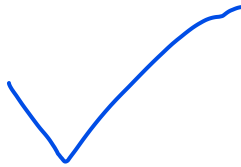# Abstract class in Java

Abstract class :

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).

It needs to be extended and its method implemented. It cannot be instantiated.

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.

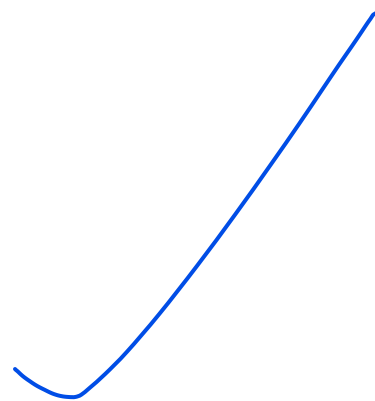**Abstraction lets you focus on what the object does instead of how it does it.**

Ways to achieve Abstraction :

There are two ways to achieve abstraction in java

- Abstract class (0 to 100%)

- Interface (100%)

Ways to achieve Abstraction :

There are two ways to achieve abstraction in java
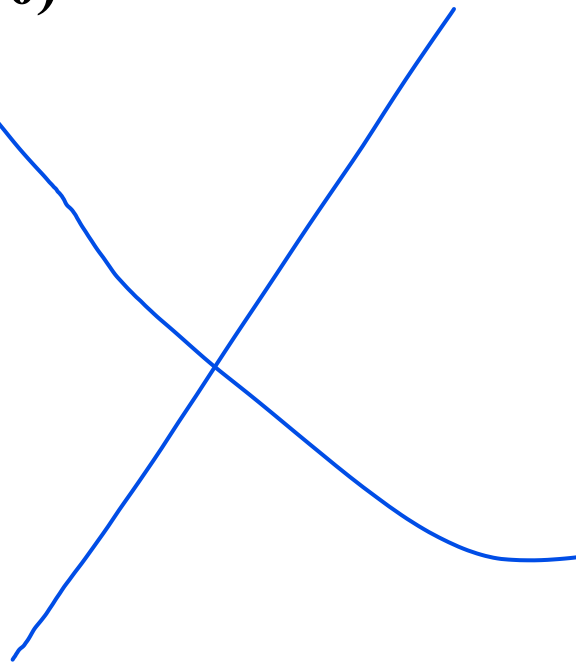
- **Abstract class (0 to 100%)**
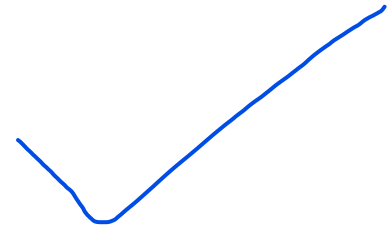
- **Interface (100%)**

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.
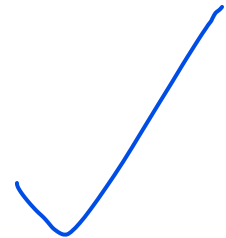- It can have final methods which will force the subclass not to change the body of the method.

Abstract class in Java

Example of abstract class

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

abstract void printStatus();//no method body and abstract

Abstract Method in Java :

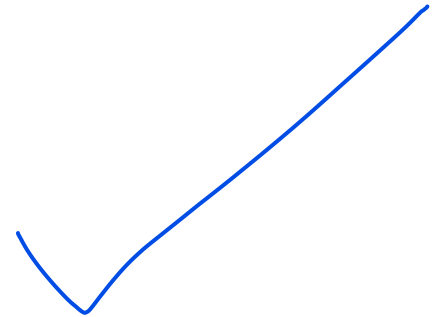**A method which is declared as abstract and does not have implementation is known as an abstract method.**

Example of Abstract class that has an abstract method

```
abstract class Bike{
  abstract void run();
}
class Honda extends Bike{
void run(){System.out.println("running safely");}
public static void main(String args[]){
 Bike obj = new Honda();
 obj.run();
}
}
```
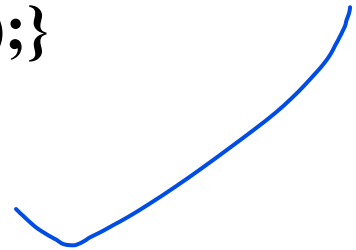
Example of Abstract class that has an abstract method

```
abstract class Shape
{
abstract void draw();
}
//In real scenario, implementation is provided by others
class Rectangle extends Shape
{
  void draw()
    {System.out.println("drawing rectangle");}
}
```

Abstract class in Java

Example of Abstract class that has an abstract method

```
class Circle extends Shape
{
  void draw()
  {System.out.println("drawing circle");}
}
  //In real scenario, method is called by programmer or user
   class TestAbstraction
  {
  public static void main(String args[])
  {
  Shape s=new Circle();
  s.draw();
  }
```

Abstract class in Java

Example of Abstract class that has an abstract method

```
abstract class Bank{
abstract int getRateOfInterest();
}
class SBI extends Bank{
int getRateOfInterest(){return 7;}
}
class PNB extends Bank{
int getRateOfInterest(){return 8;}
}
```

Abstract class in Java

Example of Abstract class that has an abstract method

```
class TestBank{
public static void main(String args[]){
Bank b;
b=new SBI();
System.out.println("Rate of Interest is:
"+b.getRateOfInterest()+" %");
b=new PNB();
System.out.println("Rate of Interest is:
"+b.getRateOfInterest()+" %");
}}
```

Abstract class in Java

Rule: If there is an abstract method in a class, that class must be abstract.

```
class Bike{
abstract void run();
}
```

compile time error

Rule: If you are extending an abstract class that has an abstract method, you must either provide the implementation of the method or make this class abstract.

Abstract class in Java

Abstract class having constructor, data member and methods

```
abstract class Bike
{
   Bike(){System.out.println("bike is created");}
   abstract void run();
   void changeGear(){System.out.println("gear changed");}
}
//Creating a Child class which inherits Abstract class
class Honda extends Bike
{
 void run(){System.out.println("running safely..");}
}
```
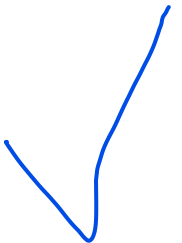
Abstract class in Java

Abstract class having constructor, data member and methods

//Creating a Test class which calls abstract and non-abstract methods

```
class TestAbstraction{
public static void main(String args[]){
 Bike obj = new Honda();
 obj.run();
 obj.changeGear();
 }
}
```