Sometimes, we require just method declaration in super-classes. This can be achieve by specifying the [**abstract**](https://www.geeksforgeeks.org/abstract-keyword-in-java/) type modifier. These methods are sometimes referred to as *subclasser responsibility* because they have no implementation specified in the super-class. Thus, a subclass must [override](https://www.geeksforgeeks.org/overriding-in-java/) them to provide method definition. To declare an abstract method, use this general form:

Interface in Java

An **Interface in Java** programming language is defined as an abstract type used to specify the behavior of a class. An interface in Java is a blueprint of a behaviour. A Java interface contains static constants and abstract methods.

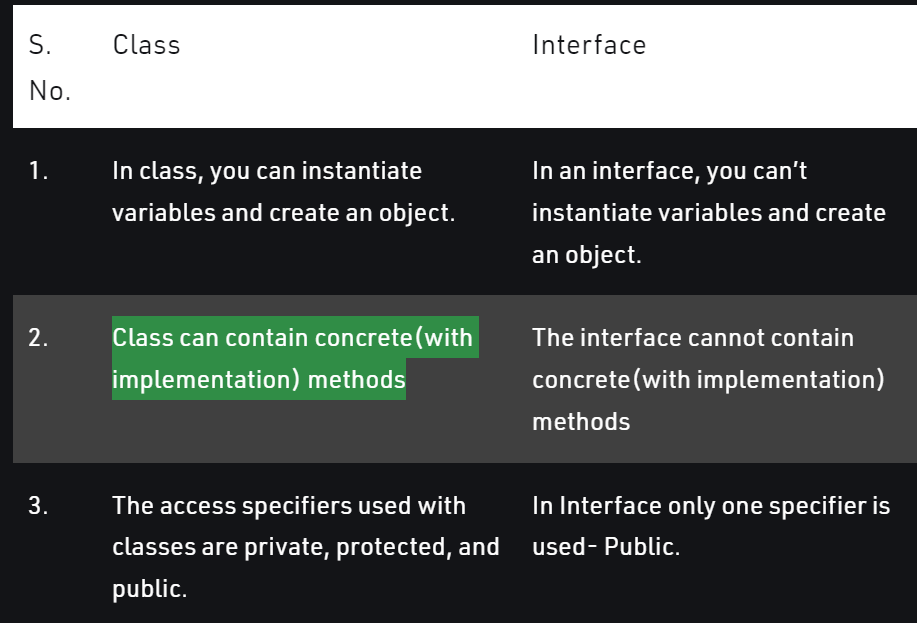
The interface in Java is *a*mechanism to achieve [abstraction](https://www.geeksforgeeks.org/abstraction-in-java-2/). There can be only abstract methods in the Java interface, not the method body. It is used to achieve abstraction and [multiple inheritance in Java](https://www.geeksforgeeks.org/java-and-multiple-inheritance/). In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body. Java Interface also **represents the IS-A relationship**.

* Interfaces specify what a class must do and not how. It is the blueprint of the behaviour.
* Interface do not have constructor.
* Represent behaviour as interface unless every sub-type of the class is guarantee to have that behaviour.
* An Interface is about capabilities like a Player may be an interface and any class implementing Player must be able to (or must implement) move(). So it specifies a set of methods that the class has to implement.
* If a class implements an interface and does not provide method bodies for all functions specified in the interface, then the class must be declared abstract.
* A Java library example is [Comparator Interface](https://www.geeksforgeeks.org/comparator-interface-java/). If a class implements this interface, then it can be used to sort a collection.

### Why do we use an Interface?

* It is used to achieve total abstraction.
* Since java does not support multiple inheritances in the case of class, by using an interface it can achieve multiple inheritances.
* Any class can extend only 1 class but can any class implement infinite number of interface.
* It is also used to achieve loose coupling.
* Interfaces are used to implement abstraction. So the question arises why use interfaces when we have abstract classes?

The reason is, abstract classes may contain non-final variables, whereas variables in the interface are final, public and static



<https://www.scientecheasy.com/2021/02/interface-interview-questions.html/>