Interface

It is one of the oops principle.

It is pure 100% abstract in nature.

Interface is use to declare only incomplete methods in it.

Features of Interface:

1. methods declared inside Interface are by default public & abstract.

2. Data Members/variable declared inside Interface are by default static and final.

3. Constructor concept in not present inside Interface.

4. Object of Interface cannot be created.

5. To create object of Interface programmer, need to make use of Implementation class using implements keyword.

6. Interface support multiple inheritance.

Implementation class:

A class which provides definitions for all the incomplete methods which are

present in interface with the help of "implements" keyword is called Implementation class.

```
package Interface;
public interface Demo1
{
		int num=10;		//static final int num=10;

		void m1();					//abstract public void m1();

		void m2();					//abstract public void m2();
}

package Interface;
//implementation class
public class Demo2 implements Demo1
{
	//static final int num=10;

	public void m1()
	{
		System.out.println(num);
		System.out.println("method m1 complete in implementation class");
	}

	public void m2()
	{
		System.out.println("method m2 complete in implementation class");
	}
}

package Interface;
public class TestDemo2
{
	public static void main(String[] args)
	{
		Demo2 d2=new Demo2();
		d2.m1();
		d2.m2();
	}
}
```

Example multiple inheritance using interface

```
package Interface;
//super interface1
public interface I1
{
	void m1();
	void m2();
}


package Interface;
//super interface2
public interface I2
{
	void m3();
	void m4();
}


package Interface;
//sub class or implementation class
public class Sample implements I1, I2
{
	//example multiple inheritance using interface

	public void m1()
	{
		System.out.println("method m1 from interface I1 completed in IC");
	}

	public void m2()
	{
		System.out.println("method m2 from interface I1 completed in IC");
	}

	public void m3()
	{
		System.out.println("method m3 from interface I2 completed in IC");
	}

	public void m4()
	{
		System.out.println("method m4 from interface I2 completed in IC");
	}
}

package Interface;

public class TestSample
{
	//example multiple inheritance using interface

	public static void main(String[] args)
	{
		Sample s=new Sample();
		s.m1();
		s.m2();
		s.m3();
		s.m4();
	}
}
```

Questions
1: can 1 class extends another class? yes
2: can 1 interface extends another interface? yes
3: can 1 interface implements another interface? No
4: can we provide method definition in interface? yes - using default method
5: what is default method in interface?
6: can we create static method in interface? yes (To define general utility methods)
7: can we create main method in interface? yes
8: can we create private method in interface? yes

up to java version 1.7 -> only public method allowed
v 1.8 -> default & static method allowed in interface
v 1.9 -> private methods also allowed

