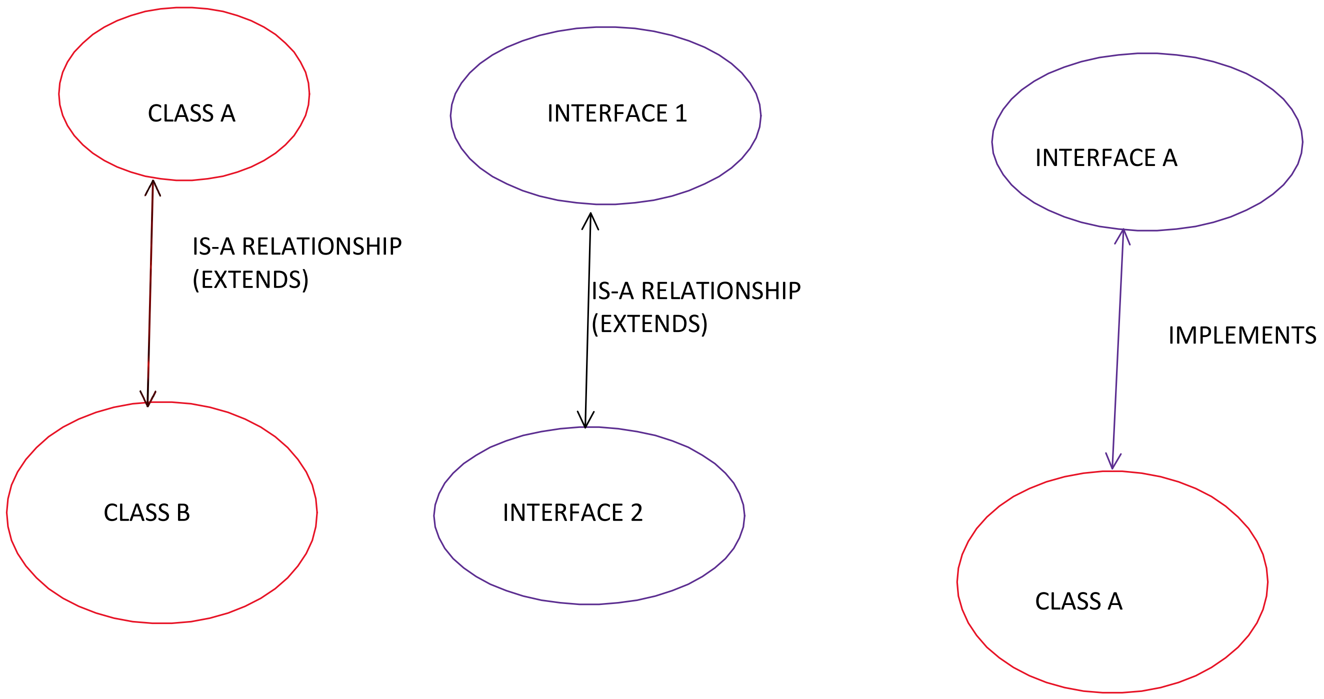
DAY-39

10 August 2023 15:40



- SINCE , AN INTERFACE CONTAINS ONLY ABSTRACT METHODS , WE PROVIDE THE IMPLENTATION FOR THE ABSTRACT METHOD IN THE IMPLEMENTATION CLASS.
- A CLASS AND CLASS HAVE A **IS-A RELATIONSHIP.**
- AN INTERFACE AND AN INTERFANCE CAN HAVE **IS-A RELATIONSHIP**
- BUT AN INTERFACE AND A CLASS WILL HAVE IMPLEMENTS RELATIONSHIP.

➢ JUST LIKE ABSTRACT CLASS, WE CAN NOT CREATE AN INSTANCT .OBJECT OF AN INTERFACE

```
package interfaceDemo;

public interface Demo2
{
	String s = "BitsQ";
	void m2();

	public static void main(String[] args)
	{
		Demo2 a1 = new Demo2(); //WE CAN NOT CREATE OBJECT OF INTERFACE
	}

}
```

➢ IF THE IMPLEMENTATION CLASS FAILS TO PROVIDE IMPLEMENTATION FOR ALL THE ABSTRACT METHODS OF AN INTERFACE THEN THE IMPLEMENTATION CLASS SHOULD BE DECLARED AS ABSTRACT AND THE IMPLEMENTATION SHOULD BE PROVIDED IN ITS FURTHER SUBCLASS.

```
}

package interfaceDemo;

public interface Demo3
{
        String s = "Java";
        void add();
        void sub();
        void prod();
}

package interfaceDemo;

public abstract class ImpClass1 implements Demo3
{
        public void add()
        {
                int x =1, y=2;
                System.out.println("sum of a and b is : "+(x+y));
        }
}

package interfaceDemo;

public abstract class ImpClass2 extends ImpClass1
{
        public void sub()
        {
                int x =1, y=2;
                System.out.println("difference of a and b is : "+(x-y));
        }

}

package interfaceDemo;

public class ImpClass3 extends ImpClass2
{

        public void prod()
        {
                int x = 1, y = 2;
                System.out.println("product of a and b is "+(x*y));
        }
        public static void main(String[] args)
        {

                ImpClass3 i = new ImpClass3();
                i.add();
                i.sub();
                i.prod();
        }

}
```

