質問１　サブクラスでオーバーライドしているメソッド内で super.修飾子を用いてスーパークラスのオーバーライドされたメソッドを呼び出すと、無限ループになりませんか？

回答
まず、質問の内容を例を用いて明確にすると、「次のようなコードを作った場合〝B〟が出力され続けることになりませんか？」となります。

```
class A
{
    public void func(){
        System.out.println(〝A〟);
    }
}

class B extends A
{
    public void func(){
        System.out.println(〝B〟);
        super.func();
    }
}

class Test1
{
    public static void main(String[] args){
        A aa=new B();
        aa.func();
    }
}
```

クラスＡはメソッド func()を持ちます。クラスＢはクラスＡを継承しています。クラスＢで同じメソッド func()を宣言していますので、これらのメソッドはオーバーライドになっています。すなわち、クラスＡの func()が呼ばれると、自動的にクラスＢの func()が呼ばれます。

クラスＡのメソッド func()は、〝A〟を画面に出力します。クラスＢのメソッド func()は、〝B〟を画面に出力した後、super.修飾子でクラスＡのメソッド func()を呼び出します。

さて、メインメソッドでは、クラスＢのオブジェクトを生成して、クラスＡの変数に代入し、クラスＡのメソッド func()を呼び出しています。

オーバーライドの動作をそのまま用いて考えると、クラスＡのメソッド func()が呼ばれるとそれは実行されずにクラスＢのメソッド func()が呼び出され実行されるので、

B
B
：

と出力されますね。実際に実行してみます。

>java Test1
B
A

実は、**super.修飾子でスーパークラスのメソッドを呼び出した場合は、オーバーライドによるサブクラスのメソッドの呼び出しはされず、呼び出されたスーパークラスのメソッドが実行される**ことになっています。

上の例では、最初はクラスAのメソッド func()は実行されずにクラスBのメソッド func()が呼び出されて実行され、"B"が出力されます。その後の super.修飾子によるクラスAのメソッド func()の呼び出しでは、そのままクラスAのメソッド func()が実行され、"A"が出力されます。

もう一つの例を見てみましょう。

```
class A
{
    public void func1(){
        System.out.println("A1");
    }
    public void func2(){
        System.out.println("A2");
    }
}

class B extends A
{
    public void func1(){
        System.out.println("B1");
        super.func2();
    }
    public void func2(){
        System.out.println("B2");
    }
}

class Test2
{
    public static void main(String[] args){
        A aa=new B();
        aa.func1();
    }
}
```

この例は、先程の例のように無限ループの構造をしていませんが、やはり、super.修飾子で呼び出した場合は、オーバーライドは行われません。実際に実行すると、

```
>java Test2
B1
A2
```

となります。