Abstraction→Data Hiding

1.abstract class→0% to 100%

2.Interface→100%

Interface

1.interface is declared by the keyword interface followed by its Name.

2.It is also a block of code.

3.An Interface is not a Class.

4.An Interface does not have its own Constructor.

5.Interface cannot be instantiated (Object of Interface Cannot be created).

6.An Interface can contain an abstract method.

7.An Interface should be implemented with a class

using the keyword "implements".

8.To Define an abstract method declared inside an interface, in Concrete class your will require the keyword word/Access Modifier/Specifier i.e. "public".

```
interface Inter1
{
        abstract void method1();
}
public class MyClass implements Inter1
{
        public void method1()
        {
                System.out.println("Method 1 implemented from Inter1");
        }
        public static void main(String[] args)
        {
                MyClass obj=new MyClass();
                obj.method1();
        }
}
```

Output:

```
Method 1 implemented from Inter1
```

Q: Is the "abstract" keyword required to declare an abstract method inside an Interface?

Ans: No "abstract" keyword is not required to declare

an abstract method inside an interface.

Q: What is the default Access Modifier of Interface Variable?

Ans: public

* All the variables are public in an Interface.

* if public keyword is not used the public property is

set implicitly(internally).

Q: Can you change the value or definition of an Interface Variable?

Ans: Every Variable in an Interface is final.

final means constant. E.g pi=3.14;

* final keyword is used to make a value constant.

9. Variable inside an Interface are by default public, final, static.

10.Even an Interface can contain a Static Method().

11.A Concrete method is declared using the keyword

"default" inside an interface.

```
interface Inter1
{
        public final static int a=23;
  // explicitly every variable inside interface is static, public
        int c=100;
// implicitly every variable inside interface is static, public

        abstract void method1(); // "abstract" keyword is not necessary


//default keyword is use to declare and define a concrete method in an Interface
        default void normal()
        {
                System.out.println("Normal method of Inter1");
        }

        static void stat()// static method
        {
                System.out.println("\nStatic method of Interface\n");
                System.out.println("a is:"+a);
                System.out.println("c is:"+c);
// static method only allows Static members, 'a' and 'c' could be accessed because they are static
implicitly.
        }

}
public class Abstraction2 implements Inter1
{
        final  int b=10; //constant
        public void method1() //public keyword is used to define an abstract method of an
Interface
        {
        System.out.println("Inter1 method implemented  in Class");

        }
        void value()
        {
                //b=b+b; b=20→Final Variable cannot change its value.
                System.out.println("Main Class Method");
                System.out.println("b is:"+b);
                System.out.println("c is:"+c);
        }
```

```java
	public static void main(String[] args)
	{
		Abstraction2 obj=new Abstraction2();

		obj.value();//Main class Method
		System.out.println("\n-----------------------\n");
		obj.normal();//Inter1 default method
		System.out.println("\n-------------------------\n");
		obj.method1();//Inter1 abstract method
		System.out.println("\n--------------------------\n");

		Inter1.stat();
		// Static Method can be called using name of the Interface in which it is declared

		System.out.println("\n*********************************\n");
		System.out.println("Method Accessed by Reference Variable of 
Interface:-\n");

		Inter1 i1=new Abstraction2();
//Interface reference variable=new MainClassConstructor();

		i1.method1();
//abstract method of Interface implemented in Main Class

		System.out.println("\n--------------------\n");
		i1.normal();

		//i1.stat();→Cannot access static method of interface using Interface Reference 
Variable

		//i1.value();-→NOT Possible because value() is Main Class Method and not 
Interface Method
		//Interface Variable can access only its own data Members and not others

	}
}
```

Smit IT Institute
Gives you wings to fly high

Output :

```
Main Class Method  →Called using obj.value();
b is:10
c is:100

-----------------------

Normal method of Inter1 →Called using obj.normal();


-------------------------

Inter1 method implemented  in Class →Called using obj.method1();


--------------------------

Static method of Interface →Called using Inter1.stat();


a is:23
c is:100

*********************************

Method Accessed by Reference Variable of Interface:-

Inter1 method implemented in Class →Called using i1.method1();

---------------------

Normal method of Inter1   →Called using i1.normal();
```