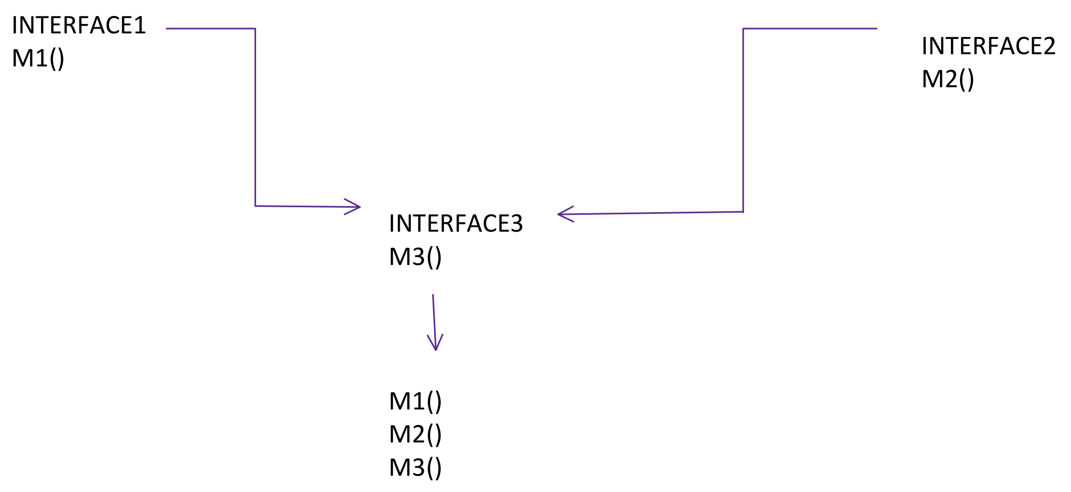
INTERVIEW QUESTIONS :

1. WHAT ARE CONCRETE METHODS?
2. WHAT IS CONCRETE CLASS ?
3. WHAT IS ABSTRACT METHODS ?
4. WHEN WILL YOU DECLARE A METHOD AS AN ABSTRACT ?
5. WHAT IS AN ABSTRACT CLASS
6. CAN WE CREATE OBJECT OF AN ABSTRACT CLASS ?
7. WHY CAN'T WE CREATE OBJECT OF AN ABSTRACT CLASS ?
8. WHERE TO PROVIDE IMPLEMENTATATION OF ABSTRACT METHOD OF ABSTRACT CLASS ?
9. WHAT HAPPENS IF SUBCLASS FAILS TO PROVIDE THE IMPLEMENATION OF PARENT CLASS ABSTRACT METHOD ?
10. CAN WE DECLARE STATIC METHODS AS ABSTRACT ?
11. WHY CAN'T WE DECLARE STATIC METHODS AS ABSTACT ?
12. WHAT HAPPENS IF WE DECLARE CONCRETE METHOD AS NON STATIC >

- A CLASS CAN IMPLEMENT MULTIPLE INTERFACE AT A TIME AS SHOWN

```
package interfaceDemo;

public interface Interface1
{
	void even();
}


package interfaceDemo;

public interface Interface2
{
	void odd();

}

package interfaceDemo;

public class InterfaceImp implements Interface1, Interface2
{
	public void even()
	{
		int x = 2;
		if (x%2 == 0)
		{
			System.out.println("even");

		}
	}

	public void odd()
	{
		int x = 3;
		if (x%2 != 0)
		{
			System.out.println("odd");

		}
	}

	public static void main(String[] args)
	{
		InterfaceImp i = new InterfaceImp();
		i.even();
		i.odd();
	}
}
```

MULTIPLE INHERTIANCE IN INTERFACE



- WITH RESPECT TO CLASSES, A SINGLE CLASS CAN NOT EXTEND FROM MORE THAN 1 SUPER CLASS.
- BUT WITH RESPECT TO INTERFACES, A SINGLE INTERFACE CAN EXTEND FROM MORE THAN 1 INTERFACE
- SINCE , INTERFACES DO NOT ALLWO CONSTRCUTORS , SO THERE IS NOT CONCEPT OF CONSTRCUTOR CHAINING HERE.
- DUE TO WHICH, WE CAN ACHIEVE MULTIPLE INHERITANCE WITH RESPECT TO ITNERFACES, BUT NOT CLASSES

```
package interfaceDemo;

public interface IntParent1
{
	void m1();
}

package interfaceDemo;

public interface IntParent2
{
	void m2();
}

package interfaceDemo;

public interface MultiChild extends IntParent1, IntParent2
{
	void m3();
}

package interfaceDemo;
```

```
public class MultiImp implements MultiChild
{
	public void m1()
	{
		System.out.println("in m1 of parent1");
	}

	public void m2()
	{
		System.out.println("in m2 of parent2");
	}

	public void m3()
	{
		System.out.println("in m3 of child");
	}
	public static void main(String[] args)
	{
		MultiImp m = new MultiImp();
		m.m1();
		m.m2();
		m.m3();

	}

}
```