

A class which is declared with the abstract keyword is known as an abstract class in [Java](#). It can have abstract and non-abstract methods (method with the body).

Abstraction in Java

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery.

Abstraction lets you focus on what the [object](#) does instead of how it does it.

Ways to achieve Abstraction

There are two ways to achieve abstraction in java

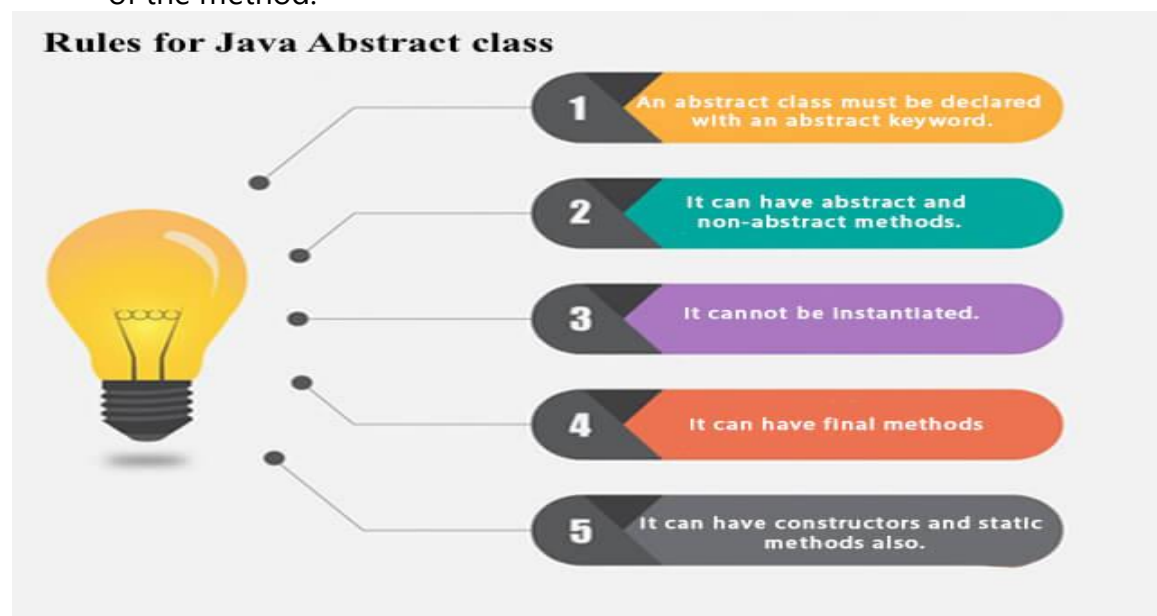
1. Abstract class (0 to 100%)
2. Interface (100%)

Abstract class in Java

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

Points to Remember

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have [constructors](#) and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.



Example of abstract class

1. **abstract class** A{

Abstract Method in Java

A method which is declared as abstract and does not have implementation is known as an abstract method.

Example of abstract method

1. **abstract void** printStatus();//no method body and abstract

Example of Abstract class that has an abstract method

In this example, Bike is an abstract class that contains only one abstract method run. Its implementation is provided by the Honda class.

```
abstract class Bike
{
    abstract void run();
}
class Honda4 extends Bike
{
    void run()
    {System.out.println("running safely");
    }
    public static void main(String args[])
    {
        Bike obj = new Honda4();
        obj.run();
    }
}
```