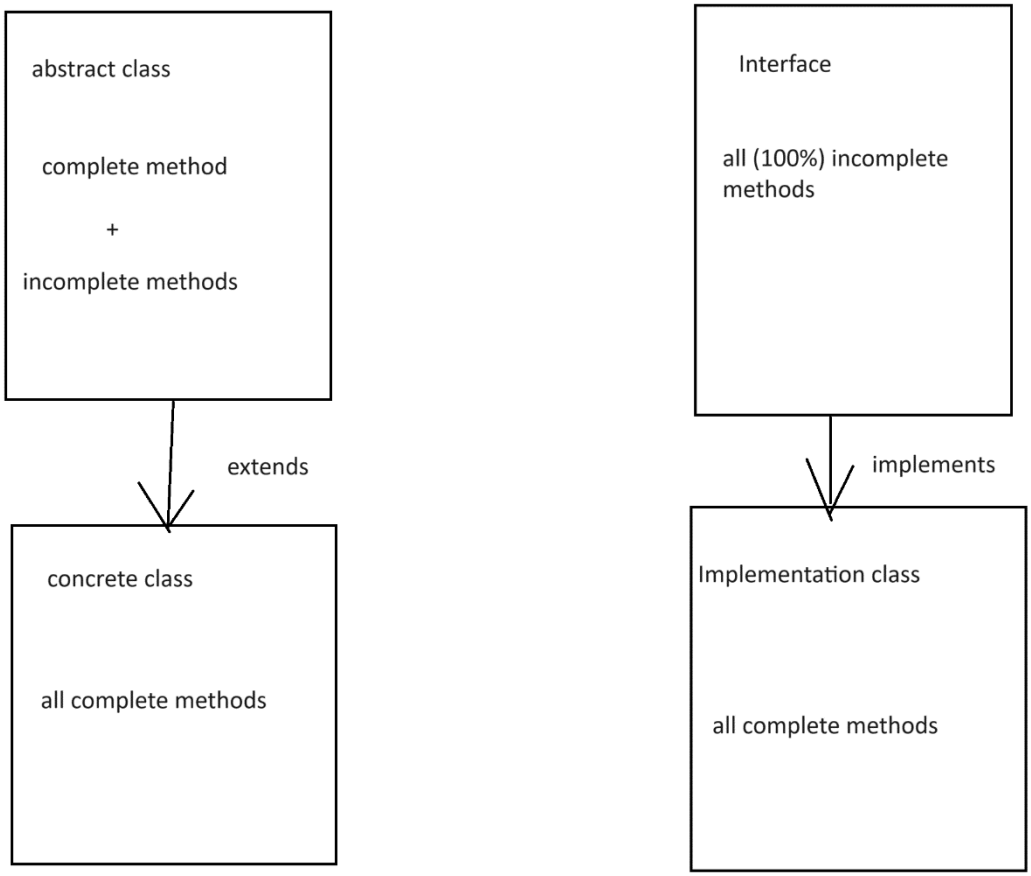
default method in interface: -

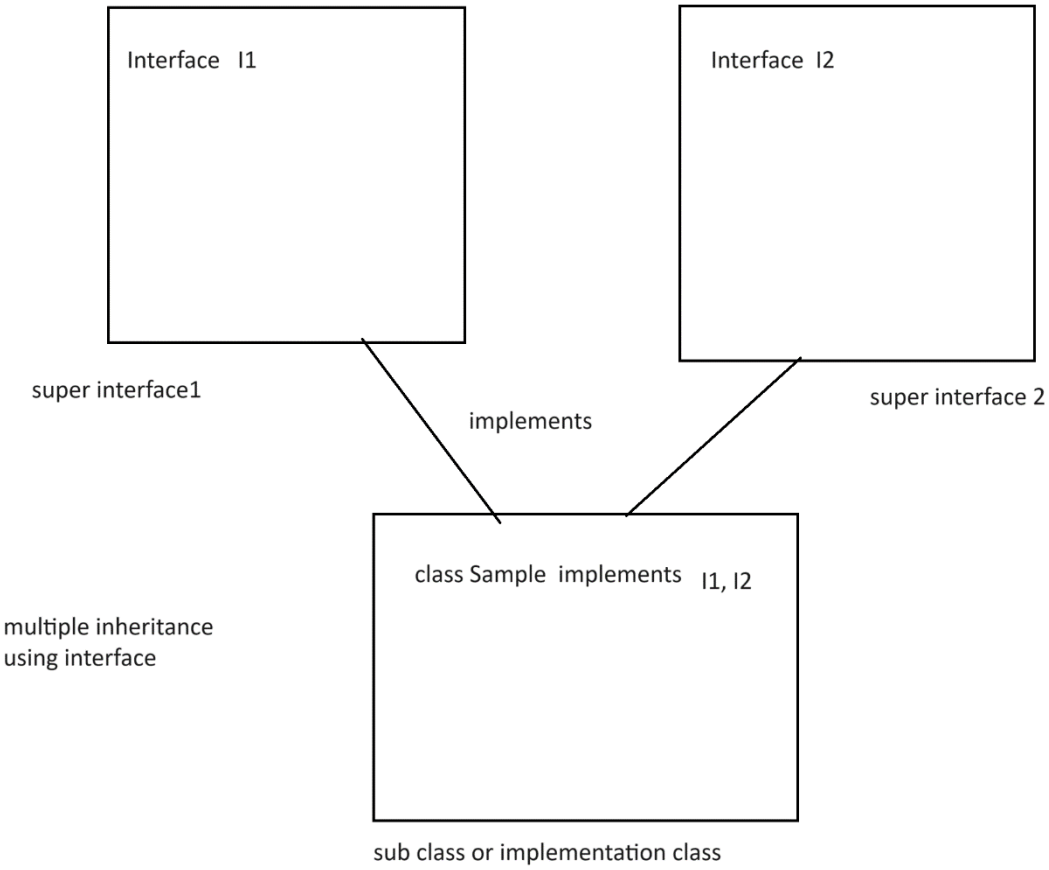
Default methods enable you to add new functionality to existing interfaces

Without affecting implementation classes if we want to add new method to the Interface we create default method in interface
Need to add default keyword with method.

```
public interface Test1
{
	void m1();

	void m2();

	default void m3()	//default method
	{
		System.out.println("running default method in interface");
	}
}


public class Test2 implements Test1
{
	public void m1()
	{
		System.out.println("method m1 in completed in IC");
	}

	public void m2()
	{
		System.out.println("method m2 in completed in IC");
	}

	public static void main(String[] args)
	{
		Test2 t2=new Test2();
		t2.m1();
		t2.m2();
		t2.m3();
	}
}
```

Note: here word default is not a modifier
why word default:- because this method is having default implementation


abstract class
complete method
+
incomplete methods
extends
concrete class
all complete methods
Interface
all (100%) incomplete methods
implements
Implementation class
all complete methods


Interface I1
Interface I2
super interface1
implements
super interface 2
class Sample implements I1, I2
multiple inheritance
using interface
sub class or implementation class