an interface, Why?

Ans: multiple inheritance is not supported in the case of class because of ambiguity. However, it is supported in case of an interface because there is no ambiguity, It is because its implementation is provided by the implementation class.

Ques: What is marker or tagged interface?

Ans: An interface which has no member is known as a marker or tagged interface for eg. Serializable, Cloneable, Remote etc.

They are used to provide some essential information to the JVM so that JVM may perform some useful operation.

JAVA NESTED INTERFACE

> An interface i.e declared within another interface interface or class is known as nested interface.

> Used to group related interfaces so that they can be easy to maintain.

> The nested interface must be referred by the outer interface or class.

> It can't be accessed directly.

> Nested interface must be public if it is declared inside the interface but it can have any access modifier if declared within the class.

> Nested interfaces are declared static implicitly.

Example of nested interface which is declared within the class :

```
class Abhi{
    interface Display{
          void show();
    }
}
public class NestedInterface implements Abhi.Display
{
    public void ~~wag~~show()
    {
        System.out.println("Implementation!!");
    }
    public static void main(String args[])
    {
        Abhi.Display ab= new NestedInterface();
        ab.show();
    }
}
```

*
Ques : Can we define a class inside the interface?

Ans : Yes, If we define a class inside the interface, java compiler creates a static nested class.

```
interface M{      classA{ }
}
```

Abstract class	Interface
1) Abstract class can have abstract and non-abstract methods.	1) Interface can have only abstract methods. Since java 8, it can have default and static methods also.
2) doesn't support multiple inheritance.	2) supports multiple inheritance.
3) can have final, non-final, static and non-static variables.	3) has only static and final variables.
4) can provide the implementation of interface.	4) can't provide the implementation of abstract class.
5) The abstract keyword is used to declare abstract class.	5) The interface keyword is used to declare interface.
6) can extend another Java class & implement multiple java interfaces.	6) An interface can extend another java interface only.
7) An abstract class can be extended using keyword "extends".	7) An interface can be implemented using keyword "implements".
8). can have class members like private, protected etc.	8) Members of a Java interface are public by default.