Java 基础（10）：深入理解数组类型

1. 数组类简介

在 java 中，数组也是一种引用类型，即是一种类。
我们来看一个例子，理解一下数组类：

```
public static void main(String[] args) {
    Class c = int[].class;
    Class cIn = Integer[].class;
    Class ccIn = Integer[][].class;
    System.out.println(c.getName());
    System.out.println(cIn.getName());
    System.out.println(ccIn.getName());
    Class f = float[].class;
    Class flo = Float[].class;
    System.out.println(f.getName());
    System.out.println(flo.getName());
}
```

输出的结果：

```
[I
[Ljava.lang.Integer;
[[Ljava.lang.Integer;
[F
[Ljava.lang.Float;
```

上面的输出结果就是通过 Class 的 toString()方法输出的内容，可以看出规律，“[”表示一维数组，“**[[”则表示二维数组，如此类推下去。同时，如果是引用类型，则还要“[”后面还要跟一个”L“+类的全限定名。而如果是基本类型，则只要跟对应的大写字母。**

2. 数组类的分类

从上面的例子的输出可以看出，数组类是在 JDK 中是有分类的：对于一维的基本类型数组，输出”[“，而对于一维的引用数组则输出”[L“，多了一个”L“。
数组类可以分类可以分成两类：

- 基本类型的数组类；
- 引用类型的数组类；

这两种数组类的最大区别在于他们的祖先类不同。

2.1 基本类型的数组类

对于基本类型来说，基本类型数组类的父类一个，就是 Object 类。

```
public static void main(String[] args) {
    int[] a = new int[3];
    Object o = a;//编译通过，类型转换成功；
    //打印一下数组的超类
    System.out.println("int[]的 superClass 是： "+int[].class.getSuperclass());
}
```

运行结果：

```
int[]的 superClass 是： class java.lang.Object
```

2.2 引用类型的数组类

对于引用类型的数组类，其所有的祖先类除了 Object 类外，还包括下面所说的类：

如果 A 是 B 的祖先类，A[]也是 B[]的祖先类，其他维度也如此类推，不同维度间，没有任何关系。

看下面的例子：

```
public class Test_3 {
   public static void main(String[] args) {
       Children[] childrens = new Children[3];
       Ancestor[] ancestors = childrens; //编译通过，类型上转成功
       // 判断 childrens 是不是 Ancestor 或者是其子类的实例，进一步证明 Children[] 是 Ancestor[]的子孙类
       System.out.println("childrens instanceof Ancestor[] : " + (childrens instanceof Ancestor[]));
   }
}


class Ancestor{//祖先类
}


class Parent extends Ancestor{//父类，继承于 Ancestor
}


class Children extends Parent{//子类，继承于 Parent
}
```

运行结果：

```
childrens instanceof Ancestor[] : true
```

上面的例子中，因为 Ancestor 是 Children 的祖先类，所以 Ancestor[]也是 Children[]的祖先类。

如果这时候我们调用执行下面的代码：

```
System.out.println(Children[].class.getSuperclass());
```

输出的结果却是：

```
class java.lang.Object
```

咦，为什么引用类型数组 Children[]的父类是 Object 类，那么与上面所说的 Ancestor[]是 Children[]的祖先类起了冲突。因为如果父类是 Object，Object 类是根类了，那么祖先类就只有一个，就不可能再有其他的祖先类。

我们看一下 getSuperclass()的 API 描述：

```
public Class getSuperclass()
```

返回表示此 Class 所表示的实体（类、接口、基本类型或 void）的超类的 Class。如果此 Class 表示 Object 类、一个接口、一个基本类型或 void，则返回 null。如果此对象表示一个数组类，则返回表示该 Object 类的 Class 对象。

-返回：

此对象所表示的类的超类。

原来，**getSuperclass 对于调用者是数组类的对象的话，那么直接返回 Object 类的 class 对象。**

1) 在 Servlet 处理请求的方式为:　C

   A.以进程的方式

   B.以程序的方式

   C.以线程的方式

   D.以响应的方式

2) 多线程有几种实现方法,都是什么?同步有几种实现方法,都是什么?

   - 多线程有两种实现方法，分别是继承 Thread 类与实现 Runnable 接口
   - 同步的实现方面有两种，分别是 synchronized,wait 与 notify

3) 写一个 Singleton（单例模式）出来。

```
public class LazySingleton {
    private static LazySingleton instance = null;
    // 默认的私有的构造方法,保证外界无法直接实例化
    private LazySingleton() {}
    // 静态方法,返回此类的唯一实例
    public  static LazySingleton getInstance() {
        if (instance == null) {
```

```
            instance = new LazySingleton();
        }
    return instance;
    }

    public void pp(){
        System.out.println("lazy ok");
    }
}
```