Abstract Class:-

Abstract class is a type of class in java which has complete as well as incomplete methods declared inside it.

Or we can say that an Abstract class have at least one complete and one incomplete method in it. We cannot create object of abstract class because to create an object of class all the method inside the method should be complete.

Now the class where we complete this incomplete methods of abstract class is known as concrete class.

Concrete class:-

As we know that we cannot create the object of abstract class because of abstract/incomplete method present inside it, so when we inherit this Abstract a class in a subclass and provide implementation to its incomplete method, this subclass where this incomplete method are completed is called as Concrete class.

Or we can say that the class in which the abstract methods/incomplete methods are completed are known as Concrete class.

A class will called concrete class only if it completing all the methods of abstract class.

Note:- Abstract keyword is used to declare a abstract class, but there is no keyword for Concrete class.

Abstract class does not have constructor in it, not even default constructor

Abstract Class

```
Public abstract class Demo{
Public void Monday(){
Syso(“Hello”)
}
}
Public void Tuesday(){
Syso(“Hello”)
}

Public abstract void Wednesday();
Public abstract void Thursday();
}
```


Concrete Class

```
Public class implementation extends
Demo{
Public void wednesday(){
Syso(“Wednesday”)
}
Public void Thursday()
{
Syso(“thursday”)
}
```

Abstract Class:-

```
public abstract  class AbstractClass {
public abstract void test();

public abstract void test1();

}
```

Concrete Class :-

```
public abstract class Concrete extends AbstractClass  {
public void test() {
System.out.println("Hello");
}

public void test1() {
System.out.println("Hi");
}
}
```

```
public class SubClass extends ParentClass {
public void test() {
System.out.println("Hello Java"); //Override
}

public void UsingSuper() {
super.test();  //points to test() method of super class
this.test();  //points to the test() method of subclass
}

public static void main(String[] args) {
SubClass obj = new SubClass();
obj.UsingSuper();

}
```

Identifiers :-

In general terms, an Identifier is simply a name given to an entity for identification. The name of a person. The employee number of an employee. The social security number of an individual. Any attribute/property of an entity that is used for identification can be called an identifier. In Java, identifiers are the names that are used for the identification of various entities in java like classes, methods, variables, packages, etc. So in Java, Identifiers are the Class names, Method names, variable names, package names, constant names, etc. Each of these identifiers is declared following a particular syntax and a naming convention.

int <variable name(Identifier)>;
The <Variable name> is the name given by the programmer to uniquely identify the int variable. Later the programmer can use that variable in his program. Hence, here the <Variable name> is the identifier.

Identifiers: Below is the list of Identifiers that are present in the above sample code.
- MainClass (Class name)
- main (Method name)
- String (Predefined Class name)
- System(Predefined Class name)
- println (Method name)