Abstraction
=============

- **Abstraction** is a process of hiding the internal implementation details and just highlighting/ showing only the functionality to the user.

- E.x User uses the ATM to withdraw the money by using an ATM card, but the user doesn't know the internal implementation.

- E.x Sending SMS where the user types the text and sends the message, but the user doesn't know the internal processing about the message delivery.

There are two ways to achieve abstraction in java

a. Abstract class (0-100% abstraction achieved)
b. Interface (100% abstraction achieved)

—-----------------

a. **Abstract class**

   i. A class which is declared by using abstract keywords is known as an **abstract class**.
   ii. Rules for abstract class
      - It contains abstract methods (incomplete method) and non abstract method (complete method).
      - We can not create object of abstract class.
      - To create an object of an abstract class programmer needs to complete the all incomplete method into concrete class(sub class).
      - If an abstract class contains 10 abstract methods and in subclass only 9 complete methods then that subclass is also called an abstract class.
   iii. Concrete class
      - A subclass which completes the implementations of all incomplete methods present in abstract class is called a concrete class.

E.x

Superclass >>

```
public abstract class TestingClass
{
        public void test1()                          // Abstract/Complete method
        {
                System.out.println("Super class");
        }
        public abstract void withdraw();             //Non abstract/Incomplete method

        public abstract void test2();                //Non abstract/Incomplete method
}
```

Subclass >>

```
public class Subclass extends TestingClass       // Concrete class(subclass)
{
        public void withdraw()        // provided implementation to incomplete method
        {
                System.out.println("completed method withdraw");
        }

        public void test2()           // provided implementation to incomplete method
        {
                System.out.println("print test2");
        }
        public static void main(String[] args)
         {
                Subclass sub = new Subclass();
                sub.test1();                    // calling complete method
                sub.test2();                    // calling incomplete method
                sub.withdraw();                 // calling incomplete method
        }
}
```

Output :

```
Super class
completed method withdraw
print test2
```

—-----